Binary Freedom: Free Software, the Internet, and Activism in the Digital Age

Binary Freedom:
Free Software, the Internet, and Activism in the Digital Age

by

Christopher Bryan Campbell

A Dissertation

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

History

Lehigh University

August 2020

SAMPLE DISSERTATION SIGNATURE SHEET (REMOVE AND REPLACE ITEMS IN PARENTHESIS WITH YOUR INFORMATION)

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Christopher Campbell
Binary Freedom: Free Software, the Internet, and Activism in the Digital Age

Defense Date

Dr. John Smith
Dissertation Director

Approved Date

Committee Members:

Dr. Gail Cooper

Dr. Stephen Cutcliffe

Dr. Henry Korth

ACKNOWLEDGMENTS

The writing of this dissertation has been one of the most significant academic challenges that I have had to face. I'd like to thank my committee members, Dr. John Smith, Dr. Stephen Cutcliffe, Dr. Gail Cooper, and Dr. Henry Korth. Each of you have provided advice and assistance throughout the research and writing process. I am extremely grateful for your support and guidance. I appreciate the time and patience you've had with this despite having other competing priorities. I would also like to offer special thanks to Dr. John Pettegrew. Although he is no longer with us, his example continues to inspire his former students, especially myself.

I would like to acknowledge everyone who played a role in my academic accomplishments. My family and friends, and in particular, my wife and children who supported me with love and understanding. Without this, I would never have reached this point. Kim, our multi-decade plan is finally complete!

I would be remiss in not thanking Donald Johnston for setting me on this path twenty years ago, and to Dr. Thomas Legg for keeping me focused on my PhD and recommending Lehigh's program.

I'd also like to thank all the folks from the Free Software Community that I consulted with during this effort, including Ken Coar, Emmett Plant, Marcia Wilbur, and Russ Pavlicek. Special thanks to Ken & Den and all the real innovators for your promethean gifts. And finally, thank you to RMS for giving me something to write about and to ESR for making it interesting.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In the 1970s, software emerged as a distinct industry as it became unbundled from computer hardware. Corporate interests such as Microsoft commoditized software by restricting access to source code and introducing licensing agreements to limit the rights of software consumers. The Free Software Movement reacted to this by collaboratively creating software free from the restrictions of commercial license agreements. As free software, such as Linux, gained popularity, programmer Eric Raymond re-articulated the movement as Open Source, a programmer-centric software development model. This re-casting sought to supplant the movement's consumer freedom focused ideology with a model that favored corporate approval. A schism emerged within the movement, and free software ideologues gravitated toward individual rights-based activism. Even as the Free Software Movement splintered, its distributed collaboration model was transposed to other cultural works, and its ideology informed later activist groups, such as the Cypherpunks and WikiLeaks.

INTRODUCTION

In September 1983, MIT Artificial Intelligence lab employee Richard Stallman felt faced with a "stark moral choice."[1] Stallman, an expert programmer, was unsettled by the changes he saw in computing culture. Software, which was previously written and shared freely, was now transitioning into intellectual property, a proprietary asset to be sold. Stallman increasingly found himself in conflict with corporations in the nascent software industry. As a result, he developed an aversion to corporations that restricted consumers from accessing, altering, or, in some cases, even discussing the programs they owned. For Stallman, the choice was clear: he could either chose to overcome his objections to proprietary software or he could commit to creating a non-proprietary system of software.

Where existing literature has sought to understand the culture and motivations of Open Source developers or to quantify the corporate acceptance of their software, this dissertation will focus on the ideologically driven Free Software Movement. The purpose of this study is to understand the environment in which the Free Software Movement emerged and to examine how both its practices and ideas transitioned into broader society. This paper will argue that Free Software emerged as a reactionary response to the cultural changes that occurred as software was commodified. Later, the Open Source reinterpretation of Free Software sought to supplant the movement's ideological basis with a simple, corporate-friendly development model. This reinterpretation served to splinter the movement just as the use of Free Software became ubiquitous. This

---

[1] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software* (Sebastopol, California: O'Reilly, 2002), 101.

dissertation will contend that the ideas and schemas put forth by the Free Software Movement were later transposed to other projects and groups. In the process, they informed the creation of activist hacking, or hacktivism, as well as the crypto-anarchist Cypherpunk group, the latter of which extended the ideas and use of Free Software to further their subversive geopolitical goals.

This dissertation examines the historical context from which Richard Stallman's Free Software Movement emerged. Software sharing occurring in the early computing industry was self-serving. The nascent computing industry lacked sufficient numbers of trained workers, so computer manufacturers composed and shared to cooperatively utilize the limited pool of trained programmers. It was in this culture that Richard Stallman was introduced into computing. When the commoditization of software changed these established practices, Stallman created Free Software as a reactionary response. His decision to compose and share programs was a conscious effort to oppose the commercialization of software. Stallman's first Free Software effort, the GNU project, served two purposes: First, it provided a programming environment outside the realm of corporate control. Second, it provided a mechanism by which he could spread his civil libertarian values, which held software to be the same as speech, requiring the same protections, and the same freedom from censorship or control.

Two conceptual tools are useful to understand the subsequent evolutions and transpositions of Richard Stallman's Free Software. The first is *ideology*, which refers to a broad set of ideas about the orderings of politics or society. Naturally, this term can have differing meanings, and even pejorative associations when employed by Marxist scholars. So, it should be understood that the use of this term within this study is intended to be neutral, in the manner used by historians such as John Kasson, where ideology is "the configuration of assumptions, values, and beliefs," by which a group can "interpret and transform their political and social relationships."[2] The second is *schema*, which as defined by Merriam-Webster, is simply a "structured framework."[3] Within the context of this research, this specifically refers to the distributed, peer-based production model employed by Free Software programmers.

The chapters that follow detail many examples that show the evolution of the movement, as well as the transposition of both its ideology and schema. The Free Software Movement played a pivotal role in the development of the commercial Internet, where Free Software programmer Bill Joy refined the network's common protocol stack, and his fellow UNIX programmers wrote Free Software such as BIND and Sendmail, which emerged as de facto standards for the burgeoning global network. The Free Software Movement also was critical in enabling the growth of the World Wide Web. In 1991, CERN (Conseil Européen pour la Recherche Nucléaire) researcher Tim Berners Lee created the world's first web browser, server, and web page. CERN realized that Berners Lee's work could be endangered by corporate co-option, so the organization released the code licensed as Free Software in 1994.[4] By this point, both the World Wide Web and other Free Software started to become widespread. University of Helsinki student Linus Torvalds' Free Software version of UNIX, Linux, quickly gained

[2] John F. Kasson, *Civilizing the Machine: Technology and Republican Values in America, 1776-1900* (New York,New York: Hill and Wang, 1999), 3.

[3] "Schema," Merriam-Webster, accessed December 3, 2019, https://www.merriam-webster.com/dictionary/schema.

[4] Tim Smith and François Flückiger, "Licensing the Web," CERN, March 12, 2014, http://home.web.cern.ch/topics/birth-web/licensing-web.(accessed January 23, 2018).

popularity, and companies such as Red Hat were founded to support it.[5] The Free Software web server Apache emerged as the dominant platform for hosting web pages.[6] The relationship between the World Wide Web and Free Software was symbiotic: much of the web was hosted on Free Software, and the web provided the platform for the development and distribution of the programs.

The late 1990s saw the rapid expansion of Free Software as well as the splintering of the movement along ideological lines. In 1997, an essay by programmer Eric S. Raymond discussed the distributed development practices of the Free Software community. Entitled "The Cathedral and the Bazaar," the text contrasted the open, distributed software development model of Free Software (the Bazaar) to the closed, hierarchal structures of corporate software development (the Cathedral). Raymond's essay was popular in the technical industry and encouraged the Netscape Corporation, which had been competing with Microsoft for the web browser market, to release the source code for their web browser. Netscape's leadership conferred with Raymond prior to their decision to release their code, and Raymond saw Netscape's planned public announcement as an opportunity for the burgeoning movement.[7] He advocated for the Free Software Movement that grew out of Richard Stallman's GNU project to be re-branded as "Open Source." Raymond was weary of the movement's "ideological conformity" with Stallman, and he sought to increase mainstream corporate adoption.[8]

---

[5] Peter H. Salus, *The Daemon, the GNU, and the Penguin: How Free and Open Source Software is Changing the World* (Marysville, Washington: Reed Media Services, 2004), 1.115.

[6] "April 2014 Web Server Survey," Netcraft, April 2, 2014, http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html. (accessed March 14, 2015).

[7] "History of the OSI," The Open Source Initiative, September 2012, http://opensource.org/history.

[8] Eric Raymond, email to Chris Campbell, March 1, 2015

By focusing on how the distributed development of software could benefit a corporation, Raymond was able to exorcise Stallman's emphasis on preserving individual freedoms.[9]

The .com bubble of the late 1990s brought wealth to some Open Source adherents such as Eric Raymond, and the use of Open Source software became widely accepted in corporate computing.[10] Meanwhile, Free Software programmers increasingly found themselves in active protest against emerging legal challenges, including the Digital Millennium Copyright Act signed into law in October 1998. The legislation significantly changed the copyright protections for digital media, limiting personal use of copyrighted content and enabling corporations to restrict media, such as DVDs or eBooks, to certain types of players.[11] The situation came to a head as programmers began to be arrested for writing software. In early 2000, the Motion Picture Association of America (MPAA) used the law against Free Software programmer John Lech Johansen, who had developed a DVD player program for the Linux operating system. In March 2000, the MPAA pursued the magazine *2600* for printing Johansen's source code. Similarly, in June 2001, programmer Dmitry Sklyarov was arrested by the FBI for composing an eBook reader for Linux. The use of the copyright law to crack down on Free Software programmers drove the movement's ideologues to the streets in protest.[12]

Activist groups within the Free Software Movement also began to transpose the movement's ideas into other arenas. Hacktivismo, an offshoot of the "cracking" group the Cult of the Dead Cow, developed a custom Free Software license and began to release

---

[9] Eric Raymond, "Goodbye, 'Free Software'; Hello, 'Open Source'," Catb.org, February 8, 1998, http://www.catb.org/esr/open-source.html, (accessed May 4, 2016).

[10] Eric Raymond, "Surprised by Wealth," Linux Today, December 10, 1999, http://www.linuxtoday.com/infrastructure/1999121000105NWLF. (accessed January 23, 2018).

[11] Marcia Wilbur, *A Decade of the DMCA* (Middletown, Delaware: lulu.com, 2009), 31-33.

[12] Ibid., 47-49, 60-63.

software to assist human rights activists throughout the world.[13] The online activist group, Anonymous gained notoriety in 2008 when they began to use Free Software programs to overwhelm the online resources of organizations and nations to which they were opposed.[14] Meanwhile, a group of Free Software programmers, the Cypherpunks, extended Stallman's ideology and schema toward their mistrust of governments. Centered around a mailing list hosted at the Free Software support company, Cygnus Solutions, the group sought dramatic societal change using technology. According to their manifesto, the group mistrusted governments and were dedicated to writing Free Software that defended privacy by creating encrypted structures removed from government oversight or control.[15] The group's ideals mirrored those of Stallman as did their solution; as with Stallman, they were focused on freedom of expression and they saw proprietary, closed systems as a form of control. However, where Stallman mistrusted corporations, this group mistrusted governments. As Stallman had done with software, the Cypherpunks aimed to create a dominion separate from, and unreachable by, government and corporate control. Their innovations included cryptocurrencies like Bitcoin, Darknets, and groups such as WikiLeaks.

The transposition of Free Software's ideology, schema, and tools resulted in structures that have influenced events far beyond software copyright concerns. Cryptocurrencies like Bitcoin have threatened the role of governments in regulating commerce by presenting a model where the Federal Reserve could be replaced by a

---

[13] Cracking is the practice of breaking into a computer system. This is commonly confused with hacking, which is the practice of developing clever solutions, or "hacks" for a technical problem.

[14] Ibid.,156-159.

[15] Eric Hughes, "A Cypherpunk's Manifesto," Activism.Net, March 9, 1993, https://www.activism.net/cypherpunk/manifesto.html, (accessed January 24, 2018).

computer. Private encryption has allowed for the creation of networks outside the reach of government monitoring. Groups, such as WikiLeaks, emerged as geopolitical players due to their ability to acquire and selectively disseminate classified and confidential information.

Existing Scholarship

Academic historiography on Free Software is sparse, and existing scholarship privileges the eventual corporate acceptance of Eric Raymond's Open Source development model at the expense of Free Software's foundational beliefs. Ignoring this aspect of the movement blurs the distinction between the Free Software Movement and the later Open Source development model. As a result, the relationships between the Free Software proponents and later Internet-based activists have also remained unexamined by scholars.

Political scientist Steven Weber's 2005 book, *The Success of Open Source,* provides a history of UNIX and Free and Open Source software, with a focus on Eric Raymond's Open Source reinterpretation. He perceives Open Source as a method of organizing production, and he specifically avoids community ethnography and ideological discussions.[16] For Weber, Open Source's success is in its efficacy as a process and in the abilities of businesses to utilize the Open Source model. Consistent with this business-centric focus, the author ignores the role of Free Software's foundational beliefs. For example, when discussing individual motivations, Weber downplays data that does not fit his thesis. The author states that the motivation for

[16] Steve Weber, *The Success of Open Source* (Cambridge, Massachusetts: Harvard University Press, 2004), 224-225.

“many Open Source developers” was doing battle with a “joint enemy,” which he suggests was Microsoft. To support this assertion, he references a Boston Consulting Group Survey that cites 11.3 percent of the respondents as suggesting this opposition as a primary motivation for their software development work. However, in the same survey, 34.2 percent of respondents reported their motivation as ideological, in that “code should be open.” The author acknowledges that this reflects an ideological commitment, but he dismisses this as inconsistent with the “observed practices of most Open Source users.” In this dismissal, he also fails to differentiate between Free Software developers and users.[17]

Anthropologist Christopher Kelty’s *Two Bits: The Cultural Significance of Free Software* also ignores the foundational beliefs of Free Software. Kelty’s study of the culture reduces it to “an international community of geeks brought together by their shared interest in the Internet.”[18] He provides an ethnographic discussion of the community of developers, which he describes as a “recursive public,” one "constituted by a shared concern for maintaining the means of association through which they come together as a public.” Kelty’s cultural model fails to account for Free Software users who have no active role in maintaining the means of association that is key to his definition.

Like Weber, Kelty focuses his narrative on Eric Raymond’s corporate-friendly Open Source development model. His discussion of Richard Stallman is limited and focused on the license complications that Stallman encountered in his development of the

[17]Steve Weber, *The Success of Open Source* (Cambridge, Massachusetts: Harvard University Press, 2004), 224-225, 139-140.

[18] Christopher Kelty, *Two Bits: The Cultural Significance of Free Software*, Experimental Futures (Durham, North Carolina: Duke University Press, 2008), 5.

GNU Emacs software.[19] In the process of detailing the schism between Free Software and Open Source, Kelty accuses Free Software ideologues as "kindling flames of worry over intellectual-property expansion."[20] Kelty rejects the role of ideology, determining that "Free Software is not an ethical stance," and its source-accessible progenitor, the UNIX operating system was "never free in any sense."[21] This conclusion is problematic. Richard Stallman's conception of free is not based on price, but rather the freedom to modify and reproduce the software. The availability of UNIX's source code allows for both freedoms. Because he overlooks the foundational ideas of Free Software, Kelty is challenged to understand the role of UNIX in broader history.

Kelty's focus is on Free Software's eventual re-interpretation as Eric Raymond's Open Source development model - and the corporate acceptance thereof, which is why the foundational ideas and beliefs of the movement remain unexplored. Part of this oversight results from Kelty's cultural review focusing on the software developers rather than examining the software's user community. Since Free Software emerged from concerns over the abrogation of software consumer rights, this aspect must be considered as well. Looking solely at the producers of the software clouds the historical conception of Free Software, ignores its philosophical origins, and obscures its connection to Internet activism.

There are several non-academic texts that explore facets of this history. In his 1985 book, *Hackers: Heroes of the Computer Revolution*, journalist Stephen Levy

---

[19] Emacs was a text editor and the first software ported to the GNU project. The acronym stands for "Editor MACroS." See "The GNU Emacs FAQ: Where Does the Name "Emacs" Come From?" GNU.org, https://www.gnu.org/software/emacs/manual/html_mono/efaq.html#Origin-of-the-term-Emacs, (accessed November 4, 2017).

[20] Kelty, 99.

[21] Ibid., 307-308.

provided a look at the early computing industry from oral histories and perspectives of the eponymous hackers. Levy traces the origin of the term hacker to the MIT Tech Model Railroad Club (TMRC) in 1959. The term referred to an individual who applied cleverness and determination to create innovative customizations (hacks) for electronic equipment such as train switches and relays. As the TMRC group moved into working with computers, the terminology transferred to the new electronic medium. According to Levy, an ethic associated with hacking emerged at MIT. It was founded on the beliefs that access to computers should be unlimited and total, all information should be free, and authority should be mistrusted. The ethic also held that hackers should be judged by their hacking, as it is possible to create beauty and art on a computer.[22]

Levy traces the history of those following his observed ethic and, in the process, touches on some of the hackers who came to be very influential in the computing industry, including Bill Gates, Steve Wozniak, and Richard Stallman. Levy's chronology traces the hacking effort from its epicenter in Cambridge in the 1950s and 1960s out to the Bay Area in the 1970s. The text, written in the early 1980s, ends back in Cambridge in 1983 with Richard Stallman, the "last of the true hackers."[23] The author's account of Stallman ends where Free Software begins, with Stallman departing MIT to create the GNU project.[24]

In his 2006 text, *What the Dormouse Said*, journalist John Markoff asserts that the 1960s counterculture shaped personal computing. Markoff ponders why the PC and the PC industry emerged on the West Coast rather than elsewhere, such as from the computer

---

[22] Levy, *Hackers* (Sebastopol, California: O'Reilly, 2010), 28-32.
[23] Ibid., 437.
[24] Ibid., 450.

hackers at MIT.[25] The author suggests that the mainstream culture failed to perceive computers as a new medium of personal consumption. Markoff includes Free Software's Richard Stallman in his discussion of Levy's hacker ethics. He discusses the topic with Brian Harvey, an MIT AI lab hacker who migrated to Stanford's Artificial Intelligence Lab (SAIL). Harvey's view of hacking emanated from aesthetics rather than ethical precepts. The author puts this interpretation in dialog with MIT AI lab hacker Richard Stallman and his notion that information should be free.

In many ways, Markoff produced a sequel to Levy's study. Markoff even uses Levy's interviews as sources, in the process revealing information that Levy had not published. Many of the interviews cited were those with counterculture publisher Stewart Brand. Brand was also the primary focus of Fred Turner's *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism.*[26] In this 2008 book, Turner argues that the counterculture was influential to the "New Communalists," who advocated for the Internet as a form of digital utopianism. While Markoff was suggesting that the 1960s counterculture influenced the development of the personal computer, Turner suggests that the counterculture influenced the adoption and use of the Internet.

In 1972, Brand wrote an article for the Rolling Stone, *SPACEWAR!* In the article, Brand discusses the popularity of the Spacewar! computer-based video game. To research his article, Brand interacted with the developers at Xerox's Palo Alto Research Center

[25] John Markoff, *What the Dormouse Said: How the Sixties Counterculture Shaped the Personal Computer Industry* (New York,New York: Penguin Books, 2006), xv.

[26] Fred Turner, *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism*, 60265th ed. (Chicago, Illinois: University of Chicago Press, 2008), 213-214.

(PARC), and he briefly highlights the newly emerged ARPAnet, precursor to the commercial Internet.[27] Brand's interest in computers grew when he interacted with Douglas Engelbart, an ARPA funded researcher who worked in the Stanford Research Institute's Augmentation Research Center (ARI).[28] Brand had performed media services for Engelbart during a demonstration of the ARI's technologies in December 1968. Many years later, Brand attended the first "Hacker's Conference" where he encountered many computer industry giants including Apple's Steve Wozniak and Richard Stallman.[29] This 1984 conference was prompted by the publication of Levy's book, and the author hosted the forum.[30]

In *From Counterculture to Cyberculture,* Turner does not directly address Free Software. He does mention Richard Stallman, but only to set him as a hacker at the 1984 conference.[31] Turner details the birth of the Electronic Frontier Foundation (EFF), a group which has frequently provided legal defense funds, and written amici curiae for Free Software related cases.[32] However, Turner's explanation of the EFF is only as an Internet advocacy group, and the author does not detail the relationship between the organization and Free Software. Turner's text also mentions, but does not discuss in any meaningful way, EFF co-founder John Gilmore. Gilmore founded the Free Software support company Cygnus Solutions[33] and cofounded the Cypherpunks.

---

[27] Stewart Brand, "Spacewar!" *Rolling Stone*, December 7, 1972, 9.

[28] The Advanced Research Projects Agency (ARPA) was a U.S. Department of Defense agency responsible for developing technologies for U.S. military consumption.

[29] Turner, 136.

[30] Ibid., 137.

[31] Ibid., 136.

[32] Ibid., 172.

[33] Keeping with the GNU's recursive naming, Cygnus stood for "Cygnus - Your GNU Support."

There are also several books that focus on specific technologies and figures. In *The Daemon, the Gnu, and the Penguin: How Free and Open Source Software Is Changing the World*, linguist Peter Salus provides a broad account of the history of various free and Open Source operating systems.[34] Salus provides a general narrative of the history of UNIX and the Free Software operating systems that followed it. The text alternates between history and advocacy, sourced material and first-person accounts. Despite the title promising discussion on how free and Open Source software is changing the world, the text limits itself to only discussing the proliferation of Open Source software.[35]

In his biography of Richard Stallman, *Free as in Freedom: Richard Stallman's Crusade for Free Software*, author Sam Williams provides a fascinating exploration of the founder of Free Software. Williams's narrative outlines the events that led Stallman to establishing the GNU software project and the Free Software Foundation, a non-profit organization founded to support the Free Software Movement. The text provides no insight into the subsequent evolution of Free Software or of the rise of related activism. For example, the Digital Millennium Copyright Act only warrants a passing mention as something that Stallman had written about on his webpage.[36] Likewise, since the text is focused solely on Stallman, it does not consider how his ideas informed other groups such as the Cypherpunks.

---

[34] The title of the text is a reference to the mascots of various operating systems and programming efforts. The daemon is BSD Unix's mascot, GNU refers to the Richard Stallman's GNU project, and the Penguin is the symbol for Linux. See Alaska Miller, "The Valley Goes Nuts for Cute Fuzzy Things," Business Insider, November 12, 2009, https://www.businessinsider.com/open-source-mascots-2009-11, (Accessed 7/15/2020).

[35] Turner, 161-165.

[36] Williams,73.

In his autobiography, *Just for Fun: The Story of an Accidental Revolutionary*, Linux creator Linus Torvalds (assisted by author David Diamond) details the creation of the Linux operating system and the subsequent world-wide adoption of the software.[37] The text features Torvalds's opinions on Free Software's ideas and beliefs. He identifies himself as a pragmatist, in opposition to idealists like Richard Stallman. Torvalds rejects the zealotry he perceives in Free Software ideologues, dismissing the views as absolutist devotion bordering on religion.[38]

All this literature tells different parts or interpretations of the larger story of Free Software, but none paints a complete picture. A common thread missing in most of the histories of Free Software is ideology. Levy's text introduced the hacker ethics. Richard Stallman mirrors these ideals in the foundation principles of the Free Software foundation. Likeminded individuals adopted Stallman's model and philosophy, creating their own Free Software projects and fueling the spread of these ideas. The movement of Free Software users and creators grew and formed the basis for later Internet-based activists, groups such as Anonymous and WikiLeaks. Ignoring the ideological aspect of the group, as these texts have, results in these relationships remaining unexplored.

Chapter Summary

[37] Specifically, Torvalds created the kernel for the operating system. The kernel is the core of a computer's operating system, acting as the intercessor between applications and a computer's CPU, memory, and hardware devices. Where the kernel interfaces directly with the hardware, the rest of the operating system broadly supports the rest of the computing environment, providing tools for compiling software and managing the machine.

[38] Linus Torvalds, *Just for Fun: The Story of an Accidental Revolutionary* (New York, New York: Harper Business, 2001),164, 194.

The first two chapters review early computer history, highlighting programming and the emergence of software and the commercial software industry. Corporations such as International Business Machines (IBM) and the Digital Equipment Corporation (DEC) formed user groups to facilitate the development and sharing of software for their respective computing platforms. This cooperative – but self-serving - approach was designed to address the shortage of programmers in the nascent industry. The open computing culture in this early period would later inform the development of the UNIX operating system at AT&T's Bell Labs as well as Dartmouth College's popular BASIC programming language. In 1969, IBM separated their lines of business into hardware, software, and services in response to a U.S. Department of Justice anti-trust challenge. This move by the period's dominant computer vendor helped establish the commercial software industry. With the advent of personal computing, the Micro-Soft Corporation emerged to sell a version of Dartmouth's BASIC that would run on these primitive machines. The company sought to commoditize software through eliminating the sharing of their programs. The efforts of companies like Micro-Soft were supported by the legal changes brought by the Copyright Act of 1976, which introduced protections for software. Microsoft later became a software industry juggernaut due to their close relationship with IBM during their introduction of the personal computer. By the mid-1980s the corporate software culture had been changed, closed, and commoditized. Even historically inexpensive and open software such as UNIX came to be closed, re-packaged, and sold at great cost.

The commercialization of software brought unwelcome changes to the world of Richard Stallman in his role as a programmer at the Massachusetts Institute of Technology (MIT) AI Lab. Chapters three and four introduce Richard Stallman and discuss how the malevolent actions of a corporation associated with the AI lab drove the young programmer to introduce the Free Software GNU project – and how the later appropriation of his software by a graduate student led Stallman to invent a software license that would protect his GNU project. Using these legal tools, Stallman was able to preserve the freedoms he felt that corporate entities were abrogating, imbuing the software products with his own anti-authoritarian, libertarian ideologies. Stallman's ideas spread to other similar, academic licenses that provided comparable protections for community software. These chapters also consider Stallman's ideological views, as articulated in his GNU project announcement and GNU manifesto, as well as the public reaction to Stallman's proposals as expressed in computer newsgroups and business magazines.[39] The section ends with the founding of the non-profit Free Software

---

[39] A note about sources: This dissertation looks at the history of various Internet-based groups outside of the mainstream culture. As such, sources include emails to mailing lists and UseNet groups. Although these are cited in a manner consistent with the source type, as done by other researchers (e.g. Kelty), individuals who are unfamiliar with Usenet software or mailing lists may be challenged in reviewing the sources through these mechanisms. To this end, it should be mentioned that this information can also easily be found in other formats on the Internet. Many UseNet groups have been archived on Google groups (e.g.comp.os.minix on https://groups.google.com/forum/#!forum/comp.os.minix) and on other websites (e.g. net.emacs on http://www.megalextoria.com/usenet-archive/news005f1/b12/net.emacs/.) Message archives also tend to exist with each project. For example, Emacs and GNUs messages can also be found via the GNU project https://lists.gnu.org/mailman/listinfo, Debian messages can be found via the Debian project (https://www.debian.org/MailingLists/) and so on. This correspondence is frequently archived online by other interested parties as well, as is the case with the Cypherpunks list(s) (http://mailing-list-archive.cryptoanarchy.wiki/) and WikiLeaks (http://cryptome.org/wikileaks/wikileaks-leak.htm). Universities like Stanford have archived issues of printed materials including niche periodicals like the Homebrew Computer Club newsletter, but these magazines can also be found online. Computer focused sites like http://www.bitsavers.org/ have archived many publications, and other more general sites, like https://archive.org have them as well.

Foundation and discusses how the group, along with Stallman's software licenses, facilitated the GNU project's growth into the Free Software Movement.

Chapters five and six describe the creation of the Internet and its symbiotic relationship with Free Software. The network emerged in the 1970s from the Defense Department's Advanced Research Projects' ARPAnet. Several years after its introduction, the ARPAnet began to experience expansion issues due to the incompatibility of proprietary software used by different vendor networks. This led to the formation of an internetworking group that led to the eventual creation of the Internet protocol, TCP/IP. In 1991, programmer Tim Berners-Lee developed the HTML protocol to foster collaboration between researchers at CERN, the European Organization for Nuclear Research. By 1994, this platform, known colloquially as the World Wide Web, was released as Free Software. Early web technologies, such as the National Center for Supercomputing Applications' (NCSA) web server and browser informed later Free Software such as Apache and the Netscape Web Browsers. By this point, the Free Software Linux operating system had begun to spread, and with it, emerged a vibrant community whose members used the Internet to coordinate development of Free Software projects. In January 1998, a group of Free Software entrepreneurs sought a corporate-friendly rebranding of the movement. Led by self-appointed leader Eric Raymond, the Open Source Initiative (OSI) encouraged the community to market their Free Software as "Open Source," serving to remove Richard Stallman's ideology from the software. Raymond's writings, such as the *Cathedral and the Bazaar*, reinterpreted Free Software as simply a development model for programming, dismissing the civil libertarian user freedoms espoused by Stallman. Raymond's public antics and heavy-handed campaign to extirpate Free Software's ideological basis did not sit well with the anti-authoritarian Free Software community, so the rebranding produced a schism within the group. The resulting ideological stratification served to divide the movement just as the Open Source re-branding brought about widespread corporate adoption. The success of Open Source with corporations such as Google brought interest from researchers in economics and sociology. At the same time, both the schema and ideology of Free Software began to spread into other arenas.

Chapter seven reviews the academic interest and the subsequent spread of Free Software ideas into legal defense and publishing, leading to the creation of Wikipedia and democratized publishing platforms such as Word Press. The growth of the movement was soon endangered by the Digital Millennium Copyright Act of 1998 (DMCA), which criminalized the composition of certain programs and was subsequently used as a tool of suppression by corporations. Researchers like Princeton's Dr. Edward Felten soon found their research articles under threat of censorship by groups citing the new legislation. The use of the law to arrest programmers and to justify court rulings that forbid the distribution of source code for programs brought the Free Software community into the streets in protest. Groups like the Electronic Frontier Foundation (EFF) coordinated outreach to computer corporations like Adobe, as the Linux User Groups (LUGS) helped plan demonstrations.

In the first part of chapter eight, the transition from Free Software to online activism, or "hacktivism" is reviewed. Texas-based group The Cult of the Dead Cow's

Hacktivismo project focused on writing Free Software tools for use by human rights activists. The group modified a Free Software license to create their own Hacktivismo Enhanced-Source Software License Agreement (HESSLA), which was deliberately designed to protect the ideological objectives of Richard Stallman's Free Software Movement, while making allowances designed to preserve the human rights of any activists who might use the software. The idea of hacktivism was brought to the public with the on-line group, Anonymous, whose high-profile website defacements and Guy Fawkes-themed public protests brought media attention to their attacks. In the second section of chapter eight, the transposition of Free Software ideologies takes a more sinister turn. It opens with a brief history of public key cryptography and details how the U.S. government sought unrestricted access to private encrypted communications. Programmers expanded the ideology and schema behind Free Software to inform the creation of the cryptography-focused group, the Cypherpunks. Just as Stallman sought to use Free Software to create a sphere free of corporate domination, the Cypherpunks developed Free Software to create an encrypted realm free of governmental oversight or control. The group pursued what co-founder Timothy May called "crypto-anarchy," a vision of radical social transformation that would come about through the use of the cryptography-focused Free Software.[40] Openly anti-democratic and ambivalent about their tools helping parties like the Russian Mafia and Aryan Nation, the group sought "to use encryption to undermine the so-called democratic governments of the world."[41] The

[40] Tim May, "The Cyphernomicon," September 11, 1994, https://ia600208.us.archive.org/10/items/cyphernomicon/cyphernomicon.txt (accessed January 21, 2020).
[41] Ibid.

chapter ends by reviewing the group and the fruits of their labor: Cryptome, Darknets, WikiLeaks, and Bitcoin.

Summary

In the history of computer software development, there has been a tension between the concept of software as a discrete corporate product and software as a communal set of instructions available to anyone to use or alter. When computers emerged as big business, industry dominating corporations such as IBM and Microsoft sought to “close” software, mandating that the programs’ user cannot access or alter the underlying code. Focusing on the rights of users, Richard Stallman opposed these changes by developing a programming schema and licensing model designed to preserve the early communal software culture by producing software that was free from corporate control.

Although computers have become a very large and powerful industry, the ideology of Free Software that had arisen in the early days of computing has proven to be a powerful countervailing force that has survived to the present day. The intellectual descendants of the Free Software Movement have expanded the ideology and its objectives, leveraging Free Software as a tool to alter the structure of society itself. To fully understand the complexity of these actions, it is important to comprehend the ideologies and evolutions that have brought us to this point.

CHAPTER 1: FROM HARDWARE TO SOFTWARE

Modern computers are the product of a long lineage of technology being developed, shared, and incrementally improved. Computing hardware has existed since antiquity; the counting frame, a precursor of the abacus, emerged in Mesopotamia around 2500 BC. Gear-based mechanical computers, like the Grecian Antikythera mechanism, emerged by the 1^{st} century BC and were used to calculate celestial events. These early calculation devices lacked both programmability and storage; their utility was limited to performing only the specific calculations for which they were constructed. With the advent of data storage mechanisms such as punched cards in the 19^{th} century, mechanical devices acquired the ability to work off a stored set of operating instructions. By the early 20^{th} century, electromechanical computing device operators composed programs on punched cards and tape. Software, stored programs on a fully electronic computer, did not appear until 1949, after a long chain of technological evolution.

This chapter reviews the emergence of software as independent from hardware and demonstrates that there was a collaborative software development and sharing culture present at all levels of early computing from mainframes, to minicomputers and microcomputers. On early computers there was no conceptual division between the hardware and the programs that ran on it. Frequently, early computers such as ENIAC were programmed through the manipulation of the hardware. Even when software emerged, it was perceived as an accessory to computer hardware, rather than as a distinct product with intrinsic value. As the importance of programming began to be understood by computer manufacturers, the lack of trained programmers and software distribution methods in the nascent industry resulted in a labor shortage that fueled interest in sharing software. Corporate user groups such as the Digital Equipment Corporation's User Society (DECUS) officially promoted and facilitated the sharing of software. Unendorsed sharing was also prevalent. This was the case with AT&T Bell Lab's UNIX operating system.[1] Here, a software sharing community emerged to address the support restrictions placed on the system by AT&T. In some situations, sharing was ideological; the BASIC programming language was intended to further the integration of computers in society by lowering the barriers between man and machine. Regardless of developer motives, a sharing or open software culture dominated the first quarter century of electronic computing. From IBM mainframes to DEC minicomputers, and even down to consumer microcomputers, the development and sharing of software was common practice.

From Hardware to Software

Early computing was done on hardware created for specific tasks. An early example of this is inventor Herman Hollerith's tabulating machine. Developed for the 1890 census, Hollerith's tabulating machine used punched cards featuring pre-determined locations for recording data such as age, gender, data of birth, and nationality. Operators would input data by punching holes in these designated fields. The completed cards were then loaded in the tabulating machine which would then calculate the totals by counting the holes in the cards by pressing them against 288 spring-loaded pins. When pushed down on the card, any of the pins that passed through a hole completed an electrical circuit, causing the counter for that dataset to advance by one. The tabulator also included a sorting box for depositing the cards after they were counted. The sorter contained 24

[1] An operating system is software that supports a computer's basic functions, such as interacting with the computer's hardware and providing a working environment for the end user(s).

lidded compartments and the tabulator could be configured to automatically open the lid on a given compartment. This allowed for calculations to be done via sequential passes through the machine.[2] This electro-mechanical data processing featured programming done through selective feeding and re-feeding of data. For example, if operators wished to count all 40-year-old men in a dataset, they could set the tabulator to sort the cards by gender. The cards sorted to the compartment for males could then be run through the tabulator again, this time sorting and counting by age. Hollerith's tinkering with punch cards also led to another contribution to computer history: in 1896, he founded the Tabulating Machine Company, progenitor to International Business Machines (IBM).[3]

The work of Herman Hollerith informed the creation of IBM's Automatic Sequence Controlled Calculator (ASCC) in 1944.[4] Colloquially known as the "Harvard Mark 1," the electro-mechanical computer was constructed under the direction of Harvard professor Howard Aiken. The Mark 1 was the first fully automatic computer. Once programmed it could run without interruption. The machine was programmed through codes punched into holes on 3-inch-wide paper tape. Early computer pioneer Grace Murray Hopper used the tape-based system to introduce many of the mainstays of computer programming: An early form of subroutine, a reusable set of instructions, was achieved through the re-use of previously used section of paper tape.[5] For programs requiring instructions to be re-run multiple times, sections of paper tape could be attached

[2] The sorting mechanism meant that decreasing demographic statistics could be sorted.

[3] Thomas J. Watson, *Father, Son and Co.: My Life at IBM and Beyond*, 1990 (New York, New York: Bantam Books, 2000), 14.

[4] Kurt Beyer, *Grace Hopper and the Invention of the Information Age*, Lemelson Center Studies in Invention and Innovation (Cambridge, Massachusetts: MIT Press, 2009), 126-129.

[5] Ibid. 95.

to one another create processing loops.[6] The U.S. Navy funded Aiken's laboratory as part of the war effort and as such, Hopper and the other lab employees were service members within the branch. The laboratory was run accordingly, with Lieutenant Hopper and Ensign Robert Block developing a strict process for the submission, processing, and testing of computing jobs. The lab ran 24 hours a day, incremented in shifts equivalent to a watch on a naval vessel.[7] Their regimen was a precursor of batch computing, where programmers develop and submit a program to the computer's system operators.[8] The jobs were queued and submitted when computing time was available. The Mark 1 computer was only in service for three years but helped shape the course of computer programming, and it launched the careers of the first computer programmers.[9]

The electro-mechanical Harvard Mark 1 was quickly eclipsed by fully electronic computers. In February 1946, University of Pennsylvania researchers John Mauchly and J. Presper Eckert introduced a general purpose electronic computer called ENIAC (Electronic Numerical Integrator and Computer.) The computer was designed to calculate firing tables for the U.S. Army's Ballistic Research Laboratory at Aberdeen Proving Ground.[10] Following the delivery of the gigantic machine, the researchers immediately began efforts on a revised version, EDVAC (Electronic Discrete Variable Automatic Computer.) Conceived during the construction of ENIAC, the EDVAC would address the shortcomings of Mauchly and Eckert's first computer and introduced

---

[6] A loop is a sequence of instructions that are repeated within a program. In modern computers, a loop would typically run until a preset criterion is reached. The Mark 1 could not, however, handle conditional logic. This makes it less capable than Charles' Babbage's engine which inspired it. See Martin Campbell-Kelly, 58.

[7] Beyer, 61.

[8] The term 'batch computing' is frequently used interchangeably with 'batch processing.'

[9] Beyer, 196.

[10] Ibid., 117-119.

innovations such as stored programs. While still in development, the computer plans were detailed in University of Pennsylvania consultant John von Neumann's *First Draft of a Report on the EDVAC*. The premature release of the report helped stymie Mauchly and Eckert's efforts to patent the device.[11] It also disseminated their ideas to other researchers working on early computers.[12]

One researcher who received a copy of the *First Draft of a Report on the EDVAC*, was Cambridge physicist Maurice Wilkes. The document laid out the principle design of the modern digital computer and it inspired Wilkes, who was trying to re-establish the computing laboratory at Cambridge, to create his own version of the device.[13] The result, Electronic Delay Storage Automatic Calculator (EDSAC) made incremental improvements to the EDVAC design. Completed before Mauchly and Eckert's planned EDVAC device, the EDSAC became the world's first fully electronic, stored-program computer in May 1949.[14] With its completion, EDSAC introduced the first electronically stored computer programs. The computer stored instructions with a central program capable of recalling these stored sequences as needed. This process, called a Wheeler Jump, introduced the modern, software-based subroutine**.**[15] The EDSAC also introduced

---

[11] Attempts to patent the design were further complicated by Mauchly's exposure to the work of computer pioneer John Vincent Atanasoff, who had built a rudimentary electronic computer device, the Atanasoff-Berry Computer. Mauchly had spent time with Atanasoff, reviewed the device, and discussed ideas that later appeared in ENIAC and EDVAC. Nonetheless, the United States Patent Office did issue patent number 3120606 to the inventors for ENIAC in February 1964 – more than a decade after their design had become irrelevant.

[12] Beyer, 127.

[13] The design came to be known as Von Neumann architecture due to him being listed as the sole author on the report. The report was based on the technologies being developed by Mauchly and Eckert, so Von Neumann was not the originator of the ideas that he reported on. Historian Kurt Beyer argues that Hopper and her colleagues at Harvard had also informed the ideas presented within the report.

[14] Martin Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*, History of Computing (Cambridge, Massachusetts: MIT Press, 2003), 82-85.

[15] Paul E. Ceruzzi, *A History of Modern Computing*, 2nd ed., History of Computing (Cambridge, Massachusetts: MIT Press, 2003), 84.

the first assembler.[16] Called "Initial Orders," the program was stored in read only memory and accepted direct symbolic instruction to control the machine.[17] Assemblers like this would later evolve into programming languages.[18] Software had arrived.

Within a year of EDSAC's 1949 introduction in England, there were two electronic computers in the United States.[19] The nascent industry required programming done through direct instruction to machines. Working directly with machine code was cumbersome and required a high level of skill that few possessed. Since there were no computers previously, there were no computer programmers. The individuals working on computers learned as they went. To this end, many early innovations focused on making computer programming more accessible.

In 1951, Grace Hopper, now an employee of the Philadelphia-based UNIVAC corporation, sought to simplify computer programming by developing a process that allowed computers to program themselves.[20] The result was the early computing system Arithmetic Language Version 0 (A-0). A-0 was not a programming language in the modern sense, but a library of pre-written subroutines which A-0 "compiled" into machine readable code.[21] Although not a full programming language, the system included the first compiler.[22] Subsequent versions of the system introduced human-readable "pseudo-code," or source code in which the computer's human operators could

---

[16] An assembler is a program that takes symbolic instruction from users and translates it into very low-level machine code.

[17] David Salomon, *Assemblers and Loaders*, Ellis Horwood Series in Computers and Their Applications (New York, New York: Ellis Horwood, 1992), 7.

[18] Computer instruction also remained through punched card media. In the early 1950s, programmatical solutions to produce sequenced computer instruction on punch cards became common. See Ceruzzi, 85.

[19] Nathan Ensmenger, *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise*, History of Computing (Cambridge, Massachusetts: MIT Press, 2010), 28.

[20] Beyer, 221.

[21] Ibid., 223.

[22] Ibid.

compose programs. UNIVAC provided its customers with the source code for A-2, the second version of the A-0 system. UNIVAC encouraged its clients to develop the software and share their improvements with the corporation.[23] Because of this collaborative approach, A-2 emerged as a popular language that allowed for simple, mnemonic-based, human readable instructions.[24] For example, a program expression to add A + B = C could now be expressed as simply: ADD 00A 00B 00C.[25] Despite its simplicity, the UNIVAC compiler was inefficient and slow: some programs took up to an hour just to translate.[26] These innovations did help simplify the process of programming, but the exponential growth of the industry outstripped the resources available for writing software. This led to the development of high-level programming languages that were faster and required less complex software development.

One of the first of these new languages began development in 1954 under the guidance of IBM Researcher John Backus. The new platform was developed specifically for mathematical applications running on the IBM 704 computer. Backus's language, FORTRAN (FORmula TRANslation), sought to "virtually eliminate programming and debugging."[27] Released in 1957, the FORTRAN compiler allowed for the direct use of expressions.[28] To add A + B = C, FORTRAN required only the equation itself: C = A + B.[29]

[23] The software collaboration at UNIVAC led to the formal corporate sharing group the UNIVAC Scientific Exchange (USE) in 1955. Comprised of UNIVAC 1103a customers including Boeing, Lockheed, and Sperry-Rand, the group developed and shared software such as the USE Compiler.
[24] Beyer, 317-321.
[25] Ensmenger, 86.
[26] Campbell-Kelly, 34.
[27] Ibid.
[28] Ensmenger, 90.
[29] Ibid., 86.

Programming languages such as FORTRAN promised to simplify the training of programmers, but the exponential growth of computers in the 1950s outpaced the industry's ability to train new programmers. Moreover, corporations employing and training programmers frequently found themselves subject to high turn-over due to the high demand for the skill. The Santa Monica, California based System Development Corporation (SDC) was created by the RAND corporation as a software engineering group for the SAGE missile defense system. In 1958, just three years after the company's founding, SDC's attrition rate was 20 percent. Only 50 percent of SDC programmers remained after 4 years. After 7 years, this number dropped to 30 percent. By the time SDC was five years old, over 4,000 programmers trained by the organization had left for other corporations.[30]

Contributing to this problem was a lack of conception of software as a discrete concept. Early computer companies such as the Eckert Mauchly Computing Company, did not give much thought to programming, as it was conceptually inseparable from the computing hardware itself.[31] As a result, they did not consider future staffing needs and generally considered their operators to be in a sub-professional role. The operators in the early period were primarily female. In her monograph *Recoding Gender,* historian Janet Abbate argues that gendered nature of the position was due in part to war time labor shortages and also in part to the limited regard that was given to the positions. World War II drove the need for computing technologies: ENIAC for ballistics research and the British Colossus, for decrypting enemy cyphers. Because men were unavailable due to

[30] Campbell-Kelly, 40-41.

[31] Janet Abbate, *Recoding Gender: Women's Changing Participation in Computing*, History of Computing (Cambridge, Massachusetts: MIT Press, 2012), 26.

the war, many women worked as human "computers." Women who were experienced as "computers" or with mathematics made ideal candidates to operate the new machines designed to perform both functions.[32]

Following the war, corporate interest in computing boomed, but the industry lacked adequate resources to keep up with the commensurate staffing demands. "The present shortage of trained programmers is not likely to be relieved in a short time," warned UNIVAC's Grace Hopper and John Mauchly in 1953, "since new computers are going into use at an ever-increasing rate."[33] With no pre-existing software industry, corporations that required software had to develop the programs themselves. With complex technology and a scarcity of programming resources, this was a prohibitive task.[34] To meet this need, corporate software sharing groups such as IBM's SHARE emerged to facilitate the development and sharing of software.

Corporate Software Sharing Groups

IBM SHARE

While many of the early electronic computers were products of academia frequently funded by the military, many corporations were quick to commercialize the new innovations. IBM was not one of these. Content to sell existing products such as tabulators and punched card calculators, the company was initially dismissive of the new computing devices. However, after IBM competitors such as Remington Rand began to

---

[32] Ibid., 12, 18-20.
[33] Ibid., 90.
[34] In addition to Janet Abbate's later work, historian Nathan Ensmenger also explores the staffing issues and the political tensions involved with the professionalization of the industry.

sell computers, IBM understood their competitive threat.[35] With the outbreak of the Korean War, the United States government asked IBM to create a computer that the military could use for designing aircraft and manufacturing munitions.[36] The result was IBM's first commercial computer was the IBM 701. Released in 1953, the mainframe was designed to compete with Remington Rand's UNIVAC 1103. Because the 701 was inferior to its competitors due to speed limitations, it was succeeded by other models such as the IBM 650 and the IBM 704 in 1954.[37] The latter computer offered innovations that its competitors did not: computer memory, high speed arithmetic functions, and later, the FORTRAN programming language.[38] Its fast processing speed and arithmetical functions made it ideal for mathematic and scientific research, but it quickly gained a corporate user base as well.[39] The success of the 704 helped fuel IBM's dominance of the computer industry.[40] It also informed the creation of one of the industry's first computer user groups, SHARE.[41]

Since the IBM 704 offered superior functionality, many users were interested in upgrading to the new platform. Despite their desire to upgrade, many customers were

---

[35] Remington Rand purchased ENIAC's creators' Eckert-Mauchly Computer Company (EMCC) in 1950 and Engineering Research Associates (ERA) in 1952. Arthur Norberg provides an interesting study of these acquisitions in his text Arthur L. Norberg, *Computers and Commerce: A Study of Technology and Management at Eckert-Mauchly Computer Company, Engineering Research Associates, and Remington Rand*, 1946-1957 (Cambridge, Mass.: MIT Press, 2005).

[36] "IBM 701 Introduction 2," International Business Machines, https://www.ibm.com/ibm/history/exhibits/701/701_intro2.html, (accessed January 27, 2020).

[37] The 701 had lower Input / Output (I/O) speeds – the rate at which the computer can communicate with devices such as printers, network cards, etc.

[38] The 704 featured memory using magnetic cores, a precursor to the RAM memory used today. It also featured the ability to perform floating-point arithmetic, a process similar to scientific notation where arithmetic processing is sped up by replacing very large numbers with representative values and exponents.

[39] Ceruzzi, 64.

[40] Ibid., 68.

[41] Despite its capitalization, SHARE is not an acronym. In fact, the group's byline is "SHARE: It's not an acronym, it's what we do."

faced with the challenge that the IBM 704 was not compatible with its predecessor. Software written for the IBM 701 would not necessarily transfer to the 704.

In 1955, a group of customers, including the RAND corporation and Lockheed, formed an IBM users group named SHARE.[42] The original purpose of the group was to work together to address common problems with the upgrade from the IBM 701 to the 704. In the following years, the group's mission expanded to fostering "the development, free exchange, and public dissemination of research data pertaining to SHARE computers in the best scientific tradition."[43] The group created the SHARE Program library. The collection was managed by the non-profit SHARE Program Library Agency (SPLA), that acted as a centralized distribution point for software written by IBM users. Originally limited to SHARE member installations, software submissions were eventually opened to include contributions from outside programmers.[44] The library contents were advertised via print catalogs sent to SHARE members. Members would select the programs that they wanted, pay a minimal fee for the processing and storage media, and the SPLA would send them the source code for the programs via punched card or tape.[45]

In the early decades of SHARE, the group developed and shared hundreds of software packages.[46] However, SHARE was not alone in fostering the community production and sharing of software that became the norm in the early computing. Other

---

[42] UNIVAC had a similar user group. In 1955, a group of UNIVAC 1103a users at Lockheed, Boeing, and Sperry-Rand formed the UNIVAC Scientific Exchange (USE). The group began a newsletter, USEful Notes, in 1956 and they also began to develop and share USEful software, such as the USE Compiler. This group was essentially the codified form of the collaboration detailed about in the discussion about A-0 and A-2 compilers. For the sake of concision, UNIVAC software sharing is not revisited here.

[43] *SHARE PROGRAM LIBRARY AGENCY User's Guide and Catalog of Programs* (Research Triangle Park, North Carolina: Triangle Universities Computation Center, 1977), i.

[44] Ibid., vi.

[45] Ibid., ii.

[46] Contributions included the SHARE Operating System for the IBM 709, and the PL/1 programming language.

manufacturers, such as the Digital Equipment Corporation (DEC), cultivated similar alliances.

Digital Equipment Computer Users Society (DECUS)

In August 1957, Ken Olsen, and Harland Anderson founded the Digital Equipment Corporation in Maynard Massachusetts. The men had worked together at MIT on one of the first transistorized computers, the TX0. They had also participated in the development of early innovations such as magnetic information storage systems. Although their original intention with DEC was to create computer modules for testing memory systems, by the end of 1959, they had constructed a computer, the Programmed Data Processor-1 (PDP-1).[47] The device became commercially available in 1960.[48] This computer was significantly smaller than the mainframes of the era, and its $120,000 purchase price was considerably more affordable. The PDP-1 was designed for timesharing, a configuration that allowed for multiple users to simultaneously use the system.[49] This was a less costly alternative to mainframe batch computing. The device's display screen and light pen provided an interface that allowed manipulation of on-screen objects and led to the development of some of the first graphical programs, such as Snowflake, software that demonstrated the computer's graphical ability by producing a series of snowflake-like shapes on the computer screen.[50]

---

[47] The devices were named Programmed Data Processor rather than computer because at the time the latter term was associated with large, expensive machines. DEC focused on the device's data processing utility to differentiate the device from its competition.

[48] Ceruzzi, 128.

[49] "PDP-1: Time Sharing," The Computer History Museum, http://www.computerhistory.org/pdp-1/timesharing/ (accessed November 17, 2017).

[50] "Snowflake," The Computer History Museum, http://www.computerhistory.org/pdp-1/0b08375e1559f5d4ebc67e29fa359da1/ (accessed November 17, 2017).

The Digital Equipment Computer Users Society, DECUS, was founded in March 1961. Because the PDP-1 was shipped without software, organizations that purchased the device needed to create their own. The recipients of the first PDP-1, Bolt, Beranek, and Newman (BBN), had a team of researchers compose several software packages for the system, including the device's first assembler program and operating system.[51] One of these scientists, Dr. Edward Fredkin, helped establish DECUS for "the purpose of fostering the interchange of information, ideas, and the advancement of the art of programmed data processing."[52] The group shared their endeavors with other group members via the "programming library facility" established by the user group.[53] Like their IBM counterpart, this library offered software developed and freely shared by PDP users.

The DECUS library acted as a centralized distribution point for software to be shared with requestors at cost. DECUS was managed and subsidized by the Digital Equipment Corporation. The group was focused around two symposia held each year where DEC exhibited new designs and discussed client projects. At these meetings members shared information about their latest programming projects. The sessions featured meetings to discuss new products and applications. This focus helped DEC leverage the meetings to encourage adoption of the technologies displayed by DEC's engineers.[54]

---

[51] Ceruzzi, 128.

[52] Elsa Newman, *DECUS Proceedings 1962: PAPERS AND PRESENTATIONS of The Digital Equipment Computer Users Society* (Maynard, Massachusetts: Digital Equipment Corporation, 1963), v.

[53] Ibid.

[54] *DEC Is Dead, Long Live DEC: The Lasting Legacy of Digital Equipment Corporation* (San Francisco, California: Berrett-Koehler, 2003), 27.

Corporate Motives for Software Sharing

Despite the high-minded mission statements about fostering the interchange of information, ideas, and advancing programming, the reality was that the cooperative approach was designed to address the labor shortage that plagued the early computing industry. Groups like SHARE and DECUS also offered benefits to the manufacturers of their respective computer platforms. First, it helped sell computers. Having more programs for their computing platform meant more utility for the devices; the more programs a computer had, the more it could do. The more a computer platform could do, the more attractive it was to potential consumers. Second, the sharing groups offered the equipment manufacturers the ability to provide additional value to their clients for little extra effort. To create a sharing group, manufacturers simply needed to set up a software library, distribution process, and then solicit participation by contacting their users or planning events such as symposiums. The users themselves wrote and shared the software; the manufacturers simply facilitated it. Finally, the groups provided the corporations a venue to sell new products, such as at the Digital Equipment Corporation's DECUS symposia each year. The DEC organization officially disapproved of using the meetings for sales (the sales and marketing teams were discouraged from attending), but the reality was that their engineers exhibited new products and focused on how the new offerings would further client projects.[55]

Manufacturer-focused user groups were a natural fit in this early period of computing, where computers were marketed as an all-inclusive package. When customers leased a computer from a vendor such as IBM, they received hardware, software, and

---

[55] Ibid.

support. The vendor worked with them to achieve their computing goals. As a result, the programs that ran on the computer were conceived as a part of the whole solution, rather than as a separate commodity.

This holistic, vendor-centric approach fit the state of the computing field at that time. There was no tenable way to create a prepackaged software industry in this period. First, just like corporate computer users, software manufacturers would have faced challenges in retaining programming staff. Second, in the early computing period when groups like SHARE were formed, there were no apparent mechanisms for software distribution outside of the manufacturer's client roster. Lastly, in the first several decades of electronic computing, there were not enough computers to sustain a separate software industry.[56] (See Table 1.)

Table 1: Electronic Computers in U.S. by Year

Year:	Electronic Computers in U.S.:
1950	2
1955	240
1960	5400
1965	25,000
1970	75,000

Source: Adapted from Ensmenger, 28.

With no pre-packaged software industry, corporations in need of software had to engage consultants to develop programs or hire and manage programmers themselves. Either option was costly. According to historian Martin Campbell-Kelly, the cost for establishing a basic programming department at this time was equal to the first year's

[56] Ensmenger., 28.

rental of the IBM 704 – over $150,000.[57] Co-operative efforts such as SHARE and DECUS lowered software costs by avoiding redundant efforts and distributing the workload for creating commonly needed software. Sharing was a self-serving solution to the programming shortage: co-operation simply cost less than competition.

SPACEWAR!

Not all programs composed by the user groups filled a deliberate business need. One of the most popular programs distributed by DECUS was the world's first video game: SPACEWAR! As described by DECUS member J. M. Graetz:

> The game starts with each player in control of a spaceship (shown on PDP scope) equipped with propulsion rockets, rotation gyros, and space torpedos [SIC]. The use of switches to control apparent motion of displayed objects amply demonstrates the real-time capabilities of the PDP-1.[58]

SPACEWAR! was conceived at MIT's Hingham Institute in December 1961. The program simulated the effects of piloting ships through space. The battle was centered around a "heavy star" whose gravity pulled the craft toward it. Players accounted for the forces exerted by this body as they maneuvered around other celestial objects and targeted "space torpedoes" at their nemesis. If either ship, "the needle" or "the wedge," seemed unable to accelerate rapidly enough to avoid destruction, the player could also have the ship jump to "hyperspace," which would instantly transport the craft to another location on the screen.[59]

Harvard programmer Stephen Russell composed the game on a PDP-1 in early 1962. The program could be played on a single console, or challengers could face off by

---

[57] Campbell-Kelly, 30-31.

[58] J. M. Graetz, *DECUS Proceedings 1962: SPACEWAR! REAL-TIME CAPABILITY OF THE PDP-1* (Maynard, Massachusetts: Digital Equipment Corporation, 1963). 38.

[59] Ibid. 37.

controlling ships from separate consoles. The software use of the PDP-1's optional cathode ray tube (CRT) display and its real-time game play showcased the processing capability of the PDP-1 and the game was commonly the first program installed to "test" new PDP installations.[60] The video game helped fuel interest in computing over the next decade. As researcher Alan Kay explained, "Spacewar! blossoms spontaneously wherever there is a graphics display connected to a computer."[61]

Counter-culture journalist Stewart Brand examined the game and the West Coast computer counterculture that was drawn to it in his 1972 *Rolling Stone* article, "SpaceWar!" In the piece, Brand explores the impetus for the program with its author, Steve Russell.

> We had this brand-new PDP-1…. It was the first minicomputer, ridiculously inexpensive for its time. And it was just sitting there. It had a console typewriter that worked right, which was rare, and a paper tape reader and a cathode ray tube display…. We decided that probably you could make a two-Dimensional maneuvering sort of thing, and decided that naturally the obvious thing to do was spaceships.[62]

Brand's examination of the game also highlighted other computing technologies that were being developed by programmers at Xerox's Palo Alto Research Center (PARC). According to the author, Spacewar! "was a flawless crystal ball of things to come in computer science and computer use." His proclamation was based on the distributed nature of the computing as well as the interactivity of the game which "bonded human and machine." He noted that it also "encouraged new programming by

---

[60] "PDP-1: Spacewar!," The Computer History Museum, http://www.computerhistory.org/pdp-1/spacewar!/ (accessed November 17, 2017).

[61] Stewart Brand, "S P A C E W A R: Fanatic Life and Symbolic Death Among the Computer Bums," *Rolling Stone,* December 7, 1972, http://www.wheels.org/spacewar/stone/rolling_stone.html (accessed November 17, 2017).

[62] Ibid.

the user."[63] He was prescient: SPACEWAR! helped prompt the development of another critical piece of software – the UNIX operating system.

Software Sharing in the UNIX Community

The UNIX operating system has been one of the most influential and emulated technologies in computer history. UNIX's modular design made it easy for the software to be widely shared, and the technology itself later informed the creation of Free Software's seminal GNU Project.

In the late 1960s a small team of engineers at AT&T Bell Labs was engaged in a cooperative project with General Electric (GE) and MIT to develop a modern time-sharing operating system named Multics (Multiplexed Information and Computing Service). Based on MIT's earlier operating system, Compatible Time-Sharing System (CTSS), Multics allowed multiple users to operate the same computer concurrently. Where CTSS was only available to MIT, the Multics project was to be commercially available on GE's mainframe computer, the GE-645.[64] MIT provided direction and funding via a Pentagon Advanced Project Research Agency (ARPA) grant. AT&T Bell Labs rented a GE mainframe and assigned a team of programmers including Ken Thompson, Robert Morris, and Brian Kernighan to compose the operating system.[65]

By 1969, the project was behind schedule because Multics was to be written in PL/1, a high-level language by IBM and SHARE, but a PL/1 compiler had yet to be written for the GE-645. To work around this deficiency, Bell programmers wrote a

---

[63] Ibid.

[64] Peter H. Salus, *A Quarter Century of Unix* (Reading, Massachusetts: Addison-Wesley Pub. Co., 1994), 8.

[65] "Multics History," Multicians, August 29, 2017, http://multicians.org/history.html (accessed March 20, 2018).

simplified, temporary compiler, EPL/1 (Early PL/1).[66] Other programmers chose to start writing applications in FORTRAN, even though the operating system also lacked a FORTRAN compiler.[67] With little progress to show for the effort, it became clear that the project delays were draining the Bell Lab's computing center budget. As researcher Dr. Douglas McIlroy explained, "They had this million dollars' worth of equipment up in the attic, that was sitting there being played with by three folks," and there was no clear benefit to the effort in the near term.[68] In April 1969, Bell made the decision to cut the project and officially all work ceased.

Despite the cancellation, the programmers working on the project were not quite ready to quit. As researcher Dennis Richie put it:

> We didn't want to lose the pleasant niche we occupied, because no similar ones were available. What we wanted to preserve was not just a good environment in which to do programming, but a system around which a fellowship could form.[69]

It was in this pleasant niche that fellow researcher Ken Thompson had written a video game similar to Spacewar! Thompson's game, Space Travel, allowed the user to explore a two-dimensional scale model of the solar system. The game was on the GE-645 intended for Multics. Bell's withdrawal from the Multics project meant the loss of the powerful GE mainframe that ran the game.

---

[66] Doug McIlroy, interviewed by Michael Mahoney, Murray Hill, New Jersey, August 18, 1989. https://www.princeton.edu/~hos/mike/transcripts/mcilroy.htm (accessed March 20, 2018).

[67] Brian Kernighan, interviewed by Michael Mahoney, Murray Hill, New Jersey, April 17, 1998. https://www.princeton.edu/~hos/mike/transcripts/kernighan.htm (accessed March 20, 2018).

[68] Doug McIlroy, interviewed by Michael Mahoney, Murray Hill, New Jersey, August 18, 1989. https://www.princeton.edu/~hos/mike/transcripts/mcilroy.htm (accessed March 20, 2018).

[69] Martin Campbell-Kelly, *Computer: A History of the Information Machine*, third ed., The Sloan Technology Series (Boulder, Colorado: Westview Press, a member of the Perseus Books Group, 2014), 213.

Thompson was also heavily engaged with the design of a hierarchal filesystem for the Multics Operating System.[70] By the time of the project's cancellation, the programmer had developed a high-level simulation of the planned software.[71] Thompson wanted to complete his research and sought a way to continue beyond this model. At the time, Bell Labs had an outdated PDP-7 that the programming team used for graphical work. It was much smaller and less powerful than the GE-645, but its graphical abilities made it a suitable candidate to run Thompson's video game. The researcher adapted "Space Travel" for the PDP-7, and in the summer of 1969, he continued his research by composing an operating system to run on the computer. Thompson wrote the software over a month, pacing the project to allocate "a week each to the operating system, the shell, the editor, and the assembler."[72]

Soon other Bell team members also began to contribute to the fledgling operating system by writing applications to run on it. By 1970, the new operating system had a name, UNiplexed Information and Computing Service or UNICS. The name, later shortened to UNIX, was a play on the software's origin in MULTICS.[73]

Despite the progress on the operating system, the effort lacked official recognition by AT&T or Bell Labs. Since the UNIX programmers were using institutional resources to continue their research, they were faced with a challenge to demonstrate the utility of the unsanctioned project. Team members Peter Neumann and Lee McMahon worked with Bell management to justify the continued use of the PDP-7 for the project, and they

[70] A file system is software that controls how data is stored, indexed, and retrieved on a storage device such as a hard disk. A hierarchal filesystem is one with a tree of file system objects such as files and folders.

[71] Ken Thompson, interviewed by Michael Mahoney, Murray Hill, New Jersey, September 6, 1989. https://www.princeton.edu/~hos/mike/transcripts/thompson.htm (accessed March 20, 2018).

[72] Ibid.

[73] Salus, *A Quarter Century of Unix*, 9.

even tried to procure a new computer for the effort. Programmer Lorinda Cherry used UNIX to develop software for typing, spellchecking, and formatting text. Collectively, the text processing suite served as a precursor to desktop publishing.[74] In 1970, the programmers demonstrated the tools to Bell's patent department. At the time, the patent group was poised to purchase a commercial typesetting product, AstroText, for formatting patent applications. The programmers used the UNIX text processing tools to develop a system that met the patent group's needs. The resulting solution was technically superior to the AstroText offering, and the patent department used the funds that they had allocated for the AstroText purchase to instead acquire a PDP-11/45 for the UNIX project.[75] Internal interest in the software grew, and by June of 1972, AT&T had ten internal installations of UNIX in production.[76] Soon the operating system expanded beyond the confines of the Murray Hill, New Jersey research facility.

On October 15, 1973, Ken Thompson and fellow researcher Dr. Dennis Ritchie travelled up the Taconic parkway to IBM's Thomas J. Watson Research Center in Yorktown Heights, New York, where they attended the Association for Computing Machinery (ACM) symposium on Operating Systems Principles. There they presented their work on UNIX. They demonstrated the operating system, recently re-written in Dr. Ritchie's new C programming language. The work they presented was published in the July 1974 *Communications of the ACM.* The reaction to the publication was overwhelming, and Bell was soon flooded with requests for the operating system.[77] This

---

[74] Ken Thompson, interviewed by Michael Mahoney, Murray Hill, New Jersey, September 6, 1989. https://www.princeton.edu/~hos/mike/transcripts/thompson.htm (accessed March 20, 2018).

[75] Ibid.

[76] Ken Thompson and Dennis Ritchie, *UNIX programmer's Manual,* 2nd Edition (Murray Hill, New Jersey: Bell Labs, 1972), ii.

[77] Salus, *A Quarter Century of Unix,* 54-55.

presented a novel problem for Bell as they were legally prohibited from selling the software.

On January 14, 1949, the U.S. Department of Justice filed an anti-trust suit against the AT&T Bell System. The suit alleged that the Bell System monopolized the manufacture, sale, and distribution of telecommunication equipment. Bell Telephone controlled 98% of the United States' long-distance network and 85% of the local telephony infrastructure across the nation. They purchased their equipment through their manufacturing subsidiary, Western Electric, which held a 90% market share in telecommunication equipment. Western Electric, in turn manufactured equipment based on the research performed at Bell Labs.[78] The suit ended in January 1956 with a consent decree that prescribed two requirements: First, AT&T was restricted from any business outside of telecommunication. Second, AT&T had to license all previously issued patents to any applicant, royalty-free.[79]

UNIX created an issue for AT&T. The first condition of the 1956 consent decree forbade AT&T from pursuing non-telephony lines of business and thus prevented the organization from selling UNIX as a product. However, the second condition required the licensing of everything they developed. Although UNIX had emerged after the consent decree, AT&T's legal department interpreted the decree as requiring them to license UNIX as well.

[78] Martin Watzinger et al., "How Antitrust Enforcement Can Spur Innovation: Bell Labs and the 1956 Consent Decree" (paper presented at the Center for Economic Studies (CES), the IFO Institute Ludwig Maximilian University of Munich, Munich, Germany, February 27, 2017), http://www.cesifo-group.de/ifoHome.html (accessed March 20, 2018). 8.

[79] Ibid.

AT&T's solution was to make the software available to outside institutions through their Western Union subsidiary. The license contract provided the software royalty-free. For universities, there was a nominal fee of $150 and for non-university institutions, the cost was $20,000.[80] For this amount, Western Union provided the binary object code of the UNIX operating system, the source code, and manuals for UNIX, C, UNIX assembler, and a text formatting program, NROFF.[81] The license stipulated that the software was unsupported, carried no warranty, and made clear there would be no future updates for either the software or documentation.[82] For academic licenses, the text specified that the software could not be used for any commercial effort; instead, it had to be employed solely for educational purposes at the institution specified. The license was non-transferrable, but copying the software was allowed when necessary for the institutional use of the program. Nothing in the licenses prohibited the modification of the source code, the addition of new applications, or the porting of the software to new platforms.

External use of AT&T's new operating system grew rapidly. Within a year of the publication in the *Communications of the ACM,* there were over three dozen institutions using the operating system. Within two years, this number had grown to almost one hundred and forty. Most of these early adopters were universities that already owned inexpensive PDP computers and found the cost of the software to be nominal. The operating system was used worldwide, with UNIX servers in the U.S., Canada, Great

---

[80] Steve Holmgren, *The Network Unix System* (Champaign, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1975). 8.

[81] *SOFTWARE AGREEMENT between WESTERN ELECTRIC COMPANY, INCORPORATED and [REDACTED]* (New York, New York: WESTERN ELECTRIC COMPANY, 1973). 5.

[82] Ibid., 3.

Britain, Australia, Israel, Austria, Belgium, Germany, and Venezuela.[83] ARPA, which had been funding UNIX's technological predecessor, MULTICS, also took an interest in Bell Labs' handiwork. They funded the University of Illinois at Urbana-Champaign's efforts to incorporate UNIX into the fledgling ARPAnet, precursor to the Internet.[84]

As counterintuitive as it may seem, AT&T's restrictions encouraged the communal development and support of the software. For example, the UNIX source code was only available for DEC PDP computers, so users who wished to use the operating system on other architectures had to port the software to the new platform themselves.[85] This type of adaption began as early as 1976, with computer scientist Richard Miller and his colleagues at the University of Wollongong in Australia. The university had an Interdata 7/32 computer, but the software running on the device had been unstable. They wished to run UNIX, but the operating system ran on PDP hardware, not the Interdata. At this time, most operating systems were composed in platform-specific assembly language. However, since UNIX was written in C, a high-level programming language that was not bound to any one hardware platform and because they had the source code, they were able to modify the operating system and port it to the new platform themselves. They began by creating a C compiler for the Interdata 7/32. Once this was done, they simply needed to recompile the UNIX source code on the Interdata.[86] This first port of

---

[83] Salus, *A Quarter Century of Unix,* 66-68.

[84] Steve Holmgren, "RFC 681: NETWORK UNIX," Internet Engineering Task Force, May 14, 1975, https://tools.ietf.org/html/rfc681 (accessed March 20, 2018).

[85] Port, in this context, means to alter and/or re-write the software to facilitate the changes needed to allow the software to run on a dissimilar computer.

[86] Salus, *A Quarter Century of Unix,* 128-129.

UNIX allowed the operating system to be run on the new platform despite the systems' vastly different hardware. [87]

The UNIX license limitations required communal support of the software. Since UNIX was not maintained by AT&T, the users of the software needed to sustain and build the platform. This user-based support can be seen in the evolution of UNIX editors. In 1975, users at Queen Mary College in London were frustrated with 'ed,' a program that allowed UNIX users to edit files. They developed an alternative application, 'em' that allowed them to edit files one line at a time. They shared this with other UNIX users. University of California Berkeley graduate student Bill Joy used the program but was concerned with the load that it placed on the computer's central processing unit. So, he re-wrote it and created a full-screen editor, which he called 'vi.'[88] This program was one of many made by Joy and his cohort at Berkeley's Computer Systems Research Group. Their incremental improvements eventually became shared as the Berkeley Software Distribution (BSD), which, starting in 1977, was shared for the cost of the media.[89]

The design of UNIX encouraged portability and this type of incremental addition. The operating system was developed on an antiquated PDP-7, a system with limited memory and storage space. The need to accommodate these constrained resources informed the design of the operating system. The result was forced elegance; UNIX possessed efficiency by necessity and featured a design focused on modularity. The central core of the operating system, the kernel, was separated from the user interface, the shell. The shell program was simple and concise; its source code fit on just seven pages

[87] The Interdata computers were the first 32-bit computers, and they also stored numerical values in memory in the opposite order from what the 16-bit PDP computers did.

[88] Salus, *A Quarter Century of Unix* ,140-142.

[89] Ibid., 158.

of printer paper.[90] UNIX also introduced a novel software concept, the pipe. The pipe permitted a program running on UNIX to redirect its output as input into another application. This allowed for complex procedures to be constructed from simple, individual commands. This approach conserved system resources by using small programs that could be arranged sequentially to perform complex tasks.

With pipes, simple UNIX commands become increasingly powerful as they are combined. This modular approach also made it possible for the UNIX users to contribute software to others in the community. Instead of having to write and support complex monolithic applications, developers could quickly write and share new, small programs and rapidly make incremental contributions to the functionality of the over-all system.

UNIX was particularly popular with academic computing departments due to its low cost and source code availability. The software was frequently used to teach students how operating systems worked, and this exposure resulted in many students and faculty developing programs as part of the educational process. The modularity of the software encouraged contributions to the platform, with developers creating and sharing programs that performed the functions they required. One of the challenges this communal development created was distribution. How would the programmers circulate the software they created or updated? Early, this occurred through word of mouth requests for software that was then shared via magnetic tape. In 1978, Bell Laboratory researcher Mike Lesk wrote UUCP (Unix to Unix Copy). Using this software, the UNIX user

[90] Ibid., 46.

community soon created their own dial-up global communication network, UUCPNET.[91] This allowed UNIX users throughout the world to freely exchange email and files. Most importantly, this enabled the easy sharing of programs and provided a means of communication for coordinating distributed software development.

BASIC and Software Sharing in Early Home Computing

While UUCPNET's global sharing allowed for the continued evolution of UNIX, Dartmouth Professor John Kemeny saw the sharing of technology as a key step in the evolution of society. Kemeny believed that the relationship between computers and mankind was akin to two symbiotic species.[92] In order for both species to advance, they needed to work together. He believed that this symbiotic evolution had the potential to solve many of the problems facing society. He foresaw technical solutions such as the creation of a national data bank that would reduce government bureaucracy by simplifying paperwork and eliminating paper-pushing jobs. This centralized information system could be used to decrease unemployment by providing information about open positions. It also could be used to detect issues, such as welfare applicants abusing the system.[93] He imagined similar technological advances that would refine society by expanding libraries and reshaping methods of commerce. While optimistic about potential applications for computing, Kemeny also perceived the challenges with his vision. First, computers were strange and alien to most people, and many members of society were afraid of the mysterious devices. Furthermore, human interactions with computers required mastering difficult and arcane programming languages. To address

---

[91] Doug McIlroy, *A Research UNIX Reader* (Murray Hill, New Jersey: Bell Labs, 1986), 3.
[92] John G. Kemeny, *Man and the Computer* (New York, New York: Scribner, 1972), 19.
[93] Ibid., 127-129.

these issues, it was necessary to redefine the relationship and provide a simplified way for humans to communicate with computers.

In 1963 John Kemeny and his colleagues at Dartmouth College determined that the world was changing; the time of co-evolution had come. Computers were “beginning to have an increasing effect on the lives of all.… They play key roles in business, industry, government, and all forms of research.”[94] To prepare students adequately for a computerized future, the college developed a curriculum that required students to use computers to do routine tasks, and the school also encouraged faculty to pursue projects that would benefit from computing. The challenge in achieving this was considerable as the college had only gotten its first computer four years earlier, and few of Dartmouth’s faculty had any familiarity the machine.[95]

To facilitate this computer curriculum, Kemeny had to develop a way to make computers accessible to all, regardless of previous experience or ability. Kemeny reasoned that instead of forcing people to learn an arcane computer language, it would be simpler to “teach a computer a language easily understandable to human beings.”[96] Kemeny and fellow faculty member Thomas Kurtz achieved this with the creation of two technical platforms: The Dartmouth Time-Sharing System (DTSS) and, more importantly, the Beginner’s All-purpose Symbolic Instruction Code (BASIC).

Dartmouth Time-Sharing System (DTSS)

---

[94] John Kemeny and Thomas Kurtz, *The Dartmouth Time-Sharing Computing System - Final Report* (Hanover, New Hampshire: Dartmouth College, 1967). 1.

[95] Ibid., 8.

[96] Ibid., 54-55.

In the 1950s and 1960s computers were large, monolithic machines that processed programs in a linear manner. This early batch processing approach was a sacerdotal affair. Programmers would use card-punch machines to write programs with each line of the program represented as a series of punches on a card. The cards were stacked in order and submitted to the computer's operators. These system operators were the only ones allowed to directly interface with the machine. They would collect the coded cards and submit them to the machine through a punched card reader. The computer would work on each job in the order it occurred in the batch. So, the first program would run for as long as it took to process and then the computer would move on to processing the second job, and so on. Once the entire batch was completed, the output for each of the jobs was printed and returned to the submitter. Batches could take hours or even days to complete and if a program contained an error, the program had to be updated to fix the problem, the request resubmitted, and processing had to be repeated. During the 1960s the punched cards were gradually replaced by magnetic tape reels, but the process remained the same: the programmer would bring their tape reel to the system operator who acted as sole intercessor with the computer. Once accepted, the program would be queued for linear batch processing.[97]

The inefficiencies of batch processing were apparent by the early 1960s. Because it made little sense to have a million-dollar computer dedicated to a single user at a time, many institutions sought to develop time-sharing operating systems that allowed multiple users to use a computer simultaneously. For Kemeny and Kurtz's project to succeed, they needed such a system. Since having the college's 250 freshmen using a single computer

[97] Ibid., 6.

one at a time was clearly impractical and there were no commercially available time-sharing systems, the professors wrote their own – the Dartmouth Time-Sharing System.

The Dartmouth Time-Sharing System was an early example of distributed computing.[98] Using a General Electric (GE) Datanet-30 mainframe, the computer team wrote an "executive program," enabling the computer to act as a front-end interface for the users. This computer handled all the user logons but did not perform any computations.[99] Instead, a larger, more powerful GE-235 mainframe was dedicated to act as the computational back end. The Datanet-30 received client requests and managed the processing that occurred on the subordinate GE-235.[100]

The Dartmouth Time-Sharing System's operating system was focused around a limited set of system commands, all of which were based on simple English words. To start a session on DTSS, a user would type HELLO and then enter their user number. They would type NEW to start a new program or OLD to retrieve a program from storage. They could also RENAME the current program or type LIST to see its contents. To run it, they typed RUN, and to stop it, they typed STOP. When they were done working on the program, they could save it by typing SAVE. They could then either work on a different program, or simply type BYE to end the session.[101]

The Dartmouth Time-Sharing System alone was insufficient to achieve the accessibility sought by Kemeny and Kurtz. To lower the technical barriers for

---

[98] Modern distributed computing where user connections are either load balanced between devices or where data analysis is delegated on computational clusters, employ a similar architecture as the DTSS configuration. Having the workload delegated in this way also allows for easy expansion to meet demand. In fact, within the first five years of operation, Dartmouth expanded the DTSS cluster to four machines.

[99] The users of DTSS connected via Teletypes, an electromechanical typewriter that could transmit to and receive information from the Datanet-30.

[100] Kemeny and Kurtz, 9.

[101] Ibid. 38.

Dartmouth's students and faculty, they also needed a programming platform that was as easily learned as DTSS and yet still powerful. Their solution was the Beginner's All-purpose Symbolic Instruction Code (BASIC).

Beginner's All-purpose Symbolic Instruction Code (BASIC)

John Kemeny and Thomas Kurtz developed the Beginner's All-purpose Symbolic Instruction Code (BASIC) language and wrote a compiler that would take the user's commands and translate it to machine code. Like DTSS, the language had very simple, English word commands. Vocabulary for the new programming language was limited, with only fifteen possible instructions. Despite this simplicity, the language could perform complex functions, handle exponents, logarithms, square roots, sine, cosine, tangent and arctangent.[102]

In addition to the programming language, the Dartmouth team also composed a library of over 500 programs for their users to share. These included standard mathematical routines, course specific software, and number of games ranging from three-dimensional tic-tac-toe to a simulated football game between Dartmouth and their rival, Princeton University.[103]

The computing program at Dartmouth was successful. By 1967, Dartmouth had introduced over two thousand students to the computing system. Student participation came from many fields: mathematics, engineering, psychology, physics, social studies, and even business administration. In his final report on the project, John Kemeny cited the example of a program that he personally wrote. Having seen references to the "largest

[102] Thomas Kurtz, "BASIC Commands: The Original-Original Version," Dartmouth College, October 26, 2005, http://www.dartmouth.edu/basicfifty/commands.html (accessed March 22, 2018).
[103] Kemeny and Kurtz, 12.

known twin prime" numbers, Kemeny was curious if the Dartmouth system was powerful enough to calculate a larger pair. Within one hour, he wrote the BASIC program needed to perform the calculation. Then, with 20 other people using the system, the program took just 11 minutes to run and report back two sets of numbers which were then the largest known twin primes.[104]

Dartmouth's computing platform attracted interest from many other institutions. In response, Dartmouth made the system freely available to educational facilities via modem connections. Institutions that used the system in these early years included Harvard, Johns Hopkins, Lehigh University, Mount Holyoke, and Princeton.[105] In addition, a few nearby high schools, such as the Phillips Exeter Academy, enjoyed regular access to the system.

This unrestricted access to the computing resources, even for external organizations fit with Kemeny and Kurtz's democratic vision of technology. Kurtz had convinced Dartmouth's trustees to share the computing resources by comparing open access to the system to an "open stack library."[106] Their rationale was that the institution would not charge for access to library resources, so why should they charge for access to computing resources? And just as with the College's computing resources, the University also chose not to seek financial gain from BASIC. To make the technology universally available, the BASIC program was not copyrighted, trademarked, or sold. Instead, Dartmouth released the software into the public domain.[107]

---

[104] Ibid. 5, 22

[105] Ibid., 24.

[106] Mark Jones Lorenzo, *Endless Loop: The History of the Basic Programming Language (Beginner's All-Purpose Symbolic Instruction Code)* (Philadelphia, Pennsylvania: SE Books, 2017), 57.

[107] Ibid.

Dartmouth was successful in disseminating the technology. By the 1970s corporations like IBM and Apple, created their own versions of the programming language and the use of the software spread. The free sharing of programs written in BASIC flourished as well. For example, when the Apple II was released in 1977, Apple provided many free BASIC programs in its reference manual. These included video games such as Breakout, Pong, Mastermind, and Dragon Maze.[108] These programs, called "type-ins," were typed into the computer and then run by the home user.

Sharing of BASIC program code was common. In 1971, Digital Equipment Corporation employee David H. Ahl assembled a collection of what he deemed to be the most popular of the shared BASIC computer games. Published as "101 BASIC Computer Games," the text shared a myriad of public domain programs including checkers, tic-tac-toe, Monopoly, poker, and, of course, SPACEWAR!.[109] In late 1974, Ahl left DEC to begin his own magazine, *Creative Computing*. For $8 per year, magazine subscribers received a bi-monthly periodical offering "Classroom Activities, Games, Puzzles, Reviews, Ideas, Social Commentary."[110] Just like the book he assembled at DEC, each issue shared many public domain BASIC type-in programs.

In the decade that followed, many similar magazines emerged. Periodicals like *Compute!, Analog Computing, Enter*, and *High-Res Magazine* offered similar content to *Creative Computing* and contributed to the culture of shared software that surrounded BASIC. In this period before ubiquitous Internet, print media offered an efficient means

---

[108] *Apple 2 Reference Manual* (Cupertino, California: Apple Computer, 1978). 57-67.

[109] David Ahl, *101 BASIC Computer Games* (Maynard, Massachusetts: Digital Equipment Corporation, 1975). 3-5.

[110] David Ahl, "Masthead," *Creative Computing*, 1974. 1-3.

of distribution for the programs that were written and shared. In addition to these formalized publications, BASIC type-ins were also exchanged by users as one would share a recipe. If you liked a game, you could give a printout of the source code to a friend and they could type the program in and run it themselves. If they wanted to change the game, they could made alterations to the recipe – the source code. and they could share the revised version as well.

Summary: The Emergence of Software

General purpose electronic computing emerged in academia in the mid-twentieth century. From the Harvard Mark 1, to ENIAC and EDVAC, each early calculating machine introduced technological improvements that incrementally advanced the field toward fully electronic programmable computing. Although these early giants borrowed ideas from older mechanical devices such as the Jacquard loom’s punched cards, they rapidly made advances in storage and processing, which led to the emergence of software, stored programs on a computer, with Cambridge University’s EDSAC.

When corporations like IBM began to produce and sell mainframe hardware, software was viewed as a necessary component for using these machines, rather than a discrete intellectual property. Faced with a shortage of both programmers and software in the nascent industry, user groups such as IBM’s SHARE formed to develop and share programs. This formal sharing of mainframe software centered around corporate hubs that handled the archiving and circulation of the programs. For the companies involved with sharing software, the motives were self-serving: a cooperative approach reduced their computing costs.

In the late 1960s AT&T scientists developed the UNIX operating system for minicomputers like the DEC's PDP-7. Legal constraints led to AT&T distributing the software royalty-free but completely unsupported. These license limitations required the operating system users to maintain it themselves. This resulted in users adopting a schema of iterative improvements through peer-based development, review, and sharing. Software was distributed initially via physical medium, such as tape reels, but the UNIX community soon developed networking software, UUCP, which allowed them to collaborate and distribute software electronically. For the programmers who created and shared UNIX software, the motive for cooperation was a necessity: AT&T eschewed any responsibility for the maintenance or upgrading of programs.

At Dartmouth College, University professors developed the BASIC programming language as a way to eliminate cultural apprehension toward computers and bring computing to a wider audience. The creator of the software, John Kemeny, saw the language as a step toward realizing an idealized vision of societal change through the symbiosis of man and machine. BASIC was released into the public domain to advance this objective through the dissemination of the software. The programming language was subsequently adopted in the emerging microcomputer market where it became the default programming language for many home computers. Individuals composed, and distributed type-in programs written in BASIC. They were also circulated via printed books and magazines. BASIC was shared for many reasons: The programming language was created and distributed to advance an idealistic world view while type-in BASIC programs were shared like recipes that others could assemble and use.

In the early part of the electronic computing era, software was conflated with the hardware on which it run. At this time distributed development and sharing of software existed on every type of computer, from mainframe to microcomputer. Dissemination methods evolved from individuals to publications, and later to computer networks. The intentions behind sharing were vast and varied, from corporate necessity to technological idealism. The open sharing culture in this early period was informed by the inchoate nature of the nascent industry. The scarcity of trained programmers, software, and distribution methods necessitated many of these practices. However, as technology progressed, and formal structures began to coalesce, the industry evolved. Soon, software was perceived as separate from hardware, and as an intellectual property with intrinsic value. As a result, the open sharing culture was about to change.

CHAPTER 2: FROM OPEN TO CLOSED

In the early era of electronic computing, the practice of sharing software was common. Programs were viewed as an accessory to computer hardware rather than a distinct product. As the burgeoning industry advanced, the number of computers quickly surpassed the supply of programmers. This contributed to the emergence of a culture that encouraged the sharing of software. As the industry expanded, corporations reinterpreted the role of software and in the process, challenged the permissibility of sharing programs. This chapter discusses how the emerging software industry changed the open sharing practices of the computing culture through legal changes, source code restrictions, and secrecy.

This software revolution began with IBM and changes in how they supported and distributed programs. Their modifications redefined software as separate from hardware and in the process fostered a new and distinct software industry. In this nascent market, software companies such as Microsoft, resisted software sharing by suggesting that the practice was theft and that free, hobbyist-produced software was inferior. The budding software industry was bolstered by legal changes that allowed copyright protection for software. Although alterations to the copyright law afforded consumers the right to modify the source code of the programs they purchased, corporations circumvented these consumer rights through license agreements.

Rapid technological advancements further transformed the industry, in particular, within home computing. Software innovations altered the way home consumers interacted with their computers. With the advent of graphical user interfaces, such as

Windows, computer users no longer had to be computer programmers. Instead of writing their own software or typing-in programs written by others, users were now consumers, and their computing experience was focused on purchasing and using programs. These changes were not limited to home computing. By the start of the 1980s, even the structured sharing communities that existed in corporate computing, such as IBM SHARE, were affected as corporations eliminated access to previously shared programs and source code.

The Birth of an Industry

In 1961 IBM faced a major obstacle with software. The scarcity of programmers in the emerging industry was a challenge for all computerized institutions at the time, but IBM bore a greater burden than most. IBM's competitors, such as National Cash Register, served specific niche markets that had limited support requirements. In contrast, IBM served a broad range of markets and, as a result, had seven separate hardware platforms with countless applications on each that needed to be maintained.[1] In addition, because the cost of hardware, software, support, and education were included in the computer lease price, IBM's income was fixed, and yet they had an ever-growing software library to support.[2]

IBM's issues were not limited to this internal support burden. Because each of IBM's hardware platforms were unique, it was difficult for IBM clients to move between products. A company using software written for one IBM model could require a more

[1] In the case of NCR, their niche was banking and retail.

[2] As part of IBM's 1956 consent decree with the U.S. government, IBM agreed to sell as well as lease its computers. For the sake of simplicity, this discussion will refer to leases even though it was possible for an IBM customer to purchase or lease an IBM computer from a third party.

powerful computer, but to accomplish this transition, the client would need to re-write all of its software for the new platform. Faced with the prospect of recomposing their programs anyway, existing clients might consider moving to one of IBM's competitors.[3]

These concerns, combined with an aging hardware catalog that included older vacuum-tube technology, led the IBM organization to create a committee of engineers to evaluate technical strategy.[4] The SPREAD (Systems Programming, Research, Engineering and Development) group released its 88-page report to IBM management on December 28, 1961. Their report focused on the "explosive growth of software" as a key concern and laid out an ambitious plan that called for the planned obsolescence of IBM's current computers, replacing them with a new family of devices.[5] The new computing suite would have six modern processor platforms that would share software and peripherals such as disk and tape drives, printers, and terminals.[6]

This proposal aimed to eliminate the software issues facing IBM. The company would have six hardware platforms, but there would be a single software portfolio to maintain. Clients would now be able to seamlessly upgrade between IBM's hardware offerings. Despite the promise offered by the new platform, it also introduced new questions. The first was how would customers react to their current computers becoming obsolete? And then there was the question of the project itself: could IBM accomplish this revolutionary proposal?

---

[3] Martin Campbell-Kelly, *Computer*, 43.

[4] Thomas J. Watson, *Father, Son and Co.: My Life at IBM and Beyo*nd (New York, New York: Bantam Books, 2000, 1990), 348.

[5] Martin Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog*, 96.

[6]Thomas J. Watson, 347-349.

IBM's System/360

IBM's System/360 project was one of the largest R&D projects undertaken by a civilian entity.[7] *Fortune* magazine called the effort "IBM's $5 Billion gamble" and "the most crucial and portentous — as well as perhaps the riskiest — business judgment of recent times."[8] IBM hired more than 60,000 new employees and opened five new manufacturing plants to gear up for production. The factories and equipment alone cost over four and a half billion dollars.[9] By 1963, IBM senior management christened the project System/360, referencing "all points of the compass."[10] Yet they still struggled with how to introduce the project without jeopardizing their customer base. IBM salesmen were in a particularly untenable position; IBM's competitors exhibited computers that surpassed IBM's current offerings, while IBM's employees were forbidden to reveal their own incredible new innovations.

The situation came to a head in December 1963 when the Honeywell Corporation released its model 200 computer. Designed to be a direct challenge to IBM, Honeywell's machine used modern semiconductor technology that afforded it greater performance than its IBM counterpart. Included with the model 200 computer was the "liberator" program, which ran software written for the popular IBM 1401, but provided demonstrably better performance leveraged by the Honeywell 200's superior processing power. The product created a nightmare scenario for IBM: existing clients could

---

[7] Campbell-Kelly, *Computer,* 127.
[8] T.A. Wise, "IBM's $5,000,000,000 Gamble," *Fortune*, September 1966. 118.
[9] Watson, 347.
[10] Campbell-Kelly, *Computer,* 128.

terminate their IBM contracts and for the same cost, seamlessly migrate to the more powerful Honeywell 200.[11]

IBM responded with the debut of System/360 on April 7, 1964. They maximized publicity by chartering a train to bring 200 reporters from Manhattan to Poughkeepsie for the main announcement while simultaneously holding secondary press conferences in sixty-three U.S. cities and fourteen foreign countries. In Poughkeepsie, IBM unveiled six new computers and forty-four peripherals.[12] The marketing blitz was effective; publications such as *Fortune* understood immediately that "System/360 was intended to obsolete virtually all other existing computers — including those being offered by I.B.M. itself."[13]

Part of the revolutionary nature of System/360 was microcode, technology that allowed IBM to run the same software on vastly different hardware platforms. Microcode (also referred to as microprogramming), created an interstitial layer between the hardware and software. Here, hardware-level instructions were stored within the computer processor's read-only memory. This allowed the microcode to act as an interpreter between the system's hardware and software, separating software design from hardware specifics by translating programmatical calls in real time. In addition to allowing consistent software on all hardware platforms, microprogramming also permitted the System/360 computers to emulate existing IBM hardware, such as the 1401.[14] This allowed current IBM customers to migrate to the new platform without needing existing programs to be re-written.

---

[11] Ibid.
[12] Thomas J. Watson, 351.
[13] T.A. Wise, , 118.
[14] Ceruzzi, 148-149.

With the introduction of the System/360 project, IBM also revised its approach to cataloging and distributing software. The company rearchitected their Program Application Library and introduced a four-way classification system to better define what IBM supported under their leasing agreement. Previously supported programs written by IBM or SHARE, were no longer covered by IBM support. (See Table 2) [15]

Table 2: IBM Program Application Library in 1964

Category:	**Description:**	**Support:**
Type I	Systems Software and Operating Systems	IBM
Type II	Application Software	IBM
Type III	Custom Client Software written by IBM	None
Type IV	Programs submitted by users (SHARE)	None

Source: Martin Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*, History of Computing (Cambridge, Massachusetts: MIT Press, 2003), 96-97.

System/360 was an immediate success. Within one month of the April 1964 product announcement, IBM received over 1100 orders for the new product line. The unexpected demand forced IBM to expand its production facilities. System/360 began to ship in 1965, and by 1970, IBM's gross income had more than doubled. The following year, the company's net earnings surpassed $1 billion.[16]

System/360 was intended to address the software problems facing IBM and its clients. But the technology also resulted in unintended side effects. The platform turned out to be a watershed event in computing; software was now conceptually independent of hardware. System/360's emulation abilities also afforded remarkable longevity to the

[15] Martin Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog*. 97.
[16] Ceruzzi, 145.

software written for early mainframes. Programs written on an IBM 1401 in 1960 could run nearly indefinitely with System/360 (and later descendants S/370 and S/390). The unintended result of this manifested in the late 1990s when date limitations in still-active IBM 1401 software contributed to the "Year-2000" issues.[17]

System/360 also had the welcome effect of broadening IBM's appeal to new customers. Now, clients could start their relationship with IBM by leasing or purchasing a low-end IBM computer, and as their needs grew, they could migrate to new hardware without business disruption. This flexibility helped IBM's market share, which had already been 70% by the mid-1960s, to continue to grow.[18]

IBM Unbundles

Unfortunately for IBM their increasing market dominance caught the attention of the U.S. Department of Justice. The government filed an antitrust suit against the corporation in January 1969. Pressured by the pending suit, IBM took an extraordinary step and changed how they did business.[19] On June 23, 1969, IBM sent a letter to its customers explaining that it was altering the cost model for IBM products and services. Where the IBM equipment lease previously covered all hardware, software, engineering, and education services, the new pricing model separated products from services. The lease now only covered the equipment and Type I software, programs "fundamental to the operation and maintenance of a system (such as loading, initializing, scheduling,

---

[17] Ibid., 151.

[18] Ibid., 143.

[19] IBMs unbundling was cotemporaneous to the anti-trust suit, but the company has never clearly stated that the change in business model was a direct result of the suit. IBM first suggested that they were going to unbundle in December 1968, several weeks before the government filed the suit against the company.

supervising, and data management.)"[20] IBM still freely shared software (Type III and IV), but these programs were unsupported. Previously supported applications (Type II) were also no longer covered. Going forward, IBM's letter announced, all Type II software such as "language compilers, general purpose utility programs, and industry or other application programs" would require an additional purchase and separate software licensing agreement.[21]

Just as with System/360, this action brought unintended consequences. Where System/360 had blurred the division between software and the hardware that it ran on, IBM's unbundling created a literal separation. It gave a separate market value to software, which had previously been provided free of cost. Software was now a separate capital good, sold and supported by a software company.[22] With this, the commercial software industry was born.[23]

Cultural Changes: From Business to Hobbyist

While IBM's 1969 unbundling allowed the corporate software market to emerge, a consumer software market was unable to take root due to a lack of established

[20] F.G. Rodgers, *IBM Product Announcement Letter* (Armonk, New York, New York: International Business Machines, 1969).

[21] Ibid.

[22] Martin Campbell-Kelly, *From Airline Reservations to Sonic the Hedgehog.* 99.

[23] To be clear, IBM was not the first software company, but the corporation's previous practices essentially stifled the emergence of the software industry. IBM had been the subject of several civil suits about the bundling of software. For example, the ADR corporation had attempted to market a software product called Autoflow. They filed suit against IBM in April 1969 arguing that IBM's practice of bundling was a monopolistic suppression of the software market. The company argued that IBM should divest its software line of service and compensate users of bundled software. This was settled out of court. Under the terms of the settlement, IBM paid ADR $2 million and agreed to market ADR products. In this light, the unbundling can be seen less as IBM creating the industry and more that IBM ceased its monopolistic suppression of the nascent industry. See Martin Campbell-Kelly, *Computer: A History of the Information Machine*, 113-115.

consumer hardware market. This began to change with the introduction of the Altair 8800, an inexpensive computer kit that started the personal computing revolution.

Altair 8800

In January 1975, *Popular Electronics* featured an article on the "World's First Minicomputer Kit," the Altair 8800.[24] Invented by H. Edward Roberts and available as a kit from his company, Micro Instrumentation Telemetry Systems (MITS), the base computer cost just $397.[25] The device was designed to appeal to the electronic hobbyist market. The Altair featured Intel's newly released 8080 processor, a 2 MHz clock chip that supported 256 bytes of memory. The Altair was designed for expansion. It introduced the S-100 bus, a modular back plane system that facilitated communication between various parts of the computer.[26] The system bus provided peripheral slots for adding up to 17 secondary plug-in cards. This encouraged expansion through the addition of peripherals such as input/output (I/O) devices and memory.[27] MITS produced several such cards, and openly shared the specifications of the S-100 bus, which allowed others to design and market cards for the computer.[28]

The challenge with the Altair 8800 was in what it did not provide; the device had no keyboard or display. Computer programming was done through machine code entered via toggle switches on the front of the computer. The flickering of lights on the front bezel were the only indication that a program was running, and should an error occur, the

---

[24] H. Edward Roberts and William Yates, "ALTAIR 8800," *Popular Electronics*, January 1975, 33.

[25] Martin Campbell-Kelly, *Computer,* 235.

[26] Roberts and Yates. 38.

[27] The Intel 8080 microprocessor could theoretically address up to 64k bytes of RAM.

[28] Roberts and Yates, 38.

whole manual process had to be redone. On top of this, the amount of memory limited the programs that the device could run.[29]

Micro-Soft

Inspired by the Altair feature in *Popular Electronics*, Harvard undergraduates Paul Allen and Bill Gates established the Micro-Soft Corporation in April 1975 to provide software to the emerging "hobbyist" computing market. Using an emulator on Harvard's PDP-10, the duo ported Dartmouth's BASIC programming language to the new MITS computer.[30] Their BASIC interpreter needed to fit within 4K of RAM, so they omitted some features of Dartmouth's original programming language.[31] As soon as the software was working, Paul Allen flew from Boston to MITS headquarters in Albuquerque, New Mexico to demonstrate the program. After successfully loading the paper tape and exhibiting their version of BASIC, Allen and Gates sold Ed Roberts on an arrangement where MITS would distribute the software as Altair BASIC and Micro-Soft would receive royalties for every copy sold.[32]

Paul Allen and Bill Gates were not the only ones inspired by the new Altair computer. By February 1975, hobbyist groups such as the "Homebrew Computer Club" emerged to allow users to "exchange information, swap ideas, talk shop, help work on a project...."[33] Orders for the Altair were backlogged, but those who received them were

---

[29] Campbell-Kelly, *Computer*, 235-236.

[30] The *Popular Electronics* article included detail schematics of the Altair. Using these, Gates and Allen were able to develop the code on an Intel 8080 emulator running on Harvard's PDP-10.

[31] An interpreter is a program that takes code or pre-written subroutines and translates and executes them a line at a time. An interpreter is generally slower than a compiler, and it does not create a binary copy of the program being translated.

[32] Levy, 232.

[33] *Amateur Computer Users Group Homebrew Computer Club Invitation* (Menlo Park, California: Homebrew Computer Club, 1975).

glad to show off their devices to others. Within these communities, there was also a strong desire for Altair BASIC, which was promised to release in June 1975.[34] Enthusiastic hobbyists pre-ordered the software and eagerly waited. In just the first three months following the Altair premier in *Popular Electronics*, over $1 million in orders for the computer and software flooded into MITS.[35]

By the summer of 1975, MITS customers were growing impatient as they awaited delivery of the products for which they had spent hundreds of dollars. Some of those who received equipment from MITS had difficulty with the hardware. MITS memory boards were notoriously faulty. With growing discontent, the MITS company arranged for the "MITS Caravan," a marketing measure where MITS engineers travelled from city to city in a motorhome. The engineers held seminars where they demonstrated the Altair and Altair BASIC to reassure the discontented consumers. The road show went to the Rickey's Hyatt house in Palo Alto in June 1975, and many Homebrew Computer Club members attended. They were delighted to see the Altair. The computer used a teletype console and ran Altair BASIC, which had been loaded via paper tape. During the demonstrations, one of the visiting hobbyists purloined a paper tape copy of BASIC. The thief approached Homebrew member Dan Sokol, who was known to have access to a tape-copying machine.[36] Sokol took the tape and made copies, which he distributed at the next Homebrew Computer Club meeting.[37]

---

[34] Ceruzzi, 233.

[35] Campbell-Kelly, *Computer,* 236.

[36] It remains unclear who, specifically, stole the paper tape. Sokol readily admitted to making the copies, justifying his actions with the rationale that many of the hobbyists had already purchased, but not received, the software.

[37] Levy, 231-233.

An Open Letter to Hobbyists

When Micro-Soft's Bill Gates learned of the unauthorized distribution of his BASIC interpreter, the 19-year old programmer responded by writing an "Open Letter to Hobbyists." Sent to industry periodicals and computer clubs in January 1976, Gates excoriated the user community for sharing software. He explained that he, Paul Allen, and Monte Davidoff had spent a year developing versions of the BASIC and had invested an estimated $40,000 in computer time in the process.[38] He reported that feedback from BASIC users revealed that most users had not purchased the software, stating "less than 10% of all Altair owners have bought BASIC" and as a result, the time spent on developing the product came out to "less than $2 an hour."[39] He attributed the problem to the hobbyists, claiming, that "…most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?"[40]

Gates claimed that sharing software prevented "good software from being written," because without royalties, companies like Micro-Soft would be disinclined to provide software to the Hobbyists. Gates dismissed the hobbyist programmers, asking, "Who can afford to do professional work for nothing? What hobbyist can put 3-man

---

[38] Since Gates and company developed the software on Harvard's ARPA-funded mainframe, it is unclear who was being paid for the computer time.

[39] Gates' 10% statistic is misleading. The Altair shipped with 256B of RAM. Altair BASIC required 4K RAM. Unless they purchased additional RAM at a significant cost, Altair owners would not have been able to even run Gates' BASIC interpreter. It is unclear how many owners committed to such an upgrade.

[40] Bill Gates, "An Open Letter to Hobbyists," *Homebrew Computer Club Newsletter*, January 31, 1976.

years into programming, finding all bugs, documenting his product and distribute for free?" The letter then denigrated the hobbyists as thieves.[41]

Gates's letter provoked a strong reaction within the hobbyist community. The month following the missive, the Homebrew Computer Club newsletter printed a letter that editor Robert Reiling suggested may "represent the predominant thinking of hobbyists on this subject."[42] The response, written by Dr. Michael Hayes of MNH-Applied Electronics, thanked Gates for the software and suggested that Gates's sales model and pricing was the real issue. Hayes suggested that Gates should give more thought to the audience to whom the software was being marketed and advised the belligerent young businessman that "calling all your potential future customers <u>thieves</u> is perhaps 'uncool' marketing strategy!"[43] Hayes's perspective was echoed by other community members. In a July 1976 letter to *Byte* magazine, reader Robert Wada responded to Gates's letter by attributing the financial woes of Altair BASIC to larger issues facing software vendors. He pointed out that the industry lacked a method for delivering high volumes of software, and because of the limited market size, vendors overpriced their products to recover costs. This made purchasing software uneconomical and, as such, encouraged the duplication and sharing of software.[44]

Other periodicals featured similar perspectives. In the next newsletter of the People's Computer Company, a Bay Area group that encouraged computer literacy, editor Jim Warren, echoed this perspective, observing that "When software is free, or so

---

[41] Ibid.

[42] Robert Reiling, "This Month," *Homebrew Computer Club Newsletter*, February 29. 1976, 1.

[43] Robert Reiling, "Regarding your Letter of 3 February 1976 Appearing in Homebrew Computer Club Newsletter Vol. 2 No. 1h," *Homebrew Computer Club Newsletter*, February 29. 1976, 2.

[44] Robert Wada, "AN OPINION ON SOFTWARE MARKETING," *Byte*, July 1976, 90-91.

inexpensive that it's easier to pay for it than duplicate it, then it won't be 'stolen'."[45] Warren goes on to advocate for free, communally developed software. As one example, Warren cited the free alternative to Gates's Altair BASIC, Tiny BASIC. Warren explained that there were at least five versions of the BASIC interpreter already running on three processors, and that a cassette containing the Tiny BASIC source code for the Intel 8080 cost only five dollars.[46] In contrast, the list price for Micro-Soft's Altair BASIC was three hundred and ninety seven dollars.[47] He also pointed out that Tiny BASIC was freely available as a type-in program published in *Dr. Dobb*s.[48] Because the source code was available, Warren reasoned, "anyone is welcome to retype it and reassemble it. No one will yell, 'thief.'"[49]

From there, Warren's response goes beyond promoting Tiny BASIC. He provides an incisive retort to Gates's rhetorical questioning of "who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free?"[50] Warren explains that since hobbyists enjoy computers as a hobby, a form of recreation, they perceive programming as fun, rather than work, and are thusly willing to share what they create, as is "the usual

[45] Jim Warren, "Response to Bill Gates Open letter to Hobbyists," *People's Computing Company Newsletter*, April 10, 1976.

[46] Warren also mentioned another example: programmer Gary Kildall was writing an operating system that he planned to sell for not much more than duplication costs. Kildall later completed his operating system, CP/M. A clone of Kildall's operating system was later sold by Microsoft as MS-DOS.

[47] "Artifact Details: Altair Basic Interpreter Source Tape, Replica," Computer History Museum, last modified April 7, 2002, https://www.computerhistory.org/collections/catalog/102631998, (accessed January 27, 2020).

[48] Jim Warren was also the editor of *Dr. Dobbs*.

[49] Jim Warren, "Response to Bill Gates Open letter to Hobbyists," *People's Computing Company Newsletter*, April 10, 1976.

[50] Bill Gates, "An Open Letter to Hobbyists," *Homebrew Computer Club Newsletter*, January 31, 1976.

practice in most other hobby environments."[51] He points out that many hobbyists are experienced computer professionals, and their competency is "more than sufficient for the design and implementation of excellent system software."[52] Where other readers had simply justified sharing due to high software costs, Warren countered Gates's argument that the hobbyists were unprofessional.

Not all hobbyists disagreed with Bill Gates. In the March 31, 1976 *Homebrew Computer Club Newsletter,* club member Charles L. Pack sent a letter supporting Gates. Pack volunteered that he was an Altair customer who purchased 8K of Memory and Altair's 8K version of BASIC. He indicates that he had no objection to paying for the software and the hardware needed to run it, and moreover, he resented "the fact that people are getting bootlegged copies."[53] Pack's investment in the technology was significant. He would have spent $397 on the Altair, $528 on the RAM, and $75 for a discounted version of Altair BASIC (adjusted for inflation, Pack spent $4,611 in 2020 dollars.)[54] Pack shared that his plan was to write business software for the new microprocessors, and as a fellow entrepreneur, he sympathized with Gates's perspective, "If I were to spend hundreds of hours of my time – not to mention some money too – on a general package, why should I give it away for someone else to sell services on?"[55]

In March of 1976, MITS held the World Altair Computer Convention (WACC) in Albuquerque New Mexico.[56] Bill Gates attended the event and gave a speech that

---

[51] Jim Warren, "Response to Bill Gates Open letter to Hobbyists," *People's Computing Company Newsletter*, April 10, 1976.

[52] Ibid.

[53] Charles L. Pack, "Response to Bill Gates," *Homebrew Computer Club Newsletter*, March 31, 1976, 3.

[54] *MITS Christmas Catalog 1975* (Albuquerque, New Mexico: MITS Corporation, 1975), 36-37.

[55] Pack, 3.

[56] *Byte*, World Altair Computer Convention Advertisement, March 1976, 17.

attempted to quiet the response to his controversial letter. Gates avoided the heavy-handed rhetoric from his original commentary, and instead he recast his allegations toward the hobbyists as simple concern over the future of software development. Following the event, Gates summarized his comments in a follow-up “Second and Final Letter” to the hobbyist community. In the missive, Gates decried the pushback from hobbyists, who “focused upon me personally and even more inappropriately upon MITS.”[57] Despite this admonishment, Gates then suggested that he only received three negative letters, and he reiterated his WACC speech where he suggested that his reason for writing the first letter was “to open the issue for discussion,” and to express concern “for the future of software development.”[58]

In his second letter, Gates also tried to clarify the apparent inconsistencies in his own conception of sharing software. While Gates’s first letter unilaterally dismissed the sharing of software as stealing, both Gates’s and Altair’s past behavior was inconsistent with this perspective. For example, in the July 1975 issue of *Computer Notes*, the Altair user group newsletter, Bill Gates encouraged Altair users to submit Free Software for the Altair Users software library, gleefully reporting:

> We started receiving programs for the Altair Users Library a few months ago, and we are getting more and more every day. As the Users Group grows and users become more sophisticated at programming their Altairs, we expect the library to grow at a fast rate and become a valuable resource for Altair users.[59]

Similar exhortations graced every issue of *Computer Notes*. The magazine heralded programs shared by users for the Altair library with a monthly contest announcing the best contributions. In the very same issue where Bill Gates’s second letter

[57] Bill Gates, “A Second and Final Letter,” *Computer Notes*, April 1976, 5.
[58] Ibid.
[59] Bill Gates, “Software Contest Winners Announced,” *Computer Notes*, July 1975.

was published, the magazine lauded user Richard Schaal's BASIC program, which reformatted BASIC programs to allow for easy editing.[60] The previous month, fellow Micro-Soft co-founder Paul Allen highlighted "Modifications to MITS ALTAIR BASIC which allow programs to be saved on cassette in source (ASCII) form" submitted by Christopher J. Flynn.[61] Gates and company were against sharing software of which they were the authors, but they seemed happy to coax others into developing and sharing applications to run on Micro-Soft BASIC.

In his letter, Bill Gates addressed the apparent inconsistencies as a matter of complexity. According to Gates, it was fine to exchange "programs less complex than interpreters and compilers."[62] These simple programs could be "written by hobbyists and shared at little or no cost." He suggested that user contributions to software libraries would be common for the foreseeable future, but it was the "standardized compilers and interpreters" such as his BASIC software that would determine "how quickly these libraries develop and how useful they are."[63] Gates's complexity rationale seemed specious when considered in concert with the software sharing present in *Computer Notes*. If software sharing was only fine for software less complex than interpreters and compilers, then why would Micro-Soft accept contributions to their "complex" BASIC interpreter from hobbyists such as Christopher Flynn?

The charges Gates leveled against the hobbyist community exemplified the tension emerging in the industry. Budding software companies such as Micro-Soft understood that it would be difficult to compete in a market dominated by free hobbyist

---

[60] "Resequencer Wins Contest," *Computer Notes*, April 1976, 10.
[61] Paul Allen, "March Software Contest," *Computer Notes*, March 1976, 8.
[62] Bill Gates, "A Second and Final Letter," *Computer Notes*, April 1976, 5.
[63] Ibid.

software. At the same time, traditional attitudes toward software sharing meant that their commercial products might be exchanged. So, to be profitable, commercial software companies such as Micro-Soft first needed to change the prevailing attitudes about sharing software. But, at the same time, these companies profited from the innovations brought to them by the software sharing culture. For example, despite Bill Gates's anti-sharing missives, the Micro-Soft co-founder seemingly ignored the fact that his sole product, BASIC, was the result of this cooperative culture. The BASIC programming language was not an original intellectual property of Gates or Allen – BASIC was developed by John Kemeny and Thomas Kurtz at Dartmouth. Kemeny's techno-idealism and the largess of Dartmouth resulted in BASIC being shared freely with the public. Had BASIC not been free, Gates and Allen would not have been able to embrace Dartmouth's innovation and extend it into their own commercial BASIC interpreter.[64] In other words, Micro-Soft's seminal product would not have existed if it were not for the sharing of software.

Tiny-BASIC

As detailed above, in his response to Bill Gates's "Open Letter to Hobbyists," *Dr. Dobb's* editor Jim Warren took umbrage with Gates's implication that computer hobbyists were unprofessional. He also predicted the growth of the sharing culture, positing that the "approach used in producing the Tiny BASICs will be continued and expanded," and as a result, a large symbiotic sharing community of software

---

[64] Even the $40,000 worth of computer time claimed was dubious; neither Gates nor Allen paid directly for the processing time spent developing the software on Harvard's PDP-10. Harvard's DEC was funded by DARPA, which led to the perspective that the taxpayers had contributed to the development of the software, and thusly sharing copies of BASIC was justified. See Levy, 231-232.

implementers, designers, and users would evolve.[65] Warren's observations were well founded; he was active in the late 1960s counterculture in the Bay Area, and he was involved with a number of counter-cultural groups, including the People's Computer Company (PCC). The Tiny BASIC project that Warren had detailed in his PCC newsletter response to Gates was created in 1975 by fellow PCC member and Stanford University lecturer Dennis Allison. Warren's magazine, *Dr. Dobb's Journal of Tiny BASIC Calisthenics & Orthodontia,* grew out of this effort. The first three issues of the periodical provided type in code to help hobbyists build their own BASIC interpreters.[66] The magazine also included reference guides, design notes, and documentation. So, when Gates suggested that BASIC was too complex for hobbyists and that amateur authors would not be able to dedicate time to "programming, finding all bugs, documenting his product" and distributing it for free, of course Warren chaffed – the Tiny BASIC project was already doing precisely this.

Following *Dr. Dobb*s' initial publication of the Tiny BASIC source code, the periodical focused on providing information about free (gratis) and low-cost software.[67] With this focus, the magazine provides unique insight into the changes that were occurring within the software community. For example, during the Gates letter controversy, *Dr. Dobb*'s editor Jim Warren urged the community to preserve the open communication and sharing of ideas that were common in the field, stating, "It is this open sharing that particularly delights me, and with which I am particularly concerned. I

---

[65] Ibid.

[66] Dennis Allison, "Build your own BASIC," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, January 1976, 3.

[67] The source code was also printed in the *People's Computer Company Newsletter*.

hope that it continues. We must all do whatever we can to encourage it."[68] Despite these hopes, it was apparent that the culture was changing. In a May 1976 *Dr. Dobb*'s article, "COPYRIGHT MANIA: It's mine; it's mine, and you can't play with it!" Warren tells readers of the awkward interactions the magazine had after contacting a software company, Rockford Research, for information about their new computer language, TRAC. The creator of the software, Calvin Mooers, responded by sending *Dr. Dobb*'s an envelope of documents focused on the software's copyright. The packet included:

- A copyrighted price list for the software and documentation
- A statement about the software's copyright protection and registered service mark
- A mimeographed copy of his policy on copyrights and trade-marks
- An article concerning a $3 million suit against some companies that used Mooers' language
- Two mimeographed copies of an article Mooers wrote about software copyrights
- Two reproduced copies of two articles from *Computerworld* magazine[69]

Warren noted that Mooers' perseverations about copyright protections did not seem to extend to others. Warren noted that the programmer neglected to include evidence showing that he had the legal right to reproduce the *Computerworld* articles. Warren then tied the copyright obsession back to Gates's previous blanket indictment of hobbyists as thieves.[70] He dismissed the growing preoccupation with proprietary software as a waste of time and parsed out the key economic challenge for the current approach reasoning that the corporations in the nascent software market failed to distinguish between hobbyist consumers and the business community. As a result,

[68] Jim Warren, "History repeats itself... I hope," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, April 1976. 3.

[69] Jim Warren, "COPYRIGHT MANIA: It's mine; it's mine, and you can't play with it!," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, April 1976. 3.

[70] Warren's assertion that Gates's letter began the copyright concerns is incorrect, at least in the case of Mooers. Mooers had argued for copyright protections for as early as 1968. See Mooers, Calvin N. Letter to the Editor, Communications of the ACM, Vol. ll, No.3.(March,1968). 148-149.

businesses priced their software for corporate clients and then were disappointed when the hobbyist market rejected both their products and pricing. The solution, Warren suggested, was to either charge little for the software and depend on volume for profit or charge a great deal for software through contracts that permitted distribution rights to hardware manufacturers. As an example, Warren refers to the pricing model for *Dr. Dobb*s, which followed the former approach.[71]

For the hobbyist market, cost was not the only consideration. Tom Pittman, one of the Tiny BASIC developers, had experimented with selling his software while charging very little. Pittman, a member of the Homebrew Computer Club, had seen Dennis Allison's "Build your Own Basic," article. At the time, MITS had just released a new model based on the Motorola 6800 processor, the Altair 680. Since the current versions of Tiny BASIC were popular, Pittman was curious if club members would be willing to pay a minimal cost for a 6800 version of the interpreter. He asked fellow hobbyists if they would be willing to purchase a new version of BASIC if it were only $5 (one twelfth of what Micro-Soft charged.) After receiving positive feedback, he wrote the new interpreter, applied for copyright protection, and offered it for sale. He sent a copy to *Byte* magazine, which printed a small announcement about the project. Pittman immediately began to receive orders. Pittman's 6800 Tiny BASIC was well received by hobbyists who gladly purchased the software. "I didn't get rich off it," Pittman recalled, "but it did pay a lot of my expenses at grad school."[72]

[71] Ibid.
[72] Tom Pittman, "Itty Bitty Computers and TinyBasic," Ittybittycomputers.com, last modified July 10, 2017, http://www.ittybittycomputers.com/IttyBitty/TinyBasic/ (accessed April 12, 2018).

In a May 17, 1976 editorial in *Dr. Dobb*s, David Allen, a purchaser of Pittman's 6800 Tiny BASIC, took issue with the software. Although he praised the product, as simple, comprehensive, and "the best $5 I've spent in a long time," he was unhappy that Pittman had refused to release the source code for the program.[73] Allen articulated the hobbyist perspective that cost was less of a concern than the ability to modify the software to work with the purchaser's system. David Allen articulated this need for the source code:

> I need to make some modifications to the program, for use with my cassette O/S, and I would like to be able to expand it. It is very frustrating to be kept so ignorant about his program, particularly since it seems to work so well.[74]

Following Allen's complaint, the magazine printed Pittman's response. Pittman agreed with the concerns with the lack of source code. He revealed that when he wrote the program, it was an experiment in software economics. As a hedge against losses, he sold copies of the source, along with maintenance documentation to an undisclosed company. This initial sale was for much more than what he charged the hobbyists. Although the sale was non-exclusive, Pittman felt that a release of the source code might devalue the company's purchase. The programmer promised to make the source code available for all future projects. This exchange shows how the source code and the ability to change it – not the low price - was the key benefit that the hobbyists sought from their software exchange.

[73] David Allen, "PRAISE FOR PITTMAN'S 6800 TINY BASIC and A Minor Complaint," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, June/July 1976, 3.

[74] Ibid.

The Tiny BASIC community also introduced, albeit in jest, the idea that their shared source code could be copyrighted. In the May 1976 issue of *Dr. Dobb*s, Homebrew Club member Li-Chen Wang provided his version of the Tiny BASIC for the Intel 80808 processor. At the top of the code listing, Wang wrote: [75]

```
TINY BASIC FOR INTEL 8080
VERSION 1.0
BY LI-CHEN WANG
10 JUNE, 1976
@COPYLEFT
ALL WRONGS RESERVED
```

The copyright gag suggested that shared software was the opposite of copyrighted software – so therefore copyleft. But this attempt at humor informed the later use of copyright law to protect Free Software, and this also gave a name to the type of software license: Copyleft.

Legal Changes

One of the interesting aspects of Bill Gates's missives against the sharing culture within the hobbyist community was that he did not invoke the law. He called the hobbyists thieves, and yet he did not pursue them legally for theft. In his letters and speeches about the issue, he did not reference any specific laws that had been broken. Clearly, the individual who stole the original paper tape copy of Gates' BASIC had committed theft. However, when it came to the duplication and dissemination of the tape, any claims of illegality would be tenuous at best due to the lack of laws protecting software.

[75] Li-Chen Wang, "TINY BASIC FOR INTEL 8080," *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, May 1976, 15.

In the early 1970s, the legal safeguards for software were minimal. Patents for software had been unsupported by the courts. In November 1972, the Supreme Court ruled against computer software as patentable in *Gottschalk v. Benson*, 409 U.S. 63 (1972) on the grounds that such a patent would be akin to affording patent protections to an abstract idea.[76] Just a month after Gates's second letter to the hobbyists, another Supreme Court Case, *Dann v. Johnston*, 425 U.S. 219 (1976), again held that software was unpatentable, in this case because the software in question performed obvious processes already commonly done within business. As such, the applicant could not support claims of originality.[77] Software was equally unsupportable as a trade secret since copies of a trade secret could not be distributed.

Since software did not qualify for patents or as trade secrets, the only other potential legal protection was copyright. However, this too was problematic. In the early 1970s the copyright law did not include software. The register of copyrights had begun to accept the submission of computer programs starting in 1964, but the software had to meet three specific criteria. First, the software had to demonstrate sufficient original authorship. Second, the software, like a written work, needed to be for publication. Third, the software had to be submitted to the register in human-readable form – source code.[78] Since copyrights were intended to protect a tangible form of human expression such as written text or painting, non-human-readable media - such as Altair BASIC on paper tape - was not eligible for copyright. Established jurisprudence supported this requirement: in

[76] "Gottschalk v. Benson, 409 U.S. 63 (1972)," JUSTIA US Law, November 20, 1972, https://supreme.justia.com/cases/federal/us/409/63/case.html (accessed April 16, 2018).

[77] "Dann v. Johnston, 425 U.S. 219 (1976)," JUSTIA US Law, March 31, 1976, https://supreme.justia.com/cases/federal/us/425/219/case.html (accessed April 16, 2018).

[78] *Final report of the National Commission on New Technological Uses of Copyrighted Works, July 31, 1978* (Washington, District of Columbia: Library of Congress, 1979), 15.

the 1908 case, *White Smith Music Publishing Co. v. Apollo Co.*, 209 U.S. 1, the Supreme Court ruled that piano rolls – music encoded on perforated tapes of paper – were not covered under copyright law. Here, the court affirmed an earlier decision (*Kennedy v. McTammany),* where the presiding Judge interpreted the paper piano tapes to be "a mechanical invention made for the sole purpose of performing tunes mechanically upon a musical instrument," rather than a creative work.[79]

The National Commission on New Technological Uses of Copyrighted Works

Recognizing that the current copyright laws did not keep pace with technology, the U.S. Congress created the National Commission on New Technological Uses of Copyrighted Works (CONTU) on December 31, 1974. The fourteen-member committee was to be comprised of copyright owners, users, and the public. The group was given three years to examine new technologies that were emerging and challenge traditional copyright models. The commission was tasked to examine:[80]

> (1) the reproduction and use of copyrighted works of authorship:
>
> > (A) in conjunction with automatic systems capable of storing, processing, retrieving, and transferring information, and
> >
> > (B) by various forms of machine reproduction, not including reproduction by or at the request of instructors for use in face-to-face teaching activities; and
>
> (2) the creation of new works by the application or intervention of such automatic systems or machine reproduction.

---

[79] "White-Smith Music Pub. Co. v. Apollo Co., 209 U.S. 1 (1908)," JUSTIA US Law, February 24, 1908, https://supreme.justia.com/cases/federal/us/209/1/ (accessed April 16, 2018).

[80] *Final report of the National Commission on New Technological Uses of Copyrighted Works,*105.?

Upon the completion of their examination of emerging technologies, the committee was tasked to make recommendations for any changes to copyright law that they deemed necessary to “assure for such purposes access to copyrighted works, and to provide recognition of the rights of copyright owners.”[81]

Consistent with their charge, the group made recommendations that attempted to balance the rights of the creators of software with that of the users of the programs. For the creators, the group advised that software be eligible for copyright protection, provided that the copyright did not “inhibit the rightful use” or “block the development and dissemination” of the software. They proposed that the law gave purchasers of programs the right to create archival copies of the product as well as the right to modify the software as needed to “allow its use in the possessor’s computer.” This consumer right to modify programs assumed the availability of source code that could be altered. The commission report suggested that any corporate objections to this right was “a contractual matter;” software manufacturers could nullify the rights permitted within the law through software use agreements.[82]

The changes outlines by the committee redefined the software industry. In affording copyright protections to software, the commission reinterpreted programming as a medium of expression. They gave purchasers of software the legal right to modify it for their own use, requiring that source code be made available. At the same time, they introduced the idea of nullifying copyright law through contractual agreements created by software developers. The result was an industry where software might no longer be sold

[81] Ibid.
[82] Ibid., 12-14.

as a product; instead, consumers merely paid for contractual rights to use the programs within the parameters outlined by one-sided, end user license agreements.

In July 1978, the CONTU commission published their proposals in the *Final Report of the National Commission on New Technological Uses of Copyrighted Works*. The recommendations in the document were far from unanimous conclusions. In a detailed dissent, Commissioner John Hersey argued against the copyright changes recommended in the final report.[83] Hershey's argument focused on the difference between human readable source code and computer readable binary object code. Since binary object code is not comprehendible by humans, it cannot be considered an intellectual expression like a novel or painting, but rather a pre-defined mechanical process that instructs the function of hardware. As such, software manufacturers should only be able to copyright and sell the source code for programs – the human intellectual expression - but the binary object code should not be eligible for copyright protections.

Hershey's dissent is supported by the Supreme Court ruling in *Baker v. Selden* (1879), which held that a specific expression of an idea can be protected by copyright, but not the idea itself.[84] The CONTU recommendations were inconsistent with this, suggesting that that idea, rather than the expression can be copyrighted. For example, when a program is changed or adapted by the software purchaser, the means of expression (i.e. the original program) has been altered and would not be protected under

[83] Dr. John Hershey was a Pulitzer Prize winning author, and while a commissioner on CONTU, he was also the president of the Authors League of America, and secretary of the American Academy of Arts and Letters.

[84] *Final report of the National Commission on New Technological Uses of Copyrighted Works, ,* 31.

previous jurisprudence. Hershey referenced the testimony given by Professor J.C.R. Licklider of MIT. Licklider echoed Hershey's concern for the copyrighting of binary object code, giving an example where a program may be translated from "machine language, or FORTRAN, or whatever level…to a higher level and back to a lower level," a modification wherein all that remains between these versions of software is "the essential underlying idea, not the mode, not the form of expression."[85]

In discussing the social effects of the proposed changes, Hershey also shared a dire prognostication from AT&T Bell Laboratories attorney Robert Nimtz. After reviewing the draft report, Nimtz expressed concerns that the changes would lead to "even deeper secrecy – by encryption, physical barriers to access, contractual restraints, nondisclosure agreements, and further innovative technical tricks for locking out pirates, thieves, and competitors…." Since "secrecy will be seen as the only effective protection" for software, the "public access to innovative programs would likely be inhibited." Hershey echoed these concerns, suggesting that copyright would be leveraged as a "device of industrial security," a blunt means to prevent rather than encourage access to innovation.[86]

The objections expressed by Hershey and Nimtz painted a dark picture of what was to come. They inadvertently foreshadowed the concerns later voiced by Richard Stallman when he created the GNU project. Despite these objections, the CONTU report's recommendations were passed into law without any congressional debate. On

---

[85] Ibid., 33.
[86] Ibid., 34.

December 12, 1980, Public Law 96-517 updated the 1976 Copyright Act and provided unprecedented protections for computer software.[87]

The United States v. AT&T

The CONTU's novel re-interpretation of copyright protection for software was not the only legal change to erode the software sharing culture. On November 20, 1974, the United States Department of Justice filed a new antitrust suit against AT&T and its subsidiaries, Western Electric, Bell Telephone Laboratories, and twenty-two of Bell's Operating Companies (BOC).[88] The complaint alleged that the companies represented a telecommunications monopoly, a violation of the Sherman Antitrust Act. In early January 1982, the parties reached an agreement on a consent decree. The remedy required AT&T to reorganize and divest its twenty-two BOCs. In return, AT&T was freed of the restrictions of their 1956 agreement which had required AT&T to "grant to all applicants non-exclusive licenses for all existing and future Bell System patents."[89] It also required the organization to furnish licensees with "the technical information necessary to manufacture the equipment for which the applicant obtained the patent license."[90] Where the original agreement also had restricted AT&T's line of business to just "the provision of common carrier communications services," the new consent decree freed

[87] "H.R.6933 — 96th Congress (1979-1980)," Congress.gov, December 12, 1980, https://www.congress.gov/bill/96th-congress/house-bill/6933 (accessed April 16, 2018).

[88] The Bell Operating Companies were local telephony service providers.

[89] "United States v. American Tel. and Tel. Co., 552 F. Supp. 131," JUSTIA US Law, February 28, 1983, https://law.justia.com/cases/federal/district-courts/FSupp/552/131/1525975/ (accessed April 17, 2018).

[90] Ibid.

AT&T to pursue other industries.[91] This meant they could now sell the popular UNIX operating system.

Previously AT&T had licensed UNIX through its Western Electric subsidiary. Although restricted to non-commercial research, the operating system was popular in academia due to its nominal licensing fee of $150.[92] By the time of the anti-trust trial, AT&T realized UNIX's potential as a valuable commercial product, and the company began to further restrict license terms accordingly. For example, UNIX licenses now prohibited the studying of the operating system source code in university courses.[93] AT&T did not want public discussions of the technology to endanger their ability to protect it for sale as a commercial product.[94]

Following the consent agreement in 1983, AT&T released System V UNIX. New pricing and licensing terms accompanied the updated version of the operating system. The software cost was now based on the number of Central Processing Units (CPU) in the computer using the software.[95] There were additional charges for programming languages such as BASIC or C. Educational pricing now started at $16,000 per CPU and the source code was not included. For corporate clients, the costs were far steeper. Pricing started with a baseline $25,000 customer agreement. On top of this, a commercial UNIX System V license cost $43,000 for the first CPU and $16,000 for each additional processor. To acquire the UNIX BASIC interpreter or C compiler, a corporation paid

---

[91] Ibid.

[92] Steve Holmgren, *The Network Unix System* (Champaign, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1975), 8.

[93] Salus, *A Quarter Century of Unix* 151-152.

[94] In response, Vrije Universiteit Amsterdam's Professor Andrew Tanenbaum created a version of UNIX, MINIX, that was free of AT&T code and thus able to be used in University courses.

[95] Since the early 1960s, many large computers had multiple processors or CPUs. This permitted multiple processes to be run in parallel on each of the processors.

$5,000 with an additional $2,500 charge per CPU.[96] On-going support was also now available for the operating system, but this was not included in the purchase price.[97] UNIX, which had previously been an inexpensive, community-supported platform suddenly became very, very expensive.

Berkeley Standard Distribution

The AT&T licensing changes were particularly burdensome for institutions that were running the Berkeley Standard Distribution (BSD) of UNIX. Over the course of a decade, this variant of AT&T's operating system had evolved from an add-on software collection to a separate version of the UNIX operating system. In 1974, the University of California, Berkeley had purchased an AT&T UNIX academic license. Almost immediately, Berkeley Professor Bob Fabry and graduate students Chuck Haley and Bill Joy began to make changes to the existing software. The work done by Fabry and his students at Berkeley's Computer Systems Research Group (CSRG) formed a set of add-on packages that they named the Berkeley Software Distribution. They began to freely distribute their software in 1977.[98] In 1978, the group acquired one of Digital Equipment Corporation's new 32-bit VAX computers. The device replaced DEC's popular 16-bit PDP series computers. The change in processor architecture required the group to re-write their software. The result was a new version of UNIX with the BSD add-on packages incorporated within. In 1980, DARPA funded the CSRG to further develop their version of UNIX for use on the ARPAnet. This led to the spread of the BSD operating system, and BSD software, such as Bill Joy's version of the Transport Control

[96] *UNIX System V and Add on Applications Pricing* (Summit, New Jersey: AT&T, 1984).
[97] To be clear, the purchase cost included three months of support.
[98] Salus, *A Quarter Century of UNIX.* 158.

Protocol / Internet Protocol (TCP/IP), the internetworking software that was being developed for the government network.[99]

The AT&T license changes created a substantial problem in that BSD was not supported by AT&T and yet it was widely used. Although it was a separate UNIX distribution, because it still shared source code with the AT&T version, anyone who used it still required an AT&T license. This meant that the AT&T license changes - and associated costs - were still applicable to users of the BSD software. Concerned over the licensing changes, the CSRG began work on a clean BSD release, a version of the operating system with no AT&T source code present.[100] Once the group's programmers had excised the AT&T code, BSD could be distributed without the need for an AT&T UNIX license.[101] The CSRG's recoding effort took years to complete, finishing in late 1988 with the group's new operating system, BSD Networking Release 1.[102] Upon its completion, the operating system was freely shared via Berkeley's servers.[103]

---

[99] The BSD TCP/IP stack was also the basis of the protocol used in later operating systems such as Windows NT.

[100] UCLA and AT&T also worked together to integrate the changes. In March 1986, the University and AT&T established a license agreement for the use of the University's 4.2 BSD and 4.3 BSD OS and in May 1989, made a separate agreement to include the University's 4.3 BSD-Tahoe OS. BSD Network Release 1 was derived from the latter.

[101] Fears over the licensing changes were well founded. In 1992, after they extirpated the AT&T code, the university was still sued over BSD UNIX in *UNIX System Laboratories, Inc. v. Berkeley Software Design, Inc.* The suit held that Berkeley had violated their UNIX license and abrogated the copyright on UNIX. The situation was particularly messy because the author of UNIX, Ken Thompson, had also contributed to BSD. The AT&T version of UNIX had also incorporated some BSD without properly crediting the BSD programmers. As a result, the University countersued. The case was settled amicably in 1994.

[102] Salus, *A Quarter Century of UNIX.* 223.

[103] The software was downloadable anonymously via the file transfer protocol (FTP).

Technology Changes

Just as legal changes challenged the open software culture and disrupted the UNIX user communities, rapid technological innovation unsettled the computing status quo for home consumer and small business users. Changes to both hardware and software fundamentally changed how consumers interacted with their computers – and further distanced the user from the source code they were running.

Home Computing

By the late 1970s the burgeoning consumer market began to offer new computers that were less cumbersome than the MITS Altair. In 1976, Homebrew Computer Club member Steve Wozniac designed his own computer and introduced it to members of the club. Wozniac soon formed Apple Computers with his friend Steve Jobs and re-architected the device for mass production. The result, the Apple II, was introduced on April 15, 1977 at the West Coast Computer Fair.[104] Apple's computer featured BASIC as the default user interface.[105] As with the Altair, the Apple II featured a BUS where cards for peripherals such as cassette readers and disk drives could be added. The computer also featured technology that allowed a television to be used as a monitor.[106]

An Apple II competitor, the Commodore PET computer, was announced in mid-1977 and demonstrated at the Consumer Electronics Show that winter. Unlike the Apple II, which was a simple computer that could be expanded by the consumer, Commodore's

---

[104] Jim Warren, *The First West Coast Computer Faire Program* (Palo Alto, California: Computer Faire, 1977).

[105] The computer's operating system loaded BASIC from a read-only memory chip and presented this as default user interface.

[106] Christopher Espinoza, *Apple II Reference Manual* (Cupertino, California: Apple Computer, 1979), 2-27.

PET 2001 was an all-in-one computer that came with an integrated monitor, keyboard, and a “Datasette” cassette tape recorder for storage. The computer featured expansion ports for adding memory, and it included an integrated parallel port for adding external peripherals.[107] The PET 2001 featured Commodore BASIC (Re-branded Microsoft BASIC) as the default user interface.[108]

Joining the Apple II and Commodore PET, the Tandy/RadioShack TRS-80 also came out in 1977. Like the others introduced that year, it was an 8-bit computer, but the TRS-80 was much more limited in its support for expansion. Like the PET 2001, the computer came with a cassette tape recorder for storage, but unlike the Apple II and earlier S-100 BUS computers, the device did not feature any expansion slots. The computer was designed to fit under the keyboard, and all peripherals were external, connected via cables. The TRS-80 also differed from its competition by having a faster processor.[109] Beyond offering a speed increase, the TRS-80’s processor was compatible with the Intel 8080 architecture type. This allowed Tandy/Radio Shack to use Dr. Li-Chen Wang’s Free Software, “Palo-Alto Tiny BASIC,” as the default user interface.[110] Like the Apple and Commodore, the TRS-80’s Tiny BASIC-based interpreter was included, stored on a ROM chip.[111]

---

[107] Specifically, the device featured Hewlett-Packards’ new IEEE-488 parallel port that was intended for adding printers and other peripherals.

[108] *The PET Personal Computer Brochure* (London, England: Commodore International, Ltd, 1978), 3-6.

[109] Where the Apple II and Commodore PET both used the 1 MHz MOS Technology 6502 CPU, the TRS-80 used the faster, 1.77 MHz Zilog Z80 processor.

[110] TRS-80 units with disk drives also featured TRSDOS, which expanded the limited functionality of TRS-80’s Tiny BASIC derivative by providing support for devices such as the data drives and printers.

[111] *TRS-80 Microcomputer System* (Boston, Massachusetts: Radio Shack, 1977), 2.

With all these early personal computers, the BASIC programming language was the default user interface.[112] This meant that some programming knowledge was required to use the systems, and as a result, the BASIC language became the lingua franca of home computing in this period. Even video game systems such as the Atari 400 featured BASIC functionality. The ubiquity of the language in the emerging personal computer market enabled the sharing of BASIC programs between users and through publications such as "101 BASIC Computer Games."[113] However, despite the popularity of the sharing culture and this type of computing, subsequent technological changes were already in motion. These advancements would soon fundamentally alter the relationship between the user and the computer, and in the process, diminish the popularity of BASIC and of program sharing.

Business Computing

As the computing industry grew, solutions had emerged for different market segments: Large corporations could afford expensive mainframes, but university departments preferred less costly minicomputers like Digital's PDP series. Home consumers could afford neither, so Apple, Commodore, and Tandy/RadioShack introduced low-cost microcomputers. Meanwhile, small-to-medium businesses that were unable to afford a mainframe or minicomputer struggled to find a suitable computing platform. There were options, but they each came with challenges. For instance, IBM introduced the model 5100 computer in 1975. Selling at $9,000, the device offered both

---

[112] Technically, these computers ran an operating system, but they each loaded BASIC as the environment in which the users worked. Other programming languages, or even more advanced versions of BASIC, could be loaded, but the default interface for the users of these computers was through BASIC.

[113] David Ahl, *101 BASIC Computer Games* (Maynard, Massachusetts: Digital Equipment Corporation, 1975), 3-5.

BASIC and APL (A Programming Language) programming environments, but the cost and lack of a vibrant user community resulted in low adoption.[114] There were also many Altair Bus (S-100) based systems available to business consumers. Period magazines are filled with advertisements courting the business market. Vendors such as the Digital Group, Equinox, Heath, MITS, PolyMorphic, Processor Technology and Vector Graphic all offered Intel 8080 chip-based systems for the business market.[115] PolyMorphic Systems, for example, advertised "The Computer for the Professional." Starting at just under $8,000, PolyMorphic's 8813 was advertised as built with the professional in mind; "it quickly and easily processes cost estimates, payrolls, accounts, inventory, patient/client records and much more."[116] Despite these promises, the out-of-the-box functionality was remarkably limited. The computer included only an assembler program and a BASIC interpreter. The actual business software needed to fill the promises of the ad cost substantially more. One vendor, Engram Associates, offered software that provided general ledger, payroll, and accounts payable/receivable functionality for a "discount" of $2250.[117] Whether it be with the IBM model 5100 or one of countless S-100 computers, small-to-medium businesses were faced with a cost of over $10,000 ($39,238.51in 2020 dollars) to obtain a single computer with which to do basic accounting.

[114] Ceruzzi, 248. APL was a programming language developed in the 1960s. It used graphical symbols to represent mathematical functions and has influenced many subsequent innovations such as programming languages, spreadsheets, and mathematic packages like MATLAB. A Micro-Soft APL release was teased by Bill Gates in his *Open Letter to Hobbyists* but was never released.

[115] Lou Frenzel, "How to Choose a Microprocessor," *Byte Magazine*, July 1978, 131.

[116] *Byte Magazine*, PolyMorph Systems Advertisement ,"The Computer for the Professional," July 1978, 13.

[117] Frenzel, 131.

MIT alumni Daniel Bricklin and Robert Frankston introduced software that would change how businesses approached computing. In January 1979, they formed a company called Software Arts. Bricklin had attended Harvard Business School and seen the use of "spreadsheets" on chalkboards. Based on these, he envisioned a software product designed to perform the same functions on a computer. The resulting product, VisiCalc, was made available for the Apple II in October 1979. The program brought the ability to do powerful calculations – formerly available only on a mainframe or with custom software – to a personal computer. In addition, the software was able to be used for accounting ledgers such as accounts payable and accounts receivable. The solution was also inexpensive; VisiCalc cost $200.[118] The Apple II cost $1298, and an add-on 5 ¼ inch disk drive for storage cost $495.[119] The combined result was an Apple computer capable of doing accounting and calculations for less than $2,000 ($6,856.30 in 2020 dollars.) The outcome was that the Apple II, previously a home consumer device, now emerged as a popular small business computer.

The IBM PC

Concerned over Apple's growing presence in the business market space, IBM sought to develop a personal computer with which to match the competition. In July 1980, IBM Senior Manager William C. Lowe proposed a revolutionary plan to IBM's leadership. Lowe posited that IBM should abandon their traditional practices, such as vertical integration, in order to be facile in the growing personal computer market. He suggested that IBM develop standards for their personal computer and then outsource the

---

[118] Ceruzzi, 267-269.
[119] *Byte Magazine*, Introducing Apple II, June 1977, 14.

production of the components needed for the design. Lowe's rationale was that using existing parts would allow IBM to develop and produce a product that they could take to market quickly. The IBM executives approved the proposal. Within two weeks of the meeting, Lowe's team produced a prototype and created a schedule that would have the computer in stores within a year. This was another break with IBM standard practice – the new IBM computer would not be sold solely through IBM but would be distributed through the same retail channels that its competitors used.[120]

IBM's outsourcing was not limited to just the hardware; they contracted for the computer's software as well. IBM initially intended to use the CP/M operating system. Composed in 1974 by Dr. Gary Kildall, a computer science professor at the Naval Postgraduate School in Monterey California, the software had become the standard disk management system for computers running the Intel-8080 processor. The popularity of the system led Kildall to form his own software company, Digital Research.[121] For reasons that remain unclear, IBM and Digital Research were unable to come to an agreement for licensing the operating system.[122] So IBM moved on to their second choice: Microsoft.[123]

When IBM came to call in July 1980, Microsoft lacked an operating system to sell. They also lacked the resources needed to develop one. Desperate for the IBM deal,

[120] Martin Campbell-Kelly, *Computer: A History of the Information Machine*. 245-246.

[121] Jim Warren, "First Word on a Floppy-Disc Operating System," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, April 1976, 5.

[122] Various accounts exist of the disconnect between Kildall and IBM. One version suggests that Kildall refused to agree to IBM's non-disclosure agreement (NDA). Another report suggests that Kildall was out of the office flying a plane when IBM visited with various stories of the result – in one account IBM was frustrated with waiting and left. In another, Kildall's wife, Dorothy was the one that refused IBM's NDA. Regardless of the specifics, the salient point is that it did not work out.

[123] Following their tenure working with MITS, Micro-Soft (short for Micro-Computer Software) simplified their name to Microsoft.

Microsoft founders Bill Gates and Paul Allen bluffed their way through negotiations, confident that they could find a platform to deliver. After securing the IBM contract, Microsoft approached a local company that had worked with them previously, Seattle Computer Products (SCP). SCP employee Tim Patterson had recently developed a clone of the CP/M operating system, Quick and Dirty Disk Operating System (QDOS), to run on the SCP computers. In December 1980, SCP renamed the software to 86-DOS and began to offer it for sale to other equipment manufacturers. In July 1981, Microsoft purchased the full rights to the 86-DOS OS from Seattle Computer Products for $75,000. The software was renamed again, becoming Microsoft Disk Operating System (MS-DOS) and Microsoft's rebranded operating system was ready for the August 1981 debut of the IBM PC.[124] IBM's personal computer release also demonstrated software that could run on Microsoft's MS-DOS operating system – games, word processors, and, most critically, VisiCalc.[125]

The IBM PC was priced at $2880, higher than its competitor, the Apple II. As a result, initial sales were sluggish. The company had originally estimated total sales of 250,000 units. By the following year, however, the computer became popular and began to surpass IBM's sales expectations. Other hardware vendors began to market their own versions of the PC using similar off-the-shelf hardware. These clones were able to run PC software like MS-DOS. The popular computer was christened the 1982 "Machine of the

---

[124] David Hunter, "The Roots of DOS: Tim Paterson," Paterson Technology, http://www.patersontech.com/dos/softalk.aspx, (accessed July 2, 2018).

[125] Because the PC was based largely on commodity hardware, IBM's competitors were able to quickly reverse engineer the device and create their own PC clones. When sold with the IBM PC, MS-DOS was branded as PC-DOS as part of the licensing agreement with IBM. With other manufacturers it was sold as MS-DOS. IBM's market dominance with the PC was short lived, as was the popularity of VisiCalc. By 1983, programmers Jonathan Sachs and Mitch Kapor released Lotus 1-2-3, a spreadsheet product that surpassed VisiCalc by taking full advantage of the PC's hardware.

Year" by *Time*.[126] By 1984, the PC and its many clones were selling over 2 million units each year.[127]

The Graphical User Interface

Starting with the August 1981 release of the IBM PC, the nature of personal computing began to change rapidly.[128] On January 19, 1983, Apple introduced the Apple Lisa. The device marked the company's move away from the BASIC centric interface. The LISA featured a Graphical User Interface (GUI), which used a pointing device, such as a mouse to navigate through a graphical representation of the computer's file system and programs.[129] The GUI did not originate with Apple, but rather was Apple's version of technologies developed in the late 1960s by Douglas Englebart at the Stanford Research Institute's Augmentation Research Center and that of other researchers at Xerox's Palo Alto Research Center (PARC) in Menlo Park, California.[130] Apple co-founder Steve Jobs had brought his company's engineers to several demonstrations of Xerox's experimental technologies in 1979. The engineers included Xerox's innovations in their planning for the Apple II replacement.[131] Apple wrote their own version of Xerox's GUI interface, and they licensed patented innovations, such as the mouse, from Xerox for a paltry sum.[132] The resulting product, the Apple Lisa cost almost $10,000 –

[126] *Time*, Machine of the Year, January 3, 1983.

[127] Jeremy Reimer, "Total Share the 8-Bit Era: 30 Years of Personal Computer Market Share Figures," ARS Technica, December 15, 2005, https://arstechnica.com/features/2005/12/total-share/4/, (accessed July 2, 2018).

[128] By this time, Apple had also introduced the Apple III, a revision of the Apple II that was targeted as a business-orientated PC competitor. The devices sold poorly and were quickly discontinued.

[129] Andrew Pollack, "Apple's Lisa Makes a Debut," *The New York Times*, January 19,1983, D1.

[130] John Markoff, *What the Dormouse Said--: How the Sixties Counterculture Shaped the Personal Computer Industry* (New York, New York: Viking, 2005), 148-152.

[131] Ibid., 240-241.

[132] The GUI, mouse, and networking technologies developed by Englebart and his team at SRI had been demonstrated publicly as early as December 1968. The group began to manufacture computers that

almost eight times the cost of the Apple II. Due to the device's prohibitive cost, sales were low.[133] Despite its poor commercial reception, the computer introduced the graphical user interface to the mass market, and other, less expensive options quickly followed. The PC-based GUI, Microsoft Windows, was announced in November 1983, and Apple's second foray with the GUI, the Macintosh, was introduced in January 1984.

This shift to the PC, and later to graphical interfaces such as Windows, fundamentally changed how users interacted with the computer. Early personal computers were BASIC-centric, and the default user experience was centered on programming in BASIC; now, using a computer was focused on clicking icons to run programs.[134] Of course, programming languages could still be used: Microsoft's DOS included BASIC, but the user's experience was no longer focused on the programming language, and programming skills were no longer a veritable pre-requisite. By 1986, direct mode BASIC devices no longer held any of the home computer market share.[135] (See Table 3.)

---

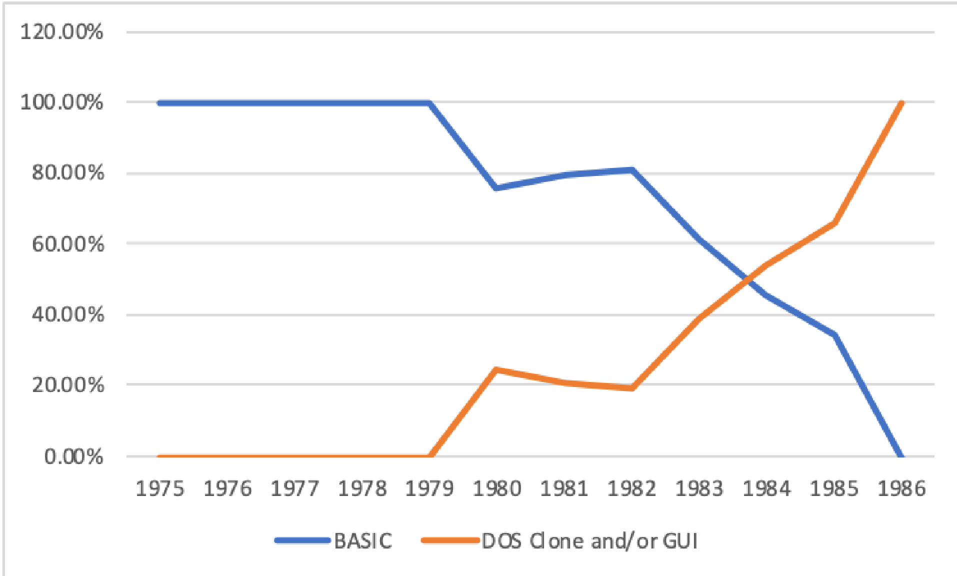
used these technologies, the Xerox Alto (I & II) as early as March 1973, but the devices were expensive and not intended for mass-market sale. The Apple engineering team thus had familiarity with the technologies long before their visit to SRI.

133 Caroline Moss, "From the 'Apple Lisa' to the U2 Ipod: Apple Products That Totally Flopped," *Business Insider*, November 5, 2013, http://www.businessinsider.com/10-old-apple-products-that-totally-failed-2013-11#the-apple-iii-1980-1981-1, (accessed July 2, 2018).

134 Although BASIC was a programming language, the early home computers loaded the language in direct mode, where it served as the default user interface.

135 The table is provided to give an overall sense of environment based on market share. The data should not be interpreted as absolute. For example, the chart represents new purchases of machines with CP/M, DOS, or later Windows vs. computers running BASIC in direct mode. This does not mean that previously purchased direct mode devices stopped being used. Likewise, there is some overlap between the devices; The IBM PC was able to have an add-on card with BASIC in ROM, and the Apple II could use the Microsoft Z-80 SoftCard that emulated a Zilog processor to run CP/M and Microsoft BASIC.

Table 3: Default User Environment by Computer Market Share



Source: Adapted from Retro Computing. "Total Market Share." Accessed July 2, 2018. http://www.retrocomputing.net/info/siti/total_share.html.

The industry changes brought by the adoption of the PC architecture were significant as well. The PC and PC clones emerged as the de-facto standard for the home and small offices; soon the Apple computer ecosystem was the only remaining viable competitor. This standardization solidified the position of companies such as Intel and Microsoft, around which the PC standards were based. This, in turn, allowed for the growth of a commercial software industry. Instead of having to develop and distribute software for multiple programming languages or operating systems, software was now generally developed for either the PC or the Apple. Having a ready supply of off-the-shelf software lowered the skills required to use a computer and hastened the decline of the development and sharing of BASIC programs. Instead of typing in each line of a BASIC program, it was much simpler to purchase a program and run it.

These changes also created a layer of abstraction between the user and the source code. With pre-packaged binary programs, and standardized hardware and operating system platforms, the average user had no need to modify source code to make software work on their system. As a result of these point-and-click systems, using the operating system's command line interface became rare, and perusing source code even more so. The underlying technology of home computing had changed, and with it the practices surrounding software.

Summary: The Closing of the Source

The transition from open to closed software resulted from a series of developments spanning almost two decades. With the introduction of System/360 in 1964, IBM divorced software from the hardware upon which it ran. The company's June 1969 decision to unbundle sales of hardware from software and service introduced a market separation between hardware and the programs that ran on it. Software, which had previously been freely shared became a commodity with intrinsic value.

The introduction of the Altair 8800 in January 1976 helped launch the hobbyist market – and triggered the creation of the Microsoft company. Microsoft co-founder Bill Gates embraced the industry developments and called for an end to the open sharing of software that was common in the computer culture at that time. Meanwhile, many programmers opposed Gates's perspective. They maintained the culture of shared software by writing and distributing software like Tiny BASIC, and they supported periodicals such as *Dr. Dobb*s, which provided free and low-cost alternatives to commercial products.

The transformation occurring in industry and culture was further reinforced by legal changes. The emerging software industry was bolstered by far-reaching modifications to U.S. copyright law. Software, which had been previously unpatentable, was given unprecedented protection as copyrighted material. The revisions to the copyright law reified the consumer's right to modify the source code of the programs they owned while simultaneously providing corporations a mechanism to curtail this privilege through end user license agreements.

Technological development also brought change to the open sharing culture. In the 1970s, home computers such as the Apple II, Commodore PET, and TRS-80 established BASIC as the default environment of home computing. With the BASIC programming language as the de facto standard for microcomputers, the free exchange of programs was common. However, the introduction of the IBM PC, and later the graphical user interface, changed this. Computers no longer booted directly into BASIC, but instead into a graphical interface. This provided a layer of abstraction between the computer's user and the command-line focused world of programming. At the same time, the popularity of the IBM PC and its countless clones thinned the consumer computer market, reducing it to PC versus Apple. This restricted focus fostered the off-the-shelf software industry as it became easier to support software for just two platforms. With readily available commercial software it became easier to purchase and run a program than to spend time typing a shared program in to BASIC.

By 1983, the software sharing culture had faded. Industrial, cultural, legal, and technological changes had successfully reinterpreted what software was, and in the process, diminished the role of program sharing. This transformation was perhaps most visible in the company where the software industry began, IBM. In February 1983, the company announced that they would no longer provide the source code for any of their programs. IBM also began to reconsider the applications they had made available to customers. The corporation restricted previously Free Software such as the CP/CMS operating system.[136] The computing giant experienced a backlash from disgruntled customers, with some corporations going as far as refusing to upgrade from older versions to which they still had the source code.[137] However well intended, the protests fell on deaf ears. The source had closed.

---

[136] IBM closed the source, re-branded CP/CMS as VM/CMS, and then released it as a commercial product.

[137] Paul Gilin, "Do Users Fear IBM Backlash?" *ComputerWorld*, May 21, 1984, 37.

CHAPTER 3: FROM ACADEMY TO BUSINESS

By early 1983, computer software had become a separate and distinct industry from hardware. The cultural, technical, and legal changes of the late 1970s and early 1980s had curtailed software sharing practiced at all levels of the computer industry. In mainframe computing, software sold and supported by IBM was now only available in executable form, and source code was no longer available. In the minicomputer market space, UNIX, the previously open and inexpensive operating system was now closed and expensive to acquire. Microsoft and Apple dominated the microcomputer market, and the once-common sharing of BASIC programs faded in favor of purchasing off-the-shelf software.

By September 1983, MIT Artificial Intelligence (AI) lab programmer Richard Stallman had grown increasingly disillusioned with the commercialization redefining the industry around him. Stallman, an expert programmer famous for EMACS, a customizable text editor, experienced the corporatization of software in a decidedly personal way. In the fall of that year, Stallman had been the target of corporate malfeasance; an AI lab associated company had spied on the programmer, issued ultimatums to restrict his activities, and had actively encouraged employee attrition within the MIT AI lab. For Richard Stallman, the AI lab was his world; when the AI hacker culture was eroded by a predatory corporation, Stallman felt like “the last survivor of a dead culture…. I don’t really belong in the world anymore.”[1]

Stallman’s apparent existential crisis may seem hyperbolic, but his sense of personal attachment to the AI lab culture was genuine. This chapter explores the AI lab,

---

[1] Steven Levy, *Hackers* (Sebastopol, California.: O'Reilly Media, 2010), 450.

how Richard Stallman came to be involved with it, and the events that led Stallman to mistrust the corporations within the burgeoning software industry. Stallman's reaction to the issues he experienced led to the creation of the GNU project, which subsequently became the basis of the Free Software Movement.

Richard Stallman

Richard Stallman grew up living in a small, one-bedroom apartment in Manhattan's Upper West Side. In *Free as in Freedom,* a biography of Richard Stallman, author Sam Williams reported that Stallman was a precocious child with a natural ability to solve puzzles and understand complex mathematical problems. In elementary school, he refused to write papers, preferring instead to focus on exercises that required analytical thinking.[2] Stallman had little interest in playing games with other students, and instead found that his only friends were his teachers. One such teacher recommended Stallman for the Columbia Science Honors Program, a curriculum of weekend courses for gifted students at Columbia University. But even among the other mathematical prodigies in the program, Stallman was considered strange. "He was unusually poorly adjusted," classmate Dan Chess recalled, "but I think he was the smartest person I've ever known." Another alumnus, Seth Breidbart, described Stallman as "very intense…but also very hardheaded in some ways."[3] This observation was echoed by Stallman's mother, who recalled her son being single-minded about individual freedoms from a very young age. She reports that as a teenager, Stallman was conservative, and he would stubbornly argue the issues of the day with her and other adults. "He saw unions as corrupt. He was also

[2] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software* (Sebastopol, California: O'Reilly, 2002), 25-28.

[3] Ibid., 29.

very opposed to social security." She recalled that by the late 1960s, Stallman's political stubbornness manifested on the other end of the political continuum as the specter of the Vietnam war hung over the adolescent. Stallman was strongly opposed to the conflict, which he felt to be immoral, but he was frustrated with his political impotence. Despite being against the war, Stallman did not care for the counterculture. He opposed the anti-war movement's use of drugs and their apparent prejudice against technology, both issues Stallman perceived in moral terms.

Richard Stallman's first exposure to programming came when a summer-camp counselor lent him a manual for an IBM 7094 mainframe. Lacking access to the actual computer, Stallman began to write out programs on paper, eager for the day that he would be able to run them on a real machine. In his junior year of high school, he finally obtained access to a physical computer while working for the IBM New York Scientific Center, a research facility in Manhattan. It was there he ran his first program.

By the time he graduated in 1969, Stallman was accepted at Harvard. Toward the end of his freshman year, Stallman learned of the MIT AI lab. With its focus on artificial intelligence, the research facility represented the cutting edge of computing. Stallman visited the lab, reasoning that he might come away with some spare manuals that would help further his computing skills. Instead of a stack of books, the Harvard freshman left with a job.[4]

The AI Lab

The MIT Artificial Intelligence (AI) lab was an offshoot of Project MAC (Multiple Access Computing), a long-term computing project funded by the Defense

[4] Ibid., 26-27,33-46.

Advanced Research Agency (DARPA) starting in 1963. Initially led by computer scientist J. C. R. Licklider, the effort focused primarily on developing computing technologies, with only a third of the funding dedicated for research into the emerging field of artificial intelligence.[5] The university's proficiency in AI research had long been established, with the MIT artificial intelligence lab previously having set the standard for AI computing with MIT professor John McCarthy's AI programming language, LISP, in 1958.[6] By the time that Stallman joined the group in 1970, the research team had secured their own lab facilities and were managed by AI pioneer Marvin Minsky.[7]

Minsky's team were self-proclaimed hackers, individuals who loved "to program and [enjoy]s being clever about it."[8] In his 1985 book, *Hackers: Heroes of the Computer Revolution,* journalist Stephen Levy explored the AI lab hacker culture, which he identified as emerging from the MIT Tech Model Railroad Club (TMRC) in 1959. According to Levy, the term hacker referred to an individual who applied cleverness and determination to create innovative customizations (hacks) for electronic equipment such as train switches and relays. As the TMRC group moved into working with computers, the terminology transferred to this new electronic medium. According to Levy, an ethic associated with hacking also emerged at MIT. This ethos held that access to computers should be unlimited and total, all information should be free, and authority should be

---

[5] Levy, , 59.

[6] Ibid., 12-13.

[7] Ibid., 48.

[8] Richard Stallman, "The GNU Operating System and the Free Software Movement," in *Open Sources: Voices from the Open Source Revolution.* Ed's. Chris DiBona, Sam Ockman, and Mark Stone (Beijing: O'Reilly, 1999), 53.

mistrusted. Adherents believed that hackers should be judged by their hacking, as it is possible to create beauty and art on a computer.[9]

According to Richard Stallman, it was in the MIT AI lab that he felt at home for the first time in his life. “It was a breath of fresh air,” Stallman recalled. “At the AI Lab, people seemed more concerned about work than status.” Stallman focused on his classes at Harvard from Monday to Thursday, but come Friday, he immersed himself in the hacker culture. For Stallman, this meant spending the weekend at a terminal, programming for 36 hours straight – if that is what it took to write a good program. It meant participating in lab rituals, such as quests for Chinese food every Friday night. Most of all, it meant adopting the group’s sense of communal responsibility, the need for every individual to personally contribute through clever hacks.[10] If a program failed to do something that was needed, the meritocratic programmers would laud whomever took the source code and updated it with a clever hack. The resultant software would be shared with the rest of the group.

In the lab, Richard Stallman worked alongside hackers who were already famous for their technical contributions. MIT student Bill Gosper had come to the AI lab after becoming enthralled with programming while taking a class with LISP creator John McCarthy. Like Stallman, Gosper was extremely gifted in mathematics, and he became fascinated with programming solutions to puzzles. He became renowned for his work in creating Project MAC's computer algebra program, Macsyma, before leaving for a post-doctoral position at Stanford.[11]

---

[9] Levy, 28-32.
[10] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software, 77,47-50..*
[11] Levy, 140-146.

Another famous AI lab hacker was Richard Greenblatt. Described by Levy as "a hacker's hacker," Greenblatt was the quintessential AI lab denizen.[12] A dedicated programmer, Greenblatt nearly lived in the lab, and his unkempt appearance seemed to reify the stereotype that had begun to emerge about programmers as "bright, young men of disheveled appearance, often with sunken glowing eyes" who were transfixed with computers.[13] As a TMRC member, Greenblatt had written the first FORTRAN compiler for the PDP-1 so that the club could calculate schedules for the complex train system they supported. He later developed Mac Hack, the first chess program to compete in a human chess tournament.[14] The gifted programmer was also co-creator of the AI lab's ITS Operating system (ITS), and later the inventor of a general purpose AI computer, the LISP machine.[15]

By 1974, Richard Stallman had graduated from Harvard with a bachelor's degree in physics. He enrolled as a graduate student at MIT, but after a year in the program, Stallman dropped out to spend more time working as an employee in the lab.[16] Stallman's life was centered on the programming community at the University, where the compulsive programmers would "…work until they nearly drop, twenty, thirty hours at a time. Their food, if they arrange it, is brought to them: coffee, Cokes, sandwiches. If possible, they sleep on cots near the computer. But only for a few hours—then back to

---

[12] Ibid., xii.

[13] Joseph Weizenbaum, "Computer Power and Human Reason: Science and the Compulsive Programmer," Santa Anna College, https://www.sac.edu/AcademicProgs/Business/ComputerScience/Pages/Hester_James/HACKER.htm, (accessed July 17, 2018).

[14] Levy, 63-64, 82.

[15] Alan Bawden et al., *Lisp Machine Progress Report* (Cambridge, Massachusetts: Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1977). 1.

[16] Richard Stallman, "Richard Stallman's Personal Site: A Serious Bio," Stallman.org, https://stallman.org/biographies.html, (accessed July 17, 2018).

the console or the printouts."[17] Working full time in this intense atmosphere, Stallman soon established himself as a bona fide hacker. In 1975, he developed the Emacs text editor, a program designed to accommodate the programming activities commonly done in the AI lab.[18]

Officially, Stallman was responsible for supporting the lab's two primary computing platforms. The first was Richard Greenblatt's ITS operating system. The other was the LISP Machine OS, an operating system based on the LISP programming language.[19] Since both systems had been developed at the university, Stallman frequently needed to perform custom programming to address any functionality needs that arose.

Richard Stallman enjoyed the informal atmosphere at the MIT AI lab, feeling at home in its cooperative, software-focused world. The group hacked away at their programs and shared the software they produced. "It was a bit like the Garden of Eden," according to Stallman, "It hadn't occurred to us not to cooperate."[20] And as with the antediluvian paradise, Stallman saw his utopia fall when the temptations of the commercial world reached the insular laboratory.

The Xerox Printer

For the first several years that Richard Stallman worked in the AI lab, the group used a printer controlled by the lab's custom operating system, ITS. The lab programmers

[17] Joseph Weizenbaum, "Computer Power and Human Reason: Science and the Compulsive Programmer,"

[18] Williams, 84.

[19] Ibid., 48, 95.

[20] Ibid., 76.

wrote software that managed print jobs and alerted the users when any issues occurred, but the printer was very slow. In 1980, the Xerox corporation gave the lab a second device, a laser printer. Richard Stallman recalled the printer was "very fast, printed a page a second, very fine in many respects, but it was unreliable, because it was really a high-speed office copier that had been modified into a printer." Like a copier, the printer frequently jammed. The challenge was that when copiers jammed, the person using it was there to fix the problem. Since the printer was networked, the device would jam, and it would be a long time before anyone would notice. The obvious programmatical solution was to modify the printer software so that whenever it encountered an issue, it could alert the users and prohibit additional print jobs from being sent to the device. The problem was that Xerox had no software for the MIT-specific ITS operating system. Despite their mastery of programming, the AI lab hackers were completely helpless in their quest to make the printer work the way they needed it to.[21]

Xerox Palo Alto Research Center scientist Robert Sproull had been the primary developer of the software for Xerox's laser printer. In 1980, Sproull took a faculty position at Carnegie Mellon University. He continued to work on the technology at his new appointment. Having learned that a Carnegie Mellon University researcher had a copy of the needed source code, Richard Stallman made a point of asking for it the next time he visited. According to Stallman, Sproull flatly refused the request. The researcher would not even discuss the device or its software, citing a contractual restriction that

[21] Richard Stallman, "Free Software: Freedom and Cooperation," (speech, New York, New York ,May 29, 2001), GNU.org, https://www.gnu.org/events/rms-nyu-2001-transcript.txt, (accessed July 17, 2018).

prohibited him from sharing the code or any knowledge about the device.[22] Richard Stallman initially interpreted the interaction as a personal slight, but he soon realized that non-disclosure agreements were an emerging corporate tool designed to restrict the dissemination of "proprietary" knowledge. Far from personal, Stallman realized that this was a new social phenomenon that would soon affect many people.[23]

The LISP Machine

The issues with the Xerox printer were a harbinger of the changes occurring in the software industry. When the corporatization of software finally emerged to the MIT AI lab, it did so through an unlikely source: the lab's LISP machine project. MIT AI lab hacker Richard Greenblatt was frustrated at the performance issues that he experienced running LISP programs on conventional hardware. The original implementations of LISP (Maclisp at MIT), were on Digital Equipment Corporation's PDP-10. As AI research advanced, the AI lab hackers found the PDP computers to be insufficient for the intense workloads required by AI programs. Rather than wait for commercial vendors to offer a powerful enough device, Greenblatt envisioned a new computer, with larger processors, additional memory, and a more complex operating system that was designed specifically for running LISP programs.[24] The device would not permit time-sharing; instead it would be a personal workstation with all system resources dedicated to running LISP code.[25]

---

[22] Williams, 8-9.

[23] Richard Stallman, "Free Software: Freedom and Cooperation."?

[24] Alan Bawden et al., *Lisp Machine Progress Report* (Cambridge, Massachusetts: Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1977), 2-3.

[25] The PDP-10 was designed with a processor that supported an 18-bit address space. As such, the compilers for the system were limited to the needs of the hardware. The LISP machine would be designed

By 1977, Greenblatt's vision was realized, and members of the AI lab team were constructing LISP workstations under a DARPA grant. The MIT group discovered that there was commercial interest in the computer, and by 1979, they had created 30 machines for DARPA and other organizations.[26] Because of the significant desire for the machines, Greenblatt decided to start a business focused on manufacturing the devices.[27]

In 1979, Greenblatt formed LISP Machines, Inc. (LMI) to construct and sell the computers. LMI was envisioned as a "hacker" company: Greenblatt intended that the organization be run solely by hackers and as such, he avoided external capitalization, which would afford control of the organization to outside investors. Instead, he envisioned a self-funding model. Greenblatt saw the first step as finding a customer willing to pay in advance. Using the funds from this initial customer, LMI would build and deliver the first batch of computers. The income from this sale would then be invested in purchasing parts for building future machines. The sale of these subsequent devices would produce revenue sufficient to provide a profit.[28] In Greenblatt's plan, LMI would hire the AI lab hackers as part-time staff, while allowing the programmers to continue part-time work at the AI lab. Maintaining their AI lab positions would act as insurance against the instability inherent to joining a start-up company. Despite

---

with a 24-bit virtual address space, and the operating system would emphasize efficiency to leave as much space for programs as possible.

[26] Levy, 441-444.

[27] Since the mid-1950s MIT had been permissive with the corporate appropriation of its technologies. This approach helped fuel the growth of Boston's route 128 as a technology corridor home to businesses like the Digital Equipment Corporation, Prime Computer, and Bolt, Beranek, and Newman Inc. These organizations were frequently founded by MIT alumni, and they proffered commercial versions of technologies developed at MIT.

[28] Richard Stallman, "My Lisp Experiences and the Development of Gnu Emacs," (speech, San Francisco California, October 28, 2002), GNU.org, https://www.gnu.org/gnu/rms-lisp.html, (accessed July 18, 2018).

enthusiasm for Greenblatt's planned venture, many of the lab hackers were unconvinced that the proposed approach would work. Realizing that he lacked business experience, Greenblatt sought the help of a former AI lab administrator, Russell Noftsker, who had left his post at the lab under duress several years prior and had since become successful in business. When Greenblatt proposed involving Noftsker, many of the AI lab hackers chaffed at the idea, but since Greenblatt had maintained a good relationship with the departed administrator, he felt strongly that Noftsker could help him establish LMI.[29]

Noftsker was enthusiastic with the prospect of a LISP machine, and he joined Greenblatt's effort. However, once involved, Noftsker made clear that he objected to Greenblatt's low-risk, low-capital plan, and he began to push for heavy market capitalization. When Greenblatt was unmoved by Noftsker's recommendations, the former lab administrator told the AI lab team members he felt that the "Hacker's Hacker" Greenblatt might be unable to manage the company. Noftsker pushed for professional management, which meant larger start-up costs that required market capitalization. This created a paralyzing tension within the group; Greenblatt was the father of the LISP Machine, but he lacked business acumen and resources. Noftsker offered a vision of a wealthy, successful company. Asserting that Greenblatt's LMI wasn't "winning" sufficiently by the end of its first year, Noftsker and several other AI lab programmers founded a competing company, Symbolics, in April 1980.[30]

In contrast to LMI, Symbolics was heavily capitalized. The organization lured away fourteen AI lab employees with generous full-time employment offers. The remaining AI lab hackers – except for Richard Stallman – worked for LMI part time.

---

[29] Levy, 44-45.
[30] Ibid., 46.

Although uninvolved, Stallman initially condoned the commercialization efforts of his former colleagues. Once there were two LISP companies, Stallman was kept busy keeping the AI lab's LISP machine updated with their innovations. The licensing agreements for the LISP machine technology entitled the university to use any innovations made by either company. Specifically, the contracts permitted MIT to review the source code produced by the licensees. However, it did not explicitly grant the Lab use of the software. As a result, to use the innovations, the AI Lab needed to re-write and locally integrate the features produced by the licensees. It fell to Stallman to write the programs needed to keep the lab's LISP machine up to date.[31]

Despite the doubts of Noftsker and his other former lab mates, Richard Greenblatt's LMI was successful. As planned, Greenblatt convinced the first customer to pay in advance, and with the resultant income LMI capitalized the construction of subsequent machines.[32] Greenblatt eventually relaxed his views on outside funding, signing an agreement with Texas Instruments. For a quarter of the LMI stock, Texas Instruments provided both money and support for the LISP machines.[33] Greenblatt's low-risk plan had succeeded in establishing LMI without capital, and the new deal with Texas Instruments promised to help the company grow.

LMI's success was unwelcome news to Noftsker and the professional executives at Symbolics. When it became clear that LMI was not failing, Symbolics' executives became determined to undermine the competing organization. First, Symbolics contacted MIT about enforcing a seldom-used university rule that disallowed part-time work. This

[31] Williams, 95-96.
[32] Stallman, "My Lisp Experiences and the Development of Gnu Emacs".
[33] Levy, 447.

forced the LMI employees out of the MIT lab. Since Symbolics had already hired away the rest of the lab hackers, Richard Stallman was left as the sole technical support employee.[34] With the AI lab now nearly empty, Symbolics began to dictate terms to MIT. On March 16, 1982, Symbolics issued an ultimatum. They demanded that the MIT AI lab only use an official Symbolics LISP machine, and they also insisted that MIT sever their relationship with Symbolics' rival, LMI.

"The way I saw it, the AI Lab was a neutral country," Stallman recalled of his perspective at the time. If the AI lab were to give into Symbolics' ultimatum, they would be choosing sides, favoring Symbolics at the expense of LMI. "We were not allowed to be neutral anymore," Stallman stated. "Up until that point, I hadn't taken the side of either company…, now, Symbolics had forced the issue." Upset at what Symbolics had done to the AI lab community, Stallman decided to oppose the aggressive company. His immediate retaliation was petty; he disconnected the microwave network link that connected Symbolics to the AI lab. Following this, he dedicated his energy to "duplicating all of the improvements Symbolics had made to the Lisp machine system. I wrote the equivalent improvements again myself (i.e., the code was my own)." Whenever Symbolics made an initial product announcement that included release notes, Stallman would document the new features and then write them for the AI lab LISP machine himself. "By the time they had a real release, I did too." Stallman reported.[35]

When the Symbolics executives found their new features appearing instantly on the AI lab LISP machine, they were furious. Convinced that Stallman was stealing their source code, they placed software on the programmer's computer so that they could spy

[34] Richard Stallman, "My Lisp Experiences and the Development of Gnu Emacs".
[35] Williams, 92-96.

on his activities.[36] But Stallman was writing the software anew. Although Stallman could review the Symbolics source code, he had concluded that it would be best not to even glance at their work.[37] Noftsker complained to MIT administration, but according to Stallman, the university supported him.[38] "We develop a program or an advancement to our operating system and make it work, and that may take three months, and then under our agreement with MIT, we give that to them." Noftsker bitterly recalled to journalist Steven Levy, "And then [Stallman] compares it with the old ones and looks at that and sees how it works and reimplements it." The actual programmers at Symbolics had a different perspective of Stallman's "master hack" of recreating the work of the entire Symbolics development team by himself. Legendary AI Lab hacker Bill Gosper recalled reacting with admiration, realizing, "Wait a minute – Stallman doesn't have anybody.... He's working alone! It's incredible anyone could do this alone!"[39]

In Stallman's view, the coming of these corporations had brought the destruction of the lab community. Where LMI's approach sought to keep ties to the MIT AI lab, Symbolics emptied it by hiring away the staff. Then, to stifle competition, Symbolics pursued the termination of the LMI affiliated staffers that remained. Now, Stallman was alone. When he called the lab, there was no one there to answer. The ritual Chinese food runs ended, as did the spirited technical discussions and weekend-long communal hacking sessions.[40] When Stallman encountered the Symbolics programmers outside of the lab, he felt shunned and ignored.[41] For Richard Stallman, this was nothing less than

[36] Ibid., 96.
[37] Stallman, "My Lisp Experiences and the Development of Gnu Emacs".
[38] Williams, 96.
[39] Levy, 449-450.
[40] Ibid., 448.
[41] Williams, 98.

the destruction of his home and his beloved loved hacker subculture.[42] "The abandonment of our lab was followed by "war" in our lab. The abandonment happened when Symbolics hired away all the hackers…the AI lab was now helpless." Stallman recalled of the situation, "Nobody had envisioned that the AI lab's hacker group would be wiped out, but it was."[43]

Richard Stallman's hacking crusade against Symbolics continued until the fall of 1983. Once he felt assured that LMI had a sporting chance against Symbolics, Stallman decided it was time to move on.[44] In Stallman's perspective, "the war that Symbolics started was what wiped out MIT." However, the decimated hacking community in the AI lab was a microcosm of the broader cultural changes occurring in computing. Everywhere, it seemed to Stallman, "people were giving up on cooperation, and together this wiped out the community and there wasn't much left."[45]

Richard Stallman saw the emerging corporate software market as culpable for the changes. Corporations such as Xerox now used the law and non-disclosure agreements to stifle the sharing of knowledge. Other corporations, like Symbolics, used software to spy on Stallman and employed bullying tactics to force their will on the AI lab. Stallman felt faced with what he considered to be a "stark moral choice."[46] He could either choose to overcome his objections to the actions of these companies and accept their proprietary software, or he had to develop another alternative.

---

[42] Ibid. 97.
[43] Stallman, "My Lisp Experiences and the Development of Gnu Emacs".
[44] LMI went bankrupt in 1987. Symbolics was sold several times before finally going defunct in 1996.
[45] Stallman, "My Lisp Experiences and the Development of Gnu Emacs."
[46] Williams, 101.

The GNU Project

Richard Stallman's hard-headed focus on individual freedoms had not changed since the political arguments of his youth. But where he had no agency in the Vietnam War, his expertise in software meant that he had power to effect change in this quagmire. Unable to resist the might of the spreading software industry, Stallman opted for a revolutionary approach. Instead of direct confrontation with software companies, Stallman would create his own software domain. It would be separate from the commercial software market and would guarantee that software users retained the freedoms that Stallman felt were being abrogated. In Stallman's new software utopia, consumers would be able to run any program for any purpose. More importantly, they would have the source code, so they would be able to adapt the software for their own needs and share their improvements with others if they desired. There would be no need for non-disclosure agreements or any other restrictions on discussing the technology. And because the users of the software had access to the source, they could review it to ensure that the programs were not being used for malicious purposes such as spying.

Richard Stallman chose UNIX as the basis for his new software platform. This decision made sense for several reasons. First, LISP, Stallman's current working environment was a niche application, designed for AI research. In contrast, UNIX was a popular general-purpose platform, one that already possessed a thriving user base accustomed to writing and sharing software. The UNIX community was also experiencing problems with the software industry. Following the 1982 anti-trust settlement, AT&T released System V UNIX and, with it, dramatically increased price.[47] UNIX had gone from inexpensive and open to costly and restricted. By creating a clone of UNIX, Stallman would have a ready-made community that already shared concerns similar to his own.

A New UNIX Implementation

Naming his effort, the GNU project, a recursive acronym, for "GNU's Not Unix," Stallman announced his intention via a message to several UNIX news groups on September 27th, 1983:[48]

> Free Unix!
>
> Starting this Thanksgiving I am going to write a complete Unix-compatible software system called GNU (for Gnu's Not Unix), and give it away free to everyone who can use it. Contributions of time, money, programs and equipment are greatly needed.
>
> To begin with, GNU will be a kernel plus all the utilities needed to write and run C programs…. We hope to supply, eventually, everything useful that normally comes with a Unix system, and anything else useful, including on-line and hardcopy documentation.
>
> GNU will be able to run Unix programs, but will not be identical to Unix. We will make all improvements that are convenient, based on our experience with other operating systems…
>
> Who Am I?
>
> I am Richard Stallman, inventor of the original much-imitated EMACS editor, now at the Artificial Intelligence Lab at MIT. I have worked extensively on compilers, editors, debuggers, command interpreters, the Incompatible Timesharing System and the Lisp Machine operating system. I pioneered

---

[47] "United States v. American Tel. and Tel. Co., 552 F. Supp. 131," JUSTIA US Law, February 28, 1983, https://law.justia.com/cases/federal/district-courts/FSupp/552/131/1525975/ (accessed July 19 2018).

[48] Richard Stallman. "New Unix implementation." Email to net.unix-wizards,net.usoft mailing lists. September 27, 1983.

terminal-independent display support in ITS. In addition I have implemented one crashproof file system and two window systems for Lisp machines.

Why I Must Write GNU

I consider that the golden rule requires that if I like a program I must share it with other people who like it. I cannot in good conscience sign a nondisclosure agreement or a software license agreement.

So that I can continue to use computers without violating my principles, I have decided to put together a sufficient body of Free Software so that I will be able to get along without any software that is not free.

How You Can Contribute

I am asking computer manufacturers for donations of machines and money. I'm asking individuals for donations of programs and work.

One computer manufacturer has already offered to provide a machine. But we could use more. One consequence you can expect if you donate machines is that GNU will run on them at an early date. The machine had better be able to operate in a residential area, and not require sophisticated cooling or power.

Individual programmers can contribute by writing a compatible duplicate of some Unix utility and giving it to me. For most projects, such part-time distributed work would be very hard to coordinate; the independently-written parts would not work together. But for the particular task of replacing Unix, this problem is absent. Most interface specifications are fixed by Unix compatibility. If each contribution works with the rest of Unix, it will probably work with the rest of GNU.

If I get donations of money, I may be able to hire a few people full or part time. The salary won't be high, but I'm looking for people for whom knowing they are helping humanity is as important as money. I view this as a way of enabling dedicated people to devote their full energies to working on GNU by sparing them the need to make a living in another way.

For more information, contact me.
Arpanet mail:
RMS@MIT-MC.ARPA

Stallman's announcement was as interesting for what it did not include as much as what it did. Stallman's opening statement emphasized that he was creating a "Free Unix." In the first paragraph and throughout the message, Stallman used the word "free" but never clarified the sense in which he meant it. The opening text seemed to be referring to free as meaning gratis – free of cost – but later in the document, his use of the word seemed more oriented toward individual freedoms, such as in the statement, "I have decided to put together a sufficient body of Free Software so that I will be able to get along without any software that is not free."[49] The lack of clarity and repeated use of this homonym gave an unclear impression of his intent.

Starting in his second paragraph, Stallman provided technical details of his plan, sharing the intended project order. He ultimately intended to write a kernel – the core of an operating system, but in order to do this, other programs needed to be written first. This included an editor to write source code in and a compiler to convert the source code to binary machine code. They also needed to create a command line-based user environment, the shell, where these programs would work. From there, the project would expand to less essential applications – text formatting, games, spreadsheets, and so on. In the third paragraph Stallman continued with the technical details by explaining the improvements that he intended to the current functionality available in UNIX. In the process, Stallman demonstrates how GNU is not UNIX, but is intended instead to expand beyond it.

[49] Ibid.

Stallman then established his bona fides, introducing himself as the inventor of the “original much-imitated EMACS editor.”[50] His GNU project resembles what he had done with EMACS in that it would be distributed free of cost with the source code available for modification. In citing the imitations of EMACS, Stallman indirectly references other versions such as the version AI lab hacker Dan Weinreb had written for the LISP machine. These were all products of the open software culture that encouraged the sharing and modification of source code.

In his “Why I Must Write GNU” section, Stallman explained that sharing software is key, writing, “I consider that the golden rule requires that if I like a program I must share it with other people who like it.”[51] He mentioned software licenses and nondisclosure agreements, but he never explains why he felt these were problematic. He does not expound on the events that led to the project such as the role of software licensing in Symbolics destructive attacks on the MIT AI lab. Stallman simply stated that he cannot work with the legal instruments as a matter of conscience. Instead, to avoid violating his principles – precepts that he does not expound upon, Stallman intended to write “Free Software” so that he will be able to avoid using “any software that is not free.”[52]

Richard Stallman closed his announcement by soliciting assistance in the form of money, computers, or labor. In the process, he explained his project’s technical schema of distributed development, which was easily achievable on UNIX due to its inherent modularity. Because UNIX software is designed to be simple, stand-alone programs, the

---

[50] Ibid.
[51] Ibid.
[52] Ibid.

software can be easily developed in a distributed fashion. Contributors can each work on one program, and provided that it is compatible with UNIX, it will work with GNU.[53]

The UNIX Community Responds

In 1983, when Richard Stallman posted his announcement, the commercial Internet did not yet exist. However, an embryonic form was in place consisting of a series of international networks connected to the U.S. government-run ARPAnet.[54] One of the primary forms of interactions on the network at that time occurred within news groups, such as the ones that Richard Stallman posted the GNU announcement to, net.unix-wizards and net.usoft. These newsgroups were a vital discussion forum for UNIX support, and membership covered the gamut of the UNIX community, including the creators of the operating system, Ken Thompson and Dennis Ritchie. As such, the newsgroup discussions provide unique insight into the UNIX community response to Stallman's plan.

The newsgroup messages reveal mixed reactions to the announcement. AT&T's recent changes with UNIX licensing set the tone of the conversation about Stallman's project. "The commercialization of Unix is sure to cause only trouble for those of us in the research community" commented Rick Spanbauer, a researcher from the State University of New York at Stony Brook. Spanbauer postulated that further restrictions on source code would soon come under the guise of it being proprietary material. "My personal feeling is that the commonly accepted principal of free flow of scientific

[53] Ibid.

[54] The networks at the time frequently used Unix-to-Unix-Copy (UUCP) rather than direct network connections. UUCP was an indirect, dial-up based approach that would use gateways to transfer messages to networks like the ARPAnet.

knowledge should apply in the case of programs." Spanbauer argued, "We will all benefit in the long run if projects can be accomplished by rewriting, modifying, or cannibalizing [SIC] existing code."[55]

Another commenter dismissed the proposition as redolent of communism, saying, "i [SIC] guess capitalism has finally become a dirty word… if an author wish [SIC] to place his work in the public domain, thats [SIC] great! but many have bills to pay and children to care for therefore employers are **very** important. ask the steel workers who have been out of work for years."[56]

University of Waterloo's Brad Templeton voiced concerns that GNU would stifle commercial competition by undercutting costs.[57] "I make most of my money from a set of programmer's utilities for Commodore machines." Templeton explained, "There is a similar product in the public domain. It is not as good, and arrived after, but it hurts my income a lot. So much so that I have seriously considered abandoning the whole market and doing something else."[58] Many newsgroup members countered these concerns. User Michael Turner postulated that GNU would motivate the industry to improve in that "the only way to undersell public domain is to do a much better job."[59] James Bray from Cambridge technology company Bolt, Beranek and Newman (BBN) agreed. "I don't think it is correct to surmise that GNU-like software would put software people out of

---

[55] Rick Spanbauer. "Whats GNU with you?" Email to net.unix-wizards,net.usoft mailing lists. October 5, 1983.

[56] Hart@cp1.UUCP. "Whats GNU with you?" Email to net.unix-wizards,net.usoft mailing lists October 11, 1983.

[57] The software Brad Templeton refers to is a BASIC programming utility named POWER. Despite his reservations, Templeton stayed in the computing industry and was the chairman of the board for the Electronic Frontier Foundation from 2000 to 2010.

[58] Brad Templeton. "GNU considered harmful to software quality." Email to net.unix-wizards,net.usoft mailing lists. October 7, 1983.

[59] Ibid.

work. In fact, I think the opposite is true." He argued, "One of the more frequent complaints [about UNIX] is that the often inadequate or entirely lacking support leads to many sites having their own resident "Unix-Person" to hack and babysit the thing."[60]

This perspective was echoed by user David Martindale. He postulated that having a good operating system on inexpensive hardware would foster more software development. As a result, where GNU software could be bad for competing producers, it would be "good for software 'consumers.'"[61] Martindale's comment gets to the focus of the GNU project, the software consumer. Where Richard Stallman certainly had animus toward corporations, his intention was not to undercut them, but to provide an alternative to them. The GNU project simply introduced a new software ecosystem that was isolated from the vicissitudes of the corporate software market. In this space, free from corporate intervention, he could recreate the open sharing culture that had become endangered by corporate overreach. The idea of sharing software was not new to computer programming; however, Stallman was the first to introduce a project deliberately designed to encourage others to consciously choose sharing over commercialization.

The GNU Manifesto

Following Richard Stallman's announcement, interest in the GNU project grew over the next year and a half. In newsgroups such as net.unix-wizards and net.usoft, the conversations were largely technical in nature as Stallman and a growing army of volunteers made progress on the effort. They completed compilers for C and Pascal, an

[60] James Bray. "Gnu and you." Email to net.unix-wizards,net.usoft mailing lists. October 18, 1983.
[61] David Martindale. "GNU considered harmful to software quality." Email to net.unix-wizards,net.usoft mailing lists. October 8, 1983.

Emacs-like text editor, and over 35 other utilities.[62] In March 1985, Stallman introduced the project to the broader computing public through the publication of the "GNU Manifesto," in *Dr. Dobb's* magazine.

Richard Stallman's manifesto borrowed heavily from his previous email announcement but provided more detail. Published in *Dr. Dobb's* "Realizable Fantasies" column, the article introduced the project: "GNU, which stands for Gnu's Not Unix, is the name for the complete Unix-compatible software system that I am writing so that I can give it away free to everyone who can use it."[63] From this introduction and throughout the article, Stallman continued his habit of using the word "free," without being clear on what he means – gratis or libre. Following his general introduction, Stallman reported on the programs that had been completed, and then he moved on to detail the technical plans for the platform and how it differed from UNIX.

Stallman included the "Why I Must Write GNU" section from the original GNU announcement, and, although he provided additional detail, he still neglected to completely explain his motivations for the project. Where he mentioned his tenure at the MIT AI lab, he does not detail the issues that occurred there. Instead, he focused his criticism broadly against software corporations, stating, "Software sellers want to divide the users and conquer them, making each user agree not to share with others."[64]

In the next section, Stallman provided detail as to why he chose UNIX. Put simply, the UNIX operating system had good features and could be easily adopted. He

---

[62] Richard Stallman, "The Gnu Manifesto," *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985, 30.
[63] Ibid.
[64] Ibid.

also discussed the copyright status of the project – a key addition from the earlier announcement:[65]

> GNU is not in the public domain. Everyone will be permitted to modify and redistribute GNU, but no distributor will be allowed to restrict its further redistribution. That is to say, proprietary modifications will not be allowed. I want to make sure that all versions of GNU remain free.

This paragraph reflected Richard Stallman's realization that to keep his software from being usurped by corporations, he required protections like what proprietary developers placed within software licenses.[66] Here, Stallman's use of the word "free" is most clearly intended to mean the ability to modify and share the software – the essence of Stallman's intentions. Despite discussing these protections, Stallman did not elaborate on how he planned to enforce them.

Following this, Stallman discussed the many other programmers who had volunteered to help with his effort. He considered their motivations, reporting that "many programmers are un-happy about the commercialization of system software. It may enable them to make more money, but it requires that they feel like competitors with other programmers rather than like comrades."[67] This examination also helps clarify the type of freedoms that Stallman intends for the users of software, but again his word choice is questionable. Having named the document "The GNU Manifesto," and using a loaded term like "comrades," while discussing a communal software project, Stallman invited comparisons to Marx's *Communist Manifesto* and the Soviet Union.[68]

---

[65] Ibid.

[66] The background behind this epiphany, and a more detailed discussion of these protections, can be found in the next chapter.

[67] Stallman, "The Gnu Manifesto," 30.

[68] It is possible that Stallman a deliberately used phrases redolent of communism as a tongue-in-cheek provocation toward critics who were determined to distill the ideas down to a Manichean world of capitalism vs. communism.

As in the original announcement, Stallman included a section on "How You Can Contribute," again repeating the project schema of distributed development by programmers. Where most of the planned software could be written in this way, Stallman noted that the operating system kernel will require a "small, tight group" to develop it due to the need for greater communication.[69]

The second half of the manifesto was an argument for the project. Stallman discussed how all computer users would benefit from the effort, as source code would be available to all for changes and sharing. "Users," wrote Stallman, "will no longer be at the mercy of one programmer or company that owns the sources and is in sole position to make changes."[70] He went on to explain how having the source code helped academic institutions by providing students additional material to study. Stallman then referred to the computer lab policy at his alma mater, Harvard, which required that any program installed on the system have its source code available.

Stallman then provided a host of "easily rebutted objections to GNU's goals," including the idea that no one would use the software because it lacked support, or that corporations needed proprietary software in order to gain a competitive edge.[71] In the process of defending the project, Stallman introduced potential business plans that would allow programmers (or corporations) to make money with GNU software. His business ideas included bundling GNU software with hardware, services such as teaching and maintenance, and programming cooperatives. After presenting multiple business models using GNU software, Stallman anticipated the criticism of pro-corporate readers. "Why is

---

[69] Ibid. 31.
[70] Ibid.
[71] Ibid. 31-33.

it," Richard Stallman pondered, "that free market advocates don't want to let the free market decide?"[72]

Stallman also speculated that future software development could be funded through a tax on computer sales. The funds received could be managed by a group, such as the National Science Foundation (NSF), that would allocate the resources to groups creating useful software. The regimen that Stallman describes is not all that different than what he had observed at MIT. Grant money provided by the U.S. government was administered by groups such as the National Science Foundation and DARPA. Funds allocated by these organizations had paid for the development of the technologies Stallman worked with in the AI lab. The creation of the LISP machine, for example, was subsidized by DARPA before corporations like Symbolics sought to commercialize it. And while funded by the government, Stallman's beloved AI lab had been harmonious. Stallman's experience suggested that it was the introduction of corporations that led to the destruction of the lab. In Stallman's proposal the software developed by the government grants would be made available to the public directly, rather than offered to corporations for commercial exploitation.

Within his itemized arguments for the GNU project, Richard Stallman cited ethical concerns with proprietary software. He reasoned that "extracting money from users of a program by restricting their use of it is destructive" in that the practice diminishes the way the software can be used. According to Stallman, such an approach was designed to enrich the software manufacturers at the expense of the consumers. "The reason a good citizen does not use such destructive means to become wealthier is

[72] Ibid. 32.

because, if everyone did so, we would all become poorer from mutual destructiveness." As such, Stallman perceived the software manufacturers as forcing consumers to violate Kantian ethics.[73]

According to Kant, the only inherently good thing was good will, and good will was only achievable when an individual dutifully acted consistent with moral law. Moral actions were ones that adhered to Kant's categorical imperatives. Kant's imperatives included the requirement that to act ethically, an individual must "act only according to that maxim whereby you can at the same time will that it should become a universal law without contradiction"[74] Any action done would only be moral if the action was acceptable if it was universally applied. Kant's moral rules also required that individuals needed to treat humanity, "never merely as a means to an end but always at the same time as an end."[75] Any action that manipulated individuals to achieve a goal was immoral.

Stallman saw sharing information to assist others as a moral duty consistent with Kant's moral laws. Software manufacturers curtailed good will by prohibiting consumers from sharing information and programmatic solutions with each other.[76] This included modifications to existing code that could benefit the whole community by providing solutions to common problems. He felt the obligation to assist fellow community members to be able to understand how software works. In order to meet these obligations, the software consumers required access to the source code and the ability to discuss the software openly. In Stallman's view, the software licenses and non-disclosure agreements

---

[73] Ibid., 33..

[74] Immanuel Kant, *Ethical Philosophy: The Complete Texts of Grounding for the Metaphysics of Morals, and Metaphysical Principles of Virtue,* 2nd ed. (Indianapolis, Illinois: Hackett Pub. Co., 1994), 32.

[75] Ibid., 36.

[76] Ibid.

prohibited him from being able to perform his moral duties. These legal instruments also infringed upon Kant's practical imperative, which held that humanity and the individual should be treated as an end, never as merely the means to an objective. By ignoring the rights of consumers and treating them solely as a means to achieve corporate profit, the organizations violated Kantian ethics. For Stallman, communication, and the sharing of information with other humans was an ethical imperative, and as such, he viewed the new restrictions placed on software and consumers via non-disclosure agreements as greedy "information hoarding," and unethical.[77]

Richard Stallman's manifesto ended with the argument that "making programs free is a step toward the post-scarcity world." Stallman stated that bureaucracy and "isometric struggles against competition," worked against technology's ability to reduce the individual's workload.[78] Sharing software allowed for code reuse. By not having to constantly re-invent the wheel by writing programs that others have already created, programmers were freed up to focus on new innovation.

Readers Respond to the GNU Manifesto

Readers responded to Richard Stallman's GNU manifesto. One reader from Tarzana, California fully supported the proposal. He reported not being a fan of UNIX but was enthusiastic about the effort, saying "The computer industry needs to be turned upside down every so often. Mr. Stallman seems just the fellow to do it this time." The letter hailed Stallman "as a true adherent to the Hacker ethic," situating Stallman's manifesto relative to Stephen Levy's book, *Hackers*, which had been published the

[77] Stallman, "The GNU Manifesto."
[78] Ibid. 34.

previous year.[79] Another commenter from Goleta, California was even more excited. "Hurrah! Three cheers for GNU! Your proposal and the philosophy behind it are so refreshing I almost couldn't believe what I was reading…." He went on to share that he and friends had been discussing ideas similar to Stallman's project as a solution to the "royalty problems that make Unix (even on Micros) so ridiculously expensive."[80] Not surprisingly, this commenter also offered to join the GNU project effort.

Other readers seemed unable to fully understand Stallman's proposal. "I hope he succeeds with GNU…because for all the reasons he mentions, I could be a beneficiary," wrote a subscriber from Southfield, Michigan. However, he also expressed concerns that as a free-market advocate, he saw it as destructive to society. "If Lotus Development gave away 'Lotus 1-2-3' instead of charging for it, we could all use it and 'society' would be richer. The problem is that Lotus Development would never have created 'Lotus 1-2-3'in the first place." This concern was moot in this scenario as Stallman's project was already well underway. "On second thought, there is nothing to worry about," he wrote to end his letter, "for if there is a need for a better operating system than GNU in the future, people will be willing to pay for it."[81] A fellow reader from Toronto, echoed similar economic concerns. He stated that he was not "opposed to the idea of a public domain version of Unix;" instead, he saw the issue as being that Richard Stallman was self-righteous. In the text of his letter, the reader argued that Stallman "forgets that he is living in a capitalist society…. Everybody needs food, even more than Free Software. So food

[79] James Glass. Letter to the Editor, *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, June1985. 8.

[80] Rollin V. Weeks. Letter to the Editor, *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, June 1985. 10.

[81] Jim Harlan. Letter to the Editor, *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, June 1985. 11.

would be distributed for free, right?"[82] While he endeavored to stretch the GNU project into state communism, his letter also demonstrated that he did not understand what Stallman wrote. The reader's missive focused heavily on the public domain, but Stallman had clearly stated in his manifesto that GNU was not for the public domain.[83]

The previous responses were reserved, however, when compared to the letter sent by Robert Schwartz of Wakefield, Massachusetts. Schwartz's response was a confused, angry screed. "The GNU Manifesto in your March issue made me furious," he wrote. "…I'll confine myself here to 2 major points." First: "Stallman doesn't believe in property rights." He supported this assertion with the fact that nowhere in his article did Stallman expressly write that "Unix is a trademark of Bell Labs." This omission, Schwartz decided, was because Stallman "doesn't recognize their claim to the name Unix." This was important, Schwartz argued, because "if it hadn't been all the time and money invested by Bell Labs" in inventing UNIX, the operating system would have no value – and if Unix had no value, then neither would GNU. Schwartz's comments suggest that he was ignorant to the origins of UNIX, and of the herculean efforts required by the UNIX developers to get their unsanctioned project recognized and funded.[84] Schwartz's second point was more cogent. "Stallman's explicit philosophy is socialist redistributionism," citing Stallman's software tax speculation. "Stallman wants to tax *me* to fund *him* to develop software he will give away for *free*." Schwartz complained, "Is this how you want your tax money spent?" Schwartz was apparently unaware that

[82] David Kettle, Letter to the Editor, *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, June 1985. 10.

[83] Stallman, "The Gnu Manifesto," 30.

[84] Curiously, Schwartz suggests that his point can be proven if one were to "reread Stallman's article, substituting 'Glorp' for 'Unix.'" It is unclear how this substantiates Schwartz's perspective.

government agencies such as DARPA used taxpayer money to fund the development of technologies that corporations later commoditized and sold. The colorful ending of Schwartz's letter summarized his perspective well. He stated that Stallman prefers the "Marxist ideals of the Soviet Union," where Schwartz preferred the "good old capitalist U. S. of A." He closed with, "Stallman says he is looking for contributions. I would be delighted to offer a one-way plane ticket to Moscow."[85]

The Business World Responds to Stallman

Dr. Dobb's Journal was considered to be "very much a 'hacker's' magazine."[86] In addition, any discussion of "Free Software" would be naturally welcome at the periodical, which itself had been originally formed to disseminate Tiny Basic. The reader responses in *Dr. Dobb's* are revealing, but they do not provide a complete picture of the reactions to Stallman's project.

In December 1984, *ComputerWorld*, a periodical focused on computing in business, had published an article about Richard Stallman and his new project. After the author, Maura McEnaney, observed Stallman's participation in a panel at the Wang Institute of Graduate Studies in Tyngsborough, Massachusetts, she interviewed Stallman and reported back on his project. In the article, McEnaney briefly highlights the GNU project and his personal goals, before moving on to some of Stallman's provocative opinions. "All software should be free," Stallman postulated, "Because it is improved upon only when it is built upon." Stallman further embellished his goals; "I wish to promote the sharing of software and interfere as much as I can with other people's

---

[85] Robert Schwartz. Letter to the Editor, *Dr. Dobb's Journal of Tiny BASIC Calisthenics and Orthodontia*, June 1985, 8-9.

[86] Thomas Spence, "The Hacker's Voice...," *Whole Earth Software Catalog*, 1984.

attempts to interfere with the sharing of software." Stallman explained that copyright laws can have a deteriorating effect on the quality of software, as it restricts the program's evolution. He compared the restrictions to the extortion of arsonists who "threaten to destroy wealth if people don't pay."[87]

Stallman's perspective was met with a fiery response. Reader Thomas Murphy of Roseville, Michigan wrote a letter to protest the article. Characterizing it as a call to abolish copyrights, Murphy attacked Stallman's proposition as an "act of moral cowardice" that ignores the first causes of innovation, which Murphy suggested being individuals "pursuing their own goals and interests." He contrasts this to Stallman, who he claims is motivated by personal vices of "dependence and parasitism." Murphy dismisses Stallman's perspective as "misconceptions one would expect to see editorialized in some third-rate Marxist tabloid." The author suggested that the shame Stallman reported feeling when using proprietary software was in fact hatred: hatred for the pride that a creator possess when making a product, hatred for the creative mind. "There is no more evil doctrine," Murphy says of Stallman's perspective, before going on to imply that Stallman's support of Free Software is a felony.[88] Two months later, in the February 1985 issue of *ComputerWorld*, another reader, Michael H. Agranoff, submitted a letter supporting Murphy's vitriolic attack on Stallman. In his letter, Agranoff complained of having heard Stallman speak at a seminar on computer ethics. He mocked Stallman, who "had the gall to say that he expected reputable firms to hire him as a consultant." After writing Stallman off as an indeterminable threat, "a vintage flower

[87] Maura McEnaney, "All Software Should Be Free, Software Developer Maintains," *ComputerWorld*, December 24, 1984, 38.

[88] Thomas Murphy, "Case Against Free Software," *ComputerWorld*, January 21,1985, 39-40.

child," Agranoff complained that the seminar program had listed Stallman as associated with the MIT Artificial Intelligence Lab, giving the impression that Stallman was employed there, and Agranoff was upset when he (erroneously) concluded that Stallman was not affiliated with the lab.[89]

Neither editorialist appears to have done a close reading of the article about which they were complaining. For example, despite Agranoff's aspersions about Stallman being unsuitable as a consultant and not being employed by the AI lab, the article made clear that Stallman was "a former affiliate of the Artificial Intelligence Laboratory at MIT," and that he currently worked as professional consultant for Lisp Machine, Inc.[90] These editorials were not reasoned, fact-based refutations of Stallman's philosophies, but rather visceral responses from members of industry.

To be fair to the readers of both *Dr. Dobb's* and *ComputerWorld*, Stallman's provocative choices in wording and content invited scrutiny. The purely ideological rejections of Stallman's proposal are not surprising considering the political context. At that time, the ideological conflict between the United States and the U.S.S.R dominated geopolitical events. In the two years since the GNU project had begun, the Soviets shot down a South Korean airliner, and Russian spy operations had been recently dismantled in several western countries. In the United States, the television film, *The Day After,* stoked fears of a nuclear apocalypse mere weeks after the U.S. invaded the Marxist nation of Grenada.

[89] Michael H. Agranoff, "Free Software Philosophy Refuted," *ComputerWorld*, March 4, 1985, 40.

[90] Maura McEnaney, "All Software Should Be Free, Software Developer Maintains," *ComputerWorld*, December 24, 1984, 31, 38.

Where the ideological opposition to Stallman's proposal was understandable, the objections invoking the free market were less so. Stallman's proposal was the rhetorical equivalent of a software license agreement. Just as with a proprietary corporate license, Stallman described the restrictions placed on his programs – except instead of constraining the use of the software, Stallman's limitations were designed to preserve access to it. And just as with corporate software, Stallman set a price for his product – it just happened that the cost was $0. In a free market, competition is healthy, and the market dictates cost and determines which product thrives. Despite the economic concerns voiced by critics, Stallman's GNU software was just another competitor in the free market.

Summary: The Advent of Free Software

By the early 1980s the corporate drive to commercialize software caused a series of cultural shifts within the computing industry. When software commercialization came to MIT, it had a pronounced effect on both the AI lab and employee Richard Stallman. Dueling corporations decimated the lab by hiring away staff members. Richard Stallman mourned the loss of the hacker fellowship he had previously experienced in the lab, and he felt targeted by the software companies. In the process, he learned to mistrust corporations. Xerox's embargo on discussing the source code for the MIT AI labs laser printer showed Stallman that corporations could use contracts to abrogate the consumer's freedom of speech. The AI lab's well intended efforts to build a business around the LISP machine led to internecine conflict when the Symbolics corporation acted against their competitor, LMI and the MIT AI lab.

Richard Stallman was personally targeted by the Symbolics corporation which installed software to spy on his activities. This reified his concerns over corporate misconduct and the use of any software that prevented examination of its source code. Because binary programs could not be read, the software could easily be made to spy on consumers or share their information. Having source code accessible meant that the consumer could review the program's functions and be reassured that the software was not being used maliciously.

Richard Stallman's solution to his "stark moral choice," was to create his own, separate software ecosystem. His schema for this new type of software was a community-based model of distributed production and dissemination. Stallman's software was libre, free as in free speech, and his software consumers were encouraged to use the source code as they saw fit. The software also happened to be gratis, free as in cost, but this was a side effect. Stallman's answer for his moral conundrum was to create a computing environment where he could act in accordance with his own ethical beliefs. Like a monastic recluse, Stallman cloistered himself in a space free from the moral complications of the outside world. Stallman was no stylite however; he invited all to join him in using Free Software.

CHAPTER 4: FROM PROJECT TO MOVEMENT

By the fall of 1983, Richard Stallman had developed an extreme distrust for software corporations due to the actions of a predatory company, Symbolics, which had targeted the AI lab and spied on Stallman himself. In the wake of these actions, Stallman faced a dilemma: he could overcome his distaste for the self-serving practices exhibited by software companies and work within the software industry, or he needed to develop an alternative. His solution was to preserve the traditional practice of sharing programs by creating his own software, the GNU project. Stallman's programs would be free: free to use, free to modify, free to share. Working extensively with programmers in on-line news groups, Stallman set out to create his brave GNU world of software.

Almost immediately he once again ran afoul of a software corporation after a fellow programmer, James Gosling, appropriated Stallman's Emacs software and re-wrote it on the UNIX operating system. Gosling initially encouraged others to contribute to this UNIX version of Emacs, but he later claimed exclusive copyright and sold the software to the UniPress corporation. UniPress then began to pressure Richard Stallman and other developers over the re-use of code they had previously shared with Gosling. Despite his best efforts to create a world apart from corporate control, Richard Stallman found himself again beleaguered by commercial interests. The situation showed that just producing Free Software was not enough. Both Stallman and the project needed to take actions to proactively protect themselves.

Richard Stallman's response was to leverage the same legal instruments used by his corporate assailants: incorporation and software licensing. Stallman formed the Free Software Foundation, and in the process, developed a mechanism to support his programming efforts and extend his Free Software message beyond news groups. He also created a series of software licenses, legal instruments designed to protect his programing efforts from being appropriated by opportunistic developers or corporate interests. In the process, he fashioned the mechanisms by which Free Software could grow beyond the GNU project and into a movement.

Emacs

In 1974, Stallman had written the Emacs (Editor MACroS) text editor. The program was intended to address the shortcomings of the TECO text editor then used by the AI lab on their Incompatible Time-Sharing System (ITS) operating system. The software organized programming macros written by others into a common interface and it soon became the standard editor on the ITS system.[1] Consistent with the open sharing common in the culture at the time, Stallman distributed the software "on a basis of communal sharing, which means that all improvements must be given back to me to be incorporated and distributed."[2] The software was popular and other programmers ported it to various operating systems. In 1978, AI lab programmer Dan Weinreb created EINE (Eine is Not EMACS), an EMACS version for the MIT Lisp Machine. That same year, Bernie Greenberg ported EMACS to Multics via MacLisp, the project MAC version of the Lisp programming language.[3]

Stallman's original schema with Emacs placed himself at the center of development. Others could contribute changes, but they needed to provide them directly

[1] Williams, 85.
[2] Ibid.
[3] Jamie Zawinski, "Emacs Timeline," JWZ.org, March 8, 1999, https://www.jwz.org/doc/emacs-timeline.html, (accessed July 15, 2019).

to him. Stallman believed that by maintaining centralized control over the project, he would be able to curtail any behavior that he deemed too restrictive.[4] The actions of a graduate student named James Gosling showed Richard Stallman that this approach was not enough to protect his software.

Gosling Emacs

In 1980, James Gosling was a graduate student at Carnegie Mellon University (CMU).[5] Gosling, a fan of Stallman's EMACS software, decided to create a port of it for the UNIX operating system. In the manual he produced for his version of the editor, Gosling provided a brief history of Emacs, which was, "first was written at MIT…the spiritual father of all the EMACS-like editors; its principal author was Richard Stallman." Gosling claimed that his version was also called EMACS with the expectation that it would encourage himself and other contributors "to bring it up to the standards of what has come before."[6] The others that Gosling referred to – community programmers – accommodated the author's request and began to contribute to Gosling's UNIX Emacs project. Programmer Fen Labalme recalled his involvement with the project. "I (and others) made many modifications to EMACS along the way. Most of these got sent back to JAG@CMU [Gosling] for incorporation into future releases."[7]

Contributors to the UNIX Emacs project became alarmed when notices attributing sole copyright ownership to James Gosling began to appear within the program's source

---

[4] Williams, 85.

[5] Gosling's dissertation director at Carnegie Mellon University was Dr. Robert Sproull, the researcher formerly of Xerox PARC who had rebuffed Richard Stallman's efforts to obtain printer source code for the MIT AI lab.

[6] James Gosling, *Unix Emacs Manual* (Pittsburg, Pennsylvania: James Gosling, 1981), 3.

[7] Fen Labalme. "Whither GNU EMACS?". Email to net.emacs mailing list. June 26, 1985.

code.[8] The programmers approached Gosling, who dismissed their concerns, explaining that the copyright notices were there simply to prevent others from appropriating the software that they were working on.[9] Because he worked directly with a number of sites that used the UNIX Emacs software, contributor Fen Labalme explicitly asked Gosling about sharing Labalme's contributions with others. "I asked Jim if I could distribute my modifications directly. He granted me such permission in an informal (email) note," Labalme claims, "This note said that I could make releases, provided that I provided to each site a complete copy of his original code. My changes were to be released separately."[10] The programmer complied, and when he released tapes, they contained both Labalme's contributions as well as the original source code for Goslings version of Emacs.[11]

On April 9, 1983, James Gosling emailed the net.emacs news group and announced that he had sold the UNIX project on which he and the other programmers had been working on.[12]

> The version of Emacs that I wrote is now available commercially through a company called UniPress.... They will be doing development, maintenance and will be producing a real manual. Emacs will be available on many machines…. Along with this, I regret to say that I will no longer be distributing it.

Despite James Gosling's reassurances to the programmers contributing to the UNIX Emacs project, the copyright notices were not simple protections for the software. Instead, they were Gosling's preparations for the sale of the program. The UNIX emacs

[8] Richard Stallman. "Whither GNU EMACS?" Email to net.emacs mailing list. June 26, 1985.
[9] Ibid.
[10] Fen Labalme. "Whither GNU EMACS?". Email to net.emacs mailing list. June 26, 1985.
[11] Ibid.
[12] James Gosling. "Emacs goes commercial". Email to net.emacs mailing list. April 9, 1983.

software, which had been previously freely distributed with the source code was now restricted. Users interested in the software were now expected to pay $400 per CPU.[13]

Gosling justified the sale as an action taken for the good of the user community. “I can't continue to look after Emacs since I recently ceased to be a grad student and have to get on with the rest of my life.” Gosling explained that the “volunteers from the community of university hackers wouldn't work,” as a “long term, full time commitment is needed to keep a system like Emacs alive,” – a commitment that UniPress would provide.[14] “The alternative of abandoning it to the public domain is unacceptable,” Gosling declared, “Too many other programs have been destroyed that way.”[15]

James Gosling’s justification for the sale seemed to overlook that his program was based on what was essentially public domain software – Stallman’s Emacs. Gosling also ignored the inconvenient fact that the software he sold as his own had been developed with contributions from community volunteers. This omission was not missed on Stallman, who posited, “Is Gosling going to pay out a share of the royalties to all of you who contributed extensions when you were under the misapprehension that you were working for the common good?”[16] Gosling’s retort to Stallman side-stepped the issue of payment in favor of denigrating the quality of the submissions he had received.[17] “When a program evolves by having a lot of different people put in their own ‘favorite hacks’ or

---

[13] Jeff Fluke. "emacs from Unipress". Email to net.emacs mailing list. April 20, 1983.

[14] James Gosling. "Emacs goes commercial". Email to net.emacs mailing list.. April 9, 1983.

[15] Ibid.

[16] Richard Stallman. "Software Sabotage". Email to net.emacs mailing list. April 17, 1983.

[17] After graduating from Carnegie Mellon University, James Gosling went on to work for Sun Microsystems where he was the father of the Java programming language – a promising platform that became notorious for quality issues, particularly around memory management.

‘quick patches,’" Gosling wrote, “it ceases being coherent and dissolves into a bag of used band-aids.”[18]

GNU Emacs

By 1985, UniPress’s Emacs was no longer the sole option for the UNIX operating system. AT&T used a version internally that had been written by researcher Warren Montgomery. However, this software was not publicly available.[19] Steven Zimmerman of the Computer Corporation of America (CCA) also developed a UNIX Emacs that CCA offered for $850 per processor ($350 for educational institutions).[20] By this time, Stallman had also released a new UNIX version of his Emacs software under the GNU project. The GNU emacs version was available via ARPAnet and UNIX sites and it was met with very positive reviews. Harvard’s Marty Sasaki reported that “GNU EMACS runs very well (comparable speed to CCA and Gosling's) … there won't be much reason to buy CCA or UNIPRESS EMACS.”[21]

The potential competition from Stallman’s version was not lost on UniPress. Fred Pack, a UniPress representative, began to raise concerns about the GNU version of the software. His complaints arose from the GNU version’s use of display code written by Fen Labalme, who had previously contributed his display software to Gosling’s UNIX Emacs project prior to Gosling’s sale of the program. UniPress’s perspective was that, despite Labalme being the original author of the software, Gosling’s sale of Emacs meant that Labalme could not legally reuse the code he had written. As a result, in distributing

---

[18] James Gosling. "Software Sabotage". Email to net.emacs mailing list. April 21, 1983.

[19] Warren Montgomery. "btl emacs". Email to net.emacs mailing list. January 19, 1983.

[20] Steven Zimmerman. "New CCA EMACS Distribution". Email to net.emacs mailing list. January 11, 1983.

[21] Marty Sasaki. "gnu emacs on VMS". Email to net.emacs mailing list. June 3, 1985.

LaBalme's code as part of the GNU emacs, Stallman was in violation of UniPress's copyright. Labalme, having previously asked and received permission from Gosling to distribute the code, had been under the impression that the re-use of his contribution was not an issue.[22]

Just as Gosling had seemingly forgotten about the many contributors to the UNIX Emacs project, he now suffered from amnesia about the use of their contributions with others.[23] But Fen Labalme was unable to locate the original email where Gosling granted permission, so his claims of being able to re-distribute his software were unsubstantiated. Despite this, Stallman refused to replace the code, posturing the situation as a moral one in a posting he made to net.emacs.

> I can't understand how honor could require me to replace any of the code I am distributing in GNU Emacs. Using the modified Gosling display code in GNU Emacs is not immoral. Redistributing any code is always morally good. It's not disobeying the spirit of the law. (We know we really do have permission.) It's not disobeying the letter of the law. (That is, we expect we could win if we were sued. That's the most anyone can ever say.) It's not breaking any promise Fen or I made. Where is the dishonor? [24]

Richard Stallman's posting shows that even beyond the vague justifications he provided in the GNU manifesto, he understood software distribution in moral terms. His apparent obdurateness suggested an increasingly Manichean view that held closed software corporations as evil and open software as good. While the conflict in the AI lab was with a single, clear enemy, Symbolics, with the GNU project, Stallman's quest for software freedoms was a universal moral imperative.

---

[22] Fen Labalme. "Whither GNU EMACS?". Email to net.emacs mailing list. June 26,
[23] Ibid.
[24] Richard Stallman. "honor". Email to net.emacs mailing list. June 18, 1985.

Computer Corporation of America's Steven Zimmerman issued a lengthy rebuttal of Stallman's statement. In disagreement with Stallman's moralist interpretation, he focused on the legality of the situation. He offered a restrictive understanding of the law, arguing that it did not matter if Fen Labalme wrote the code or had a statement from Gosling about being able to reuse the code, it would not be transferrable to others such as Stallman. According to Zimmerman, there would need to be a software licensing statement with extremely specific rights granted, or the code re-use would be illegal.[25] To achieve the distribution rights needed for re-use GNU emacs, Zimmerman suggested that the statement would have to read:

> You have my permission to make copies of Emacs #85 for whomever you want, whenever you want, without limitation. Furthermore, this permission can never be revoked.... Furthermore, you may grant permission to copy these files to anyone else, and they may in turn grant such permission to anyone else, and so on, ad infinitum.[26]

Steven Zimmerman was in no way a lawyer, but he had experienced similar issues himself several years prior. In May 1983, Bell Laboratories pursued Zimmerman's company for using source code from AT&T's internal version of Emacs. Zimmerman was forced to re-write the software and issue a notice to the user community directing them to replace the offending code.[27]

Eventually, Gosling joined the conversation, stating, "In no way did I ever give Fen Lebalm [SIC] (and by dubious transitivity, Richard Stallman) permission to distribute any part of the Emacs that I wrote." Gosling emphasized that the software was

[25] Zimmerman's rationale appears flawed. Applying Zimmerman's logic to Gosling would suggest that his use of code written by Labalme and others would be equally questionable since the contributors did not provide a document explicitly granting Gosling the rights to use or redistribute their software.

[26] Steven Zimmerman. "honor". Email to net.emacs mailing list. June 24, 1985.

[27] Steven Zimmerman. "Important notice for EMACS users". Email to net.general,net.unix-wizards,net.emacs mailing list. June 13, 1983.

not in the public domain, and he claimed that he "was very careful to get everyone who recieved [SIC] a copy to agree to not redistribute it." But just as Fen Labalme was unable to produce any documentation of explicit permission, Gosling could not provide any record of such agreements.[28]

This complex discussion about permissions and copyright was further complicated by the University of Louisiana's Juha Heinanen, who pointed out that Gosling developed his Emacs on Carnegie Mellon's UNIX system and therefore the copyright for the software was governed by AT&T's educational license terms. The educational license agreement specified that UNIX could only be used for "academic and educational purposes;" using the operating system to develop commercial products was strictly forbidden.[29] Any enhancements or addition developed on the system had to be "made available to anyone…without restriction on use, copying, or further distribution."[30] What started as an inquisition into GNU project ended up with the legality of Gosling's sale to UniPress being called in question.

Stallman eventually relented and agreed to rewrite the contested software. "I have decided to replace the Gosling code in GNU Emacs… in order to keep people's confidence in the GNU project," Stallman explained, "I expect to have the job done by the weekend."[31] Ever provocative, Stallman used the announcement as an opportunity to further delineate his opposition to corporations attempting to use copyright to constrain

---

[28] James Gosling. "Re: Permission". Email to net.emacs mailing list. June 21, 1985.

[29] Heinanen assumes that all educational licensing terms are the same. This is not the case. The clauses referenced do not exist in earlier licenses discussed in Chapter 1 (1973 to 1975). Based on the mailing lists from this period (July 1975), Carnegie Mellon does not appear to have been involved in this period, but the actual CMU UNIX license would need to be reviewed to confirm if Heinanen's concerns are applicable.

[30] Juha Heinanen. "Re: Permission". Email to net.emacs mailing list. June 25, 1985.

[31] Richard Stallman. "GNU Emacs". Email to net.emacs mailing list. June 27, 1985.

user access, “Software sharers are happy if you get good software,” Stallman explained, where organizations such as CCA and UniPress sought “ways they can restrict you, because each restriction they can manage to impose means more pressure on you to pay them.”[32] He implicated CCA’s Steven Zimmerman’s arm-chair lawyering in the discussions as an effort to convince users “of the most restrictive interpretation of the law at every juncture--to get you to forfeit as much as possible.”[33] Gosling fared no better. Stallman pointed out how the developer had used the sharing culture to his own advantage. “Gosling says he supports Free Software--yet he was clearly making preparations to sell Gosling Emacs even while urging the community to contribute.”[34] To Stallman, the episode reinforced his concerns over the perfidy of the corporations who were changing the software culture.

On July 5, 1985, Stallman announced that GNU Emacs was now free of Labalme’s Gosmacs code. It had taken just over a week for Stallman to replace the software and test the program.[35] Several weeks later, on August 1, UniPress acknowledged the changes in a message reading:[36]

> You can consider this to be an official statement from UniPress: There is nothing in Gnumacs version 16.56 that could possibly cause UniPress to get upset. If you were afraid to use Gnumacs because you thought we would hassle you, don't be, on the basis of version 16.56.[37]

[32] Ibid.
[33] Ibid.
[34] Ibid. Combine these four Stallman ref’s. as they are all to the same source in a single para.
[35] Richard Stallman. "Independence Day". Email to net.emacs mailing list.. July 5, 1985.
[36] Michael Gallaher. "Unipress and Gnumacs". Email to net.emacs mailing list.. August 1, 1985.
[37] Despite this clear statement of absolution, in 1990 Unipress filed a complaint with IBM after the corporation began to distribute GNU code.

Catalyst

In some ways, the situation with UniPress was simple and easily resolved. The organization never filed suit against Labalme or Stallman. Even if they had, it is unclear if they would have won. Ultimately, Stallman's solution was simply to re-write the offending code. But the situation demonstrated how something as simple as writing software could easily become a litigious mess. Here, software licenses created more questions than answers. Could UniPress sue users of the GNU software over their concerns over the code that Labalme wrote? Could they sue Stallman or Labalme? If the latter had produced written permission from Gosling, would it be legally recognizable? Was it legal for Gosling to sell Unipress software written by others, such as Labalme? Was it legal for Gosling to sell software written on the CMU UNIX systems since it was in violation of the AT&T license agreement? Was it even legal for Gosling to port and sell his own version of the Emacs software written by Stallman?

This last question underscored the precarious nature of the situation for Stallman and the GNU project. He was the original creator of the Emacs software. Five years later, a programmer ported it to a different platform, claimed copyright, and sold it. The corporation that purchased it then had the temerity to suggest that the original inventor of the program could be liable for copyright infringement. The justification behind H.R. 6933, the law that afforded copyright protection to software was that the idea (in this case the Emacs software), rather than the expression of the idea (the UNIX version of Emacs), is copyrighted.[38] By this rationale, it would be Stallman who had the original copyright to the program. And it would be Stallman, not UniPress or Gosling, who had the right to

[38] *Final report of the National Commission on New Technological Uses of Copyrighted Works, July 31, 1978* (Washington: Library of Congress, 1979), 31.

sue. But since Stallman's intention was to make the software available to all, he had essentially left it in the public domain. Unfortunately, this also meant that the software was left exposed; there was nothing to stop an unscrupulous individual from taking the software and proffering it as their own product.

The Gosling Emacs issues also demonstrated that the contributors to the GNU project were legally exposed as well. UniPress had not challenged the GNU project on its re-use of code but had directly approached its programmers. Legality was separate from morality, so regardless of how moral Stallman believed his actions to be, the reality was that he was still legally culpable for the activities undertaken as part of the GNU project.

The episode, although brief, was transformative for Stallman. It revealed the way in which both he and his GNU software project were exposed. To address the issues revealed by the Gosmacs fiasco, Stallman turned to the same legal instruments used by the industry he distrusted – incorporation and software licenses.

The Free Software Foundation

On August 5, 1985, exactly one month after Stallman announced that GNU Emacs had been re-written to resolve the Gosling Emacs debacle, another GNU announcement was sent to the emacs newsgroup. The message laid out a plan to re-organize the GNU project under the new Free Software Foundation. The notice, sent by the foundation's first full-time employee, Leonard H. Tower Jr., heralded the new organization and presented its objectives: "To create GNU as a full development/operating system," and "To distribute GNU and other useful software with source code and the permission to copy and redistribute."[39]

The key part of the announcement was understated: no longer was the effort to support just the GNU project; now the group aimed to also provide "other useful software."[40] That October, the foundation was incorporated as a 501c non-profit organization. With this action, Richard Stallman and his fellow GNU programmers were shielded from liability for their efforts. The incorporation offered them the same legal protections afforded to commercial entities such as Symbolics. With limited liability and a structure designed to go beyond the seminal GNU project, the organization enabled Stallman's ideals to be expanded to other Free Software efforts. Where the GNU project was an embodiment of the Free Software schema of distributed contribution and sharing, the Free Software Foundation provided a tangible platform for the expression of the ideology behind it.

To understand this ideology, it is important to comprehend the hacker culture that had emerged in the MIT AI Lab. As discussed within Steven Levy's *Hackers*, this culture was focused on writing and tweaking programs. The better the software – or the better the hack to an existing program – the higher social standing a hacker achieved. In this way, the culture was similar to the hot rod hobbyists who performed extensive modifications to automobiles. The hot rod culture evolved around the alterations, use, and display of the customized cars. Within this culture, the individual was valued based on the creativity, efficacy, and attractiveness of what they created. Stallman's dilemma can also be seen in

---

[39] Leonard H. Tower. "Another GNU software distribution". Email to net.emacs mailing list. August 5, 1985.

[40] Ibid.

this comparison: the closing of software source code would be akin to the automotive industry welding automobile hoods shut under the guise of protecting copyright.

For the hackers, the medium for individual expression within this culture was the software source code itself. Within this context, software was perceived as speech and to write and share software freely was seen as equal to a U.S. citizen's right to free expression. This right of expression presupposed subsequent freedoms: A writer must be free to compose what they would like. They must be free to distribute their writings. Readers must be free to read and examine the prose, and they must be free to exercise their own rights to respond, augment, or dispel what they have read. If they would like, they could try to sell their compositions. Within the hacker culture such was the case with software as well.

In holding the sharing of software as akin to free speech, hackers perceived any limitations placed on the practice as a civil liberty issue, an abrogation of personal freedom. Since they perceived the supremacy of individual rights over that of any purported authority – such as a corporation -- any discussion of commercial copyright was a secondary concern. In the realm of intellectual property, hackers like Richard Stallman were conservative, holding on to traditional interpretations of how copyright applied to software. Their view -- that source code was freedom of expression -- was consistent with the legal interpretation of copyright law prior to the revisions passed in December 1980 (Public Law 96-517.) Many of the objections to the National Commission on New Technological Uses of Copyrighted Works (CONTU) findings were focused on the fact that the recommendations ignored established jurisprudence on the matter. From a historical interpretation of how the law addressed source code, the sudden pro-business copyright changes were a complete reversal of established interpretations.

With the introduction of the CONTU changes, binary code was now protected by copyright as well. This allowed companies to create artificial scarcity by restricting access to source code in favor of binary copies of software. This limited the purchaser's ability to alter the program. To hackers like Richard Stallman, this was no less than an abrogation of the individual's right to expression. Where the freedom of expression permitted a reader to read and examine the prose, software users could no longer do the same. Likewise, where access to the prose allowed readers to respond, augment, or dispel what they have read, software users could not.

Since its inception, the Free Software Foundation has attempted to articulate its perspective on software freedoms.[41] These efforts eventually evolved into a declaration the organization referred to as the four essential freedoms:

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works and change it so it does your computing as you wish (freedom 1).
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3).[42]

These freedoms extended mainly from the perspective of the software consumer, with only the final freedom being relevant to software developers. Access to the source code was necessary for these freedoms, but eschewing commercialism was not: "You may have paid money to get copies of Free Software, or you may have obtained copies at

[41] Richard Stallman, *Free Software, Free Society: Selected Essays of Richard M. Stallman* (Boston, Massachusetts: Free Software Foundation, 2002), 41. Have you used this ref. earlier?

[42] These freedoms evolved over the second half of the 1980s. They are represented here as they were when last updated (1990). There were only three freedoms initially, with freedom zero added in 1990 as a precursor.

no charge."[43] Users could purchase or sell Free Software – it was not the cost of the programs, but the freedom to access and modify it that mattered.

A GNU Type of License

Where the Free Software Foundation helped articulate the aims of Free Software to a broader movement, the situation with UniPress and Emacs also resulted in another change -- the introduction of Free Software licenses. The conflict with UniPress reified the need to protect the GNU project from the fate that befell Emacs, where Gosling had copied Stallman's program, encouraged others to work on the purportedly Free Software, and then sold the copyright to a company.

Starting on June 3, 1985, Emacs releases began to include the "GNU Emacs Copying Permission Notice," a document that reserved the copyright for the software to Stallman. It also included the conditions under which Stallman granted use of the program:

> Permission is granted to anyone to make or distribute verbatim copies of this document provided that the copyright notice and this permission notice are preserved…I permit everyone to have and run copies of GNU Emacs, at no charge, and to redistribute copies under certain conditions which are designed to make sure that that all modified versions of GNU Emacs remain as free as the versions I distribute. [44]

GNU Emacs General Public License

By March 1986, the document had evolved from a simple permission notice into the GNU Emacs General Public License. The license gave substance to Stallman's claim in the previous year's GNU manifesto that the software would not be in the public

[43] Stallman, *Free Software, Free Society…,. 42.*

[44] Richard Stallman, *Gnu Emacs Availability Information* (Cambridge, Massachusetts: GNU Project, 1985).

domain.[45] To achieve this, Richard Stallman created another type of license that existed on a spectrum between proprietary licenses and the public domain. Public domain software granted users access to do anything: copy, display source, distribute, execute, modify and sell the software. However, it did not retain the copyright to the creator, which is what allowed usurpation by programmers like Gosling. Proprietary licenses retained copyright, but generally limited users to executing pre-compiled binary copies of the program. Stallman's license created a hybrid where the creator retained ownership of the copyright, but allowed their intellectual property to be copied, executed, modified and sold (See Table 4.) With Stallman's GNU Emacs General Public License, the only restrictions were on distribution; any derivative works had to be released under the same GNU Emacs license. This type of license, which stipulated the preservation of rights within derivative works, eventually became known as a Copyleft license, a reference to Li-Chen Wang's May 1976 Tiny BASIC code listing that bore a "@COPYLEFT" notice.[46] As with Wang's original, the Copyleft license turned traditional copyright on its head – in this case, by using an intellectual property contract to provide user rights rather than to take them away.

---

[45] Richard Stallman, "The Gnu Manifesto," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985, 30.

[46] Li-Chen Wang, "TINY BASIC FOR INTEL 8080," *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, May 1976, 15.

Table 4: User Rights by Software License

	Public Domain	GNU	Commercial
Copy	Yes	Yes	**No**
Display Source	Yes	Yes	**No**
Distribute	Yes	Yes	**No**
Execute / Use Software	Yes	Yes	Yes
Modify	Yes	Yes	**No**
Re-Sell	Yes	Yes	**No**
Creator Retains Copyright	**No**	Yes	Yes
Change Copyright	Yes	**No**	Yes

This new type of license introduced confusion as the programming community struggled to understand its parameters. For instance, Sun Microsystems programmer John Gilmore had been writing a tape control program, TAR (Tape Archive and Recovery) for the GNU project. Concerned with Richard Stallman's new "funny restrictions," he spoke with Stallman to ensure that his TAR program would remain in the public domain.[47] Gilmore later recalled, "Up until then, most licenses were very informal."[48] Despite his initial reservations, the programmer soon became an ardent supporter of Stallman's Free Software philosphy, and he later co-founded Cygnus Solutions, a corporation dedicated to providing support for the GNU project's Free Software.

Other programmers also expressed concerns with the relationship between the GNU project and existing public domain programs. UCLA Computer Science Department programmer Scott Turner expressed concern that the GNU project could take public domain software, apply the GNU Emacs General Public License to it and then the

[47] John Gilmore. "Re: gnu copyrights". Email to net.emacs mailing list. May 21, 1986.
[48] Williams, 125.

software would be restricted to the GNU license terms.[49] Turner's concerns essentially describe what Gosling had done with Emacs and a commercial license. It is interesting that Turner's concern over co-option emerged in response to Stallman's project when the same potentiality already existed with proprietary copyright licenses.

Despite the initial confusion, many programmers embraced the approach and began to discuss how to expand the GNU Emacs license to their own programs. In November 1986, John Gilmore emailed Richard Stallman, suggesting that the Emacs license be made more universal. "You should probably remove "EMACS' from the license and replace it with "SOFTWARE" or something. Soon, we hope, Emacs will not be the biggest part of the GNU system, and the license applies to all of it." [50]

Gilmore was not the only one contemplating a more general approach. Stallman was already considering ways to extend the Emacs license to cover other GNU programs being worked on. His solution was to purge all the licenses' specific references to Emacs and re-write the document to provide a "generic copyright umbrella for GNU Project Software." The result was the GNU General Public License.[51]

The GNU General Public License

On February 25, 1989, Free Software Foundation employee Leonard Tower shared the new GNU General Public License (GPL) with the GNU programming community via the project's newsgroup.[52] The new license began with a lengthy

[49] Scott Turner. "Re: gnu copyrights". Email to net.emacs mailing list. May 23, 1986.
[50] Williams, 125.
[51] Ibid. 126.
[52] Leonard Tower. "New General Public License". Email to gnu.announce mailing list. February 25, 1989. gnu.announce. Accessed August 14, 2018.

preamble that made clear its intention to support the software rights sought by the Free Software Foundation, the "freedom to share and change Free Software--to make sure the software is free for all its users." The document made clear that the Free Software being conveyed was focused on "freedom, not price," and the software created under the license could be sold or given away free. The fundamental requirement was that the user of the software received the source code and could change it as they saw fit. The preamble defined how the license worked, protecting the consumer's rights by making "restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights." In order to fully convey, "these restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it."[53]

The terms of the license outlined how these rights were preserved. The first section mandated the copy and distribution of the license along with the source code.

> You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you… keep intact all the notices that refer to this General Public License and to the absence of any warranty; and give any other recipients of the Program a copy of this General Public License along with the Program. [54]

The second section dictated the reciprocity inherent in a Copyleft license – that any distribution or derivative work carried with it the same protections under the GNU General public license, so any modifications that a programmer decided to distribute or publish, had to be "licensed at no charge to all third parties under the terms of this General Public License." And although the software itself was free of cost, the license was clear that it could be sold, in that it was permissible to "charge a fee for the physical

[53] Richard Stallman, *GNU General Public License: Version 1* (Cambridge, Massachusetts: Free Software Foundation, 1989).
[54] Ibid.

act of transferring a copy," or "offer warranty protection in exchange for a fee." The license then addressed the need for source code. It permitted the distribution of the software in pre-compiled binary form provided that the source code accompanied it or was made available upon request. The license concluded with an exculpatory section freeing the creator and distributors of software created under the license from any legal liability, obligations of warranty or support. The GNU General Public License was designed to be self-executing; where the use, modification, or distribution of the software were interpreted as an acceptance of the license terms. Likewise, with each redistribution, "the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions."[55] These rights remained whole; subsequent re-distribution could not be used to lessen the original rights granted within the license.

The GPL was an embodiment of the hacker values revered by Stallman and it guaranteed users the freedoms sought by the Free Software Foundation. Users could run the program for any purpose, study how the program worked, change it, and re-distribute it. Most importantly, the license was intended as a tool to spread these ideals and encourage the creation of Free Software. With the license, the Free Software Foundation also included an appendix outlining how programmers could use the document as a template and apply its terms to their own programs.[56]

For Free Software adherents like John Gilmore, the GNU Public License was precisely what they had been looking for. Other programmers were not as enthusiastic.

[55] Ibid.
[56] Leonard Tower. "New General Public License". Email to gnu.announce mailing list. February 25, 1989.

"Have you read the GNU Public License? It's a legal virus, contaminating everything it touches," complained Confederate Microsystems programmer James Maynard in a November 1989 newsgroup discussion. If you use a GNU compiler, Maynard posited, "the entire program you write is magically brought under its terms. I'm not sure if this applies to GNU cc's [C Compiler] libraries or not, but it wouldn't surprise me."[57] Maynard's perspective emanated from a misunderstanding of the licensing terms: he did not discern between software derived from GNU projects (which must be re-distributed under the GPL) and programs created using the software (which bore no such restrictions.) Others on the newsgroup clarified this distinction and explained to Maynard that the GNU license did not require software written on the platform to be covered under the license as well.[58]

The GNU Library General Public License

Maynard's confusion exemplifies the concerns that some programmers had with the new type of license.[59] These came from misunderstanding the document's terms; however, this confusion was so frequent that Richard Stallman soon drafted another license, the GNU Library General Public License (LGPL.) This new license was intended to clarify the role of programming libraries, which were central to the uncertainty regarding software created with GNU tools. The issue was that there were three potential scenarios and little distinction between them:

---

[57] James Maynard. "Re: Perl may be great, but...". Email to alt.religion.computers mailing list. November 24, 1989.

[58] Greg Lindahl. "Re: Perl may be great, but...". Email to alt.religion.computers mailing list. November 27, 1989.

[59] Despite these early objections James Maynard went on to support the Hercules Project, an Open Source IBM S/370 emulator. He also gained a degree of infamy on the Internet for wearing an indiscrete homemade "Tron" outfit to science fiction conventions. See http://www.conmicro.com/.

Scenario one would be a program that used GPL software, modifying it, and redistributing it. By the terms of the GPL, the modified software was considered a derivative work. According to the license, the software must also be released under the GPL license.

Scenario two would be a program compiled using a GNU compiler such as GCC (the GNU C Compiler.) The program did not use any existing code at all. It is merely compiled using the GNU tools. In this case, the software could be distributed under any type of license, without any of the GPL restrictions.

The problem comes with scenario three. Programming languages sometimes used collections of pre-written functions called libraries. Under the terms of the GPL license, any program that used a GNU program library, would be required to also be released under the GPL. Although the software is simply linked to the library, the inclusion of GPL code makes the software a derivative work. As a result, programmers who wished to create proprietary programs could not use the GNU libraries. Since this was not his intention, Richard Stallman addressed the issue by developing the GNU Library General Public License (LGPL). The LGPL lacked the reciprocity required by the standard GPL and the document made clear that software libraries released under the LGPL could be used without the resultant program needing to be released under the same license. In Richard Stallman's words, this allowed "developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them."[60]

---

[60] Richard Stallman. "Latest draft of library license". Email to gnu.g++.help,comp.lang.c++ mailing list. January 5, 1991.

Academic Licenses

While the GNU Emacs license was evolving into a general-purpose Free Software license, other, similar "open" software licenses had also begun to emerge. Like the GNU GPL, the earliest of these was also the result of changes to the software industry – in particular, the commercialization of UNIX and the closure of its source code.

Berkeley Software Distribution License

AT&T had restricted access to the UNIX source code and began to offer the operating system commercially. The UNIX licensing changes were particularly concerning for institutions that were running the Berkeley Standard Distribution (BSD), a variant of the UNIX operating system that had been developed by the University of California, Berkeley. When AT&T changed their licensing terms, Berkeley's Computer Systems Research Group (CSRG) began work on a version of their operating system with all AT&T source code removed. With the new effort, the group was faced with a challenge: how to license the software that they had created.[61]

Their solution was the Berkeley Software Distribution License, a short contract with four simple clauses. The first two clauses permitted free distribution of the software (in both binary and source code) provided that the copyright notice and conditions were included. The third clause required that all advertising materials for any derivative products or programs include an acknowledgement that it incorporates software developed by Berkeley. The final clause stipulated that neither Berkeley nor any of BSD's developers could be referenced in a way suggesting that they endorse or promote

[61] Salus, *A Quarter Century of UNIX. 223.*

any derivative products. The remainder of the license was similar to the GPL's exculpatory disclaimer disavowing all liability, warranty, or support.[62]

The BSD license was not a Copyleft license like the GPL; there was no reciprocity requiring that any derivative efforts be released under the same license terms. But the license was similar in such that it used the software contract in a permissive way, giving the program's user virtually unrestricted use of the software and its derivations. The simplicity of the BSD license made it an appealing template for other licenses. Software developers could take the basic license document and replace the references to the University of California/Berkeley with their own name or organization and use it for their own software.

The Massachusetts Institute of Technology License

The Berkeley Software Distribution License informed the creation of several other academic licenses, most notably the license for software produced at the Massachusetts Institute of Technology. This variant simplified the BSD license by removing the four clauses in favor of an introduction that itemizes the exclusive rights granted by the license:

> Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions…[63]

---

[62] *Berkeley Software Distribution License* (Berkeley, California: The Regents of the University of California, 1988).

[63] *Massachusetts Institute of Technology License* (Cambridge, Massachusetts: Massachusetts Institute of Technology, 1988), https://opensource.org/licenses/MIT. (Accessed July 15, 2019).

The final sentence allowed for the introductory paragraph to be followed by an iterative list of stipulations for the use of the software. This made the license extensible; any additional clauses can be simply added. The text of the paragraph also improved upon the BSD license by its specificity in outlining almost all the exclusive rights under copyright law.[64]

Summary: From the GNU Project to Free Software Movement

Richard Stallman started the GNU project in response to changes within the open computing culture resulting from the commercialization of software. His original intention was to create a software ecosystem that was separate from, and unassailable by, the unethical practices he observed in the software industry. The Gosling episode had demonstrated the need for Stallman and the GNU project to have legal protections. This led to two key developments: incorporation and licensing.

Stallman incorporated the Free Software Foundation as a non-profit organization focused on supporting the creation of his GNU project and other Free Software. The organization served as a platform for supporting Stallman's GNU project and, by defining the principles of Free Software, was able to articulate his ideology to a larger audience.

Stallman developed the GNU Emacs license as a legal instrument to protect his software and ensure that it could not again be taken, modified, and sold as a commercial project. This soon evolved into the general use GNU General Public License. Following

[64] Lawrence E. Rosen, *Open Source Licensing: Software Freedom and Intellectual Property Law* (Upper Saddle River, New Jersey: Prentice Hall, 2005), 86.

the introduction of the GNU license, other similar academic licenses also began to emerge.

Beyond affording Richard Stallman and the GNU project legal protections, the Free Software Foundation and the Free Software licenses allowed for Stallman's ideas to spread to other software developers. No longer was Free Software just synonymous with the GNU project. Other groups, such as the BSD programmers at Berkeley's Computer Systems Research Group, were able to preserve the open software culture through the creation and licensing of their own programs.

Just as these legal mechanisms allowed for Richard Stallman's ideology and schema to spread to other programmers, his GNU software brought his message to a broader audience. In a January 1989 *New York Times* article, journalist John Markoff detailed Stallman's crusade to the broader public. Markoff reported that Richard Stallman "wants to bring back the good old days when programming was a communal activity and those toiling at the craft freely shared their ideas." The author detailed how over the previous two decades, copyright protection became the "foundation of the modern software industry," and as such, Stallman's argument "raises bitter objections from many programmers and companies."[65] Despite these protests, Markoff reported that Stallman's ideas were gaining traction with computing corporations such as I.B.M., the Digital Equipment Corporation (DEC), and Sun Microsystems, all of which had begun to support open software standards.[66] Beyond the corporate interest in open specifications,

[65] John Markoff, "One Man's Fight for Free Software," *New York Times*, January 11, 1989. D1.

[66] Markoff's discussion of open software standards refers to a corporate drive to standardize technologies away from proprietary protocols. This movement was contemporaneous to the publication of the article, and it emerged, in part, from the commercialization of UNIX. Once AT&T began to re-license UNIX to corporations, a myriad of derivative, proprietary systems began to emerge. Hewlett Packard's UNIX was HPUX, IBM's UNIX was AIX, Microsoft proffered Xenix, and so on. In 1987, following

several corporations, including Data General, Hewlett Packard, Sony, and Sun Microsystems, had also begun to officially support the GNU project development efforts. In addition, Apple co-founder Steve Jobs decided that the GNU software would come bundled with his new computer systems.[67] Jobs, in exile from his first company, Apple Computers, had recently formed the NeXT corporation which offered high-end workstations running an operating system based on BSD UNIX.[68]

Where some mainstream computing companies had begun to accept Richard Stallman's ideas and software, there was no easy way to share the GNU software with the public at large. The developers that worked with Stallman communicated through newsgroups via UUCP (Unix to Unix Copy). They could download the GNU software from anonymous sites on the ARPAnet or from one another's computers. They could work on their own contributions and share it likewise, but home consumers generally lacked the skill and the network access needed for the software to grow beyond the highly technical publics in these newsgroups. The situation was about to change, however, as the networks connected to the ARPAnet would soon begin their evolution into the commercial Internet. By September 1989, companies such as CompuServe and

---

AT&T's purchase of a sizable percentage of Sun Microsystems, the two companies announced that they would be merging the version of BSD used by Sun and AT&T's system V Unix. Their competitors (DEC, HP, IBM, Nixdorf, Siemens, etc.) saw the move as a hostile gesture intended to make UNIX proprietary. In response, these companies formed the Open Software Foundation, a consortium dedicated to creating a single specification for UNIX. By the time of Markoff's article, Sun and AT&T had formed their own group, UNIX international, dedicated to establishing their own standard and marketing their new, combined version of UNIX. These groups should be understood to be establishing a specification of what UNIX was, rather than any explicit support for Richard Stallman's philosophy or project.

67 Markoff, "One Man's Fight for Free Software," D7.

68 NeXT was purchased by Apple Computer, and the BSD-derived operating system, NeXT Step, informed the creation of Mac OSX. The later OS continued the practice of including Free Software and Open Source software.

MCI Mail began to connect the public with the burgeoning network.[69] The following year, Tim Berners-Lee, a programmer at the European Organization for Nuclear Research (CERN) in Switzerland, developed the hypertext markup language (HTML) and the server and browser software to support it. His innovation, the World Wide Web, was released as Free Software under a CERN-specific license in 1994.[70] The platform would eventually make the Internet – and Free Software - accessible to ordinary people.

---

[69] Yvonne Lee, "Compuserve, MCI Mail Introduce Gateways to Internet Network," *Infoworld*, September 25, 1989, 32.

[70] Tim Smith and François Flückiger, "Licensing the Web," CERN, March 12, 2014, http://home.web.cern.ch/topics/birth-web/licensing-web, (accessed August 15, 2015). The software was released into the public domain in 1993, but CERN re-released the software once they realized that it could be appropriated by an unscrupulous corporation. The CERN software license was not a copyleft license. It is described within the referenced CERN article as Open Source; however, this is anachronistic: This license was created in 1993, but the Open Source re-branding of Free Software did not occur until 1998.

CHAPTER 5: FROM OUTER SPACE TO INTERNET

In the early morning hours of October 5, 1957, a Soviet R-7 ICBM rocket roared to life in southern Kazakhstan. Within minutes, the missile delivered its payload, Sputnik 1, the world's first artificial satellite, to a low earth orbit. In the weeks following the launch, the White House was soon flooded with bi-partisan concerns that the nation had been surpassed technologically by the Soviets.[1] Among Eisenhower's responses to the concerns was the creation of the Advance Research Projects Agency (predecessor to DARPA), a group designed to facilitate scientific research projects with governmental, private, and educational institutions. Since its inception, the agency has been responsible for the cooperative development of many modern technologies including the ARPAnet.

This chapter explores the origins of the Internet in the government funded ARPAnet and details key technologies involved in the popularization of the commercial Internet, such as the World Wide Web. This analysis demonstrates the symbiotic relationship between Internet technologies and Free Software. Finally, the chapter considers the role of Internet adoption in the growth of the Free Software Movement and the emergence of key technologies such as the Linux operating system.

ARPAnet

In February 1966, computer scientist Robert Taylor was the newly appointed director of the Information Processing Techniques Office (IPTO), an independent section of the ARPA agency. The IPTO was tasked with supporting advanced computer research and development efforts. In Taylor's third floor office at the Pentagon, a row of teletype

---

[1] Charles Murphy, "The White House Since Sputnik," *Fortune*, January 1958, 98-101.

terminals enabled the IPTO director to connect to ARPA-funded mainframes throughout the nation. Each computer system was unique, requiring different procedures to connect, and each used different operating systems and programming languages.[2] The result was that Taylor and his staff required a broad cross-section of skills to operate the systems. The nature of the teletype terminals also added to their frustration. Electromechanical typewriters were used to send and receive messages. When they received a response, the device noisily typed it out on paper. Using several such devices at the same time overwhelmed the office in an unbearable cacophony. The “terminal problem,” as Taylor called it, had beleaguered previous IPTO directors back to the office’s founder, computer scientist J.C. R. Licklider.

Shortly after becoming the leader of the IPTO, Taylor approached his boss, ARPA Director Charles Herzfeld, about the issue. Taylor explained that many of the IPTO contractors were requesting additional computing resources. Much of the group’s research was conducted at universities and it seemed that increasingly each principal investigator desired their own computer. However, Dartmouth’s Time-Sharing System had proven that a centralized mainframe could be shared by many users. Funding individual computers for each principal investigator meant an expensive duplication of resources throughout the IPTO research community. Taylor reasoned that if they were able to find a way to electronically link the machines, the researchers could share computing resources. Creating timesharing communities would decrease the computing

---

[2] Katie Hafner, *Where Wizards Stay Up Late: The Origins Of The Internet* (New York, New York: Simon & Schuster, 1998), 11-13, 40-41.

costs and, at the same time, eliminate the "terminal problem."[3] The key challenge would be in finding a way to achieve compatibility among the vastly different computer systems used by the IPTO researchers. Taylor was confident that they could achieve this. Herzfeld approved the proposal and granted a million-dollar budget on the spot.[4] The ARPAnet project was born.

Despite Bob Taylor's glib confidence when selling the network, the reality was that the ARPAnet represented a series of significant challenges. A series of technological obstacles would need to be overcome to create the network. Physical connections were the easy part – AT&T already had a telephony network in place.[5] But what would the network design look like? How would it route network traffic? How would the network be supported? What would happen if part of this network became unavailable? A series of fortuitous events led to answers in short order.

After the project was approved, Bob Taylor recruited Lawrence Roberts, who at that time had been working at ARPA funded MIT Lincoln Labs. Roberts had worked with computer scientist Wesley Clark, father of the TX-0, predecessor to Digital Equipment Corporation's PDP minicomputers. Clark provided the answer to how the network could be supported by recommending the use of dedicated computers at each network location.[6] The network design was further informed by the work of several contributors. Computer Scientists Paul Baran and Donald Davies had both developed plans for a packet-switched network where the data would be assembled in packets of

---

[3] Robert Taylor, interviewed by William Aspray, Palo Alto, California, February 28, 1989. https://conservancy.umn.edu/bitstream/handle/11299/107666/oh154rt.pdf.

[4] Hafner, 41-42.

[5] Ibid., 51.

[6] Ibid., 45,73.

message blocks. Each message block contained header information and part of the data being transmitted. The header within each block included destination information for the packet. This allowed the data to be re-routed and properly re-assembled in the event of a network disruption.[7] Roberts learned of these innovations at a 1967 symposium sponsored by the Association for Computing Machinery. Roberts was presenting the ARPAnet project at the Gatlinburg Tennessee conference. There he met a member of Donald Davies' team, Roger Scantlebury. At a hotel bar following a long day of presentations, Scantlebury met with Roberts and several other scientists. At this meeting, Scantlebury recommended the packet-switching network design for the proposed network.[8]

Building the Network

By July 1968, Lawrence Roberts solidified the network design and created a request for proposal that was sent to the dominant computing firms of the era. IBM, DEC, Honeywell, and Raytheon all bid on the job. In December of that year, the contract was awarded to a small company outside of Cambridge, Massachusetts.[9]

Bolt, Beranek and Newman (BBN) was founded in 1948 by Richard Bolt and Leo Beranek, both MIT faculty who had performed acoustic research for the Navy during World War II. The company frequently recruited staff from their alma mater and became involved in computing after luminaries like J.C. R. Licklider joined their ranks. Shortly before the ARPAnet request for proposal, BBN had hired Bob Kahn, an electrical

---

[7] Janet Abbate, *Inventing the Internet* (Cambridge, Massachusetts: MIT Press, 1999), 18-22.
[8] Hafner, 76,77.
[9] Ibid., 79.

engineering professor on leave from MIT. Kahn was working in the company's information sciences division and had recently started research on computer networking. Unaware of the ARPAnet project, Kahn had been sending memos to both Bob Taylor and fellow former MIT researcher Lawrence Roberts about his efforts. When BBN received the ARPAnet request for proposal, Kahn was a natural fit.[10]

The BBN proposal was the product of six months of internal planning and research. The 200-page proposal cost over $100,000 to produce, but it was rich in detail and provided a clear blueprint for the ARPAnet.[11] The organization proposed to create four Interface Message Processors (IMPS) based on the Honeywell DDP-516 computer to support the network backbone.[12] The software for the network would be written in the Honeywell's low-level assembly language. The high-performance computer and low-level programming approach resulted in a software design that would be able to achieve over ten times the specified bandwidth (data transfer rate) requirements.[13]

The challenges in implementing the proposed network were not limited to BBN. The staff maintaining the computers at the sites hosting the ARPAnet had a key role to play as well. "We had lots of questions," UCLA computer scientist Steve Crocker recalled: "…how IMPS and hosts would be connected, what hosts would say to each other, and what applications would be supported."[14] Representatives from the first four ARPAnet sites began to meet in the summer of 1968 to discuss the possibilities. Their initial discussion was memorialized in the first "Request for Comment" (RFC)

---

[10] Ibid., 81-82,95.
[11] Ibid., 100.
[12] Abbate, 57.
[13] Hafner, 110,100.
[14] Ibid.

document.[15] Besides discussing the immediate technical concerns that the group sought to address – the initial connection between two computers on the network, the document achieved another key thing; it recorded the creation of the Network Working Group.

The Network Working Group was comprised of representatives from each ARPAnet connected site. As hosts of the ARPAnet, they were responsible to determine computer software standards for the new network. Their relationship to BBN's efforts were explained in the introduction of RFC-1, with the document suggesting, "The software for the ARPA Network exists partly in the IMPs and partly in the respective HOSTs. BB&N has specified the software of the IMPs and it is the responsibility of the HOST groups to agree on HOST software."[16]

Where BBN was tasked to create a functioning network, the Network Working Group needed to determine what could be done on the platform. The team had to come to an agreement on how the ARPAnet host computers would share resources. Consistent with the open computer culture of the period, the group wrote, shared, and modified programs.[17] As they proposed the standards for the emerging network, they created RFC documents that were distributed to others in the network community for consideration, response, and adoption.[18]

By the fall of 1969, the first two IMP's had been assembled and installed at the Stanford Research Institute (SRI) and the University of California Los Angeles (UCLA.)

---

[15] Initially, RFCs were typed and sent via the U.S. postal service to the various ARPAnet working group members. This eventually transitioned to the electronic transmission with the growth of the network and introduction of file sharing protocols such as FTP. See Abbate, 74.

[16] Steve Crocker, *Network Working Group Request for Comment 1* (Santa Barbara, California: University of California, 1969).

[17] Hafner, 145.

[18] The cautious and collaborative RFC approach remains in place on the Internet where the documents are produced by various working groups and maintained by the Internet Engineering Task Force (IETF) and the Internet Architecture Board (IAB).

BBN's network software was installed on the computers at these sites. The program, if it worked correctly, would allow one of the computers to transparently connect to the other as if it were a local terminal. On October 29, 1969, the network between the two sites was activated. UCLA undergraduate Charles Kline used UCLA's SDS Sigma 7 mainframe and successfully connected to SRI's timesharing computer, a SDS 940.[19] From the terminal at UCLA, the network was transparent. The connection and login were performed as if the Sigma 7 was just a terminal connected to the SRI computer.[20] By December of that year, the other two network nodes from the original ARPAnet contract were activated.[21] The University of California Santa Barbara's IBM System 360 mainframe and the University of Utah's PDP-10 completed the initial network. The IMP controllers and packet-switching had all worked as intended, but to fulfill Bob Taylor's original vision of interconnected timesharing research communities, the network still needed to grow.[22]

Inter-Networking

The IPTO group moved quickly to add the other ARPA research institutions to the growing ARPAnet. Starting from the original four nodes in 1970, the network expanded to twenty-three locations by the spring of 1971. The following year, Lawrence Roberts, now director of the IPTO, organized a public demonstration of the new network

---

[19] The intimal test performed by Kline did connect successfully; however, the remote SRI computer crashed during logon. This was due to a local issue with the machine. The computer was reconfigured, and within hours, subsequent connections were made that did not see this problem.

[20] Hafner, 152-154.

[21] Node is a networking term referring to the connection point on a computer network. This can refer to either separate network through a routing device or a local computer.

[22] Robert Taylor, interviewed by William Aspray, Palo Alto, California, February 28, 1989.

at the first International Conference on Computing Communications (ICCC).[23] The conference had over a thousand participants, and the IPTO allowed all of them to experience network sessions on computers throughout the nation. The experience reified the concept of networking for the conference's attendees and resulted in many universities and research institutes requesting connections to the new network. And the network grew; within a year of the demonstration there were forty-five nodes on the ARPAnet. Within four years, the network had grown to 111 network locations.[24]

By the early 1970s the ARPAnet had begun to rapidly evolve. Protocols, such as TELNET, allowed for direct connections to remote ARPAnet hosts. Like the initial test connection, this allowed network users to log on to a remote computer and use it as if they were at a local terminal. The network also began to feature several indirect usage methods as well. File Transfer Protocol (FTP) allowed network users to utilize remote computer resources without having to log on or know how to use the remote computer's operating system. The network user only needed to know the commands required to upload or download files within FTP.[25] The FTP protocol quickly evolved into a system to deliver electronic messages on the network. This use of FTP for "Network Mail" as defined in late 1973 was immensely popular and a driving force in network adoption.[26]

The interest in computer networking began to transcend the ARPAnet. In September 1974, IBM announced its own networking system, Systems Network

---

[23] Robert Taylor resigned after being sent to Saigon in 1969 to investigate the inconsistent reports (casualties, supplies, etc.) coming from the battlefront. Following his resignation, Taylor began work on a computer graphics project at the University of Utah. Within a year, Taylor's efforts moved to the Xerox Palo Alto Research Center where computer scientist where working on several computer graphics projects, including the Graphical User Interface (GUI) and WYSIWYG editing programs.

[24] Campbell-Kelly et al., *Computer*, 283.

[25] A Bhushan, *RFC 114: A File Transfer Protocol* (Network Working Group, 1971).

[26] Campbell-Kelly et al., 284.

Architecture (SNA). This network software allowed IBM computers to directly communicate with each other over a proprietary network.[27] DEC followed suit and introduced the DIGITAL Network Architecture (DNA), hardware and software technologies that allowed for the creation of the DECnet network. DECnet enabled networking of DEC's PDP-11 computers for the purpose of transferring data, sharing resources, or performing an early form of distributed processing.[28] Both the SNA and DECnet networks were limited to computers from the same manufacturer. For the computer vendors, this proprietary networking was beneficial since they retained complete control of the network technologies – and the sales that went with it. However, the limitations the vendors imposed on their network software meant that popular functions, such as "Network Mail," would be limited only to others on the restricted, proprietary network. To transcend these network boundaries, a method for inter-network communications was needed.[29]

In early June 1973, at the American Federation of Information Processing Societies (AFIPS) National Computer Conference in New York, the International Packet Network Working Group (INWG) met to discuss proposals to achieve inter-networking. The consortium, modeled after the ARPAnet Networking Group, had formed the previous year following Lawrence Roberts demonstration at the International Conference on Computing Communications (ICCC). After the initial meeting, they formed two sub-groups to develop solutions for the inter-networking challenges. The first sub-group

[27] *Systems Network Architecture General Information* (Research Triangle Park, North Carolina: IBM Systems Development Division, 1975). ii.

[28] *PDP-11 Decnet-11m, Phase II Brochure* (Maynard, Massachusetts: Digital Equipment Corporation, 1978). 2.

[29] Campbell-Kelly et al., 285.

considered packet transfers between networks, while the second focused on developing network protocols that would allow devices on the network to communicate with each other.[30] Directing the group was Vinton Cerf, an assistant professor at Stanford, who, as a graduate student at UCLA, had been heavily involved in writing the Network Control Protocol used to connect the ARPAnet's first two nodes. Leading up to the convention, Cerf had been working with Bob Kahn, who by this point had departed BBN and was employed by the IPTO. Kahn had helped demonstrate the ARPAnet at the 1972 ICCC conference and had since been attempting to integrate the ARPAnet with radio and satellite-based networks. At the meeting, Vinton Cerf and Bob Kahn discussed their efforts with the others in the INWG. They had reached two determinations. First, was the realization that to conjoin the networks, a dedicated computer acting as a network gateway would be needed. Second, they determined that the ARPAnet's host-to-host Network Control Protocol would not be able to meet the demands of the proposed inter-networking.[31]

In May 1974, Vinton Cerf and Bob Kahn published "A Protocol for Packet Network Intercommunication" in the Institute of Electrical and Electronics Engineers' journal, *Transactions on Communications*. The paper described the Transmission Control Protocol, a replacement for NCP that would allow for inter-networking.[32] Their proposal was eventually refined into two structures, Transmission Control Protocol (TCP) and the

[30] Alexander McKenzie, "INWG and the Conception of the Internet: An Eyewitness Account," *IEEE Annals of the History of Computing*, January-March 2011, 66-67.

[31] Hafner, 223-225.

[32] Vinton Cerf and Robert Kahn, "A Protocol for Packet Network Intercommunication," *IEEE Transactions on Communications*, May 5, 1974, 637-48.

Internet Protocol (IP), that jointly formed TCP/IP, a networking suite that met the INWG's goal of allowing the interconnection of divergent networks.

The Internet

The Internet arose from the inter-connection of many existing networks. The modern conception of the network is the product of a series of incremental changes made to the ARPAnet. In 1984, the ARPAnet's military constituents had left to form the separate and secure MILNET. Following this departure, the ARPAnet traffic was moved to the National Science Foundation's NSF backbone. On April 30, 1995, the NSFnet backbone was retired and the network was handed off to commercial Internet Service Providers (ISP) and the modern, commercial Internet was born.[33] The very first step in this progression was the adoption of the TCP/IP internetworking protocol.

The Adoption of TCP/IP

Despite the 1974 introduction of the protocol suite, TCP/IP was not operational for nearly a decade. The TCP/IP software needed to be written and its protocol standards developed and tested.[34] More burdensome, the applications that previously hosted ARPAnet services also had to be re-architected for the new protocol. New specifications were developed for programs such as TELNET and FTP. Email also now emerged as a separate, FTP-less process via the Simple Mail Transfer Protocol (SMTP).[35]

[33] Abbate, 199.

[34] By the time of the ARPAnet's January 1983 transition to TCP/IP, the protocol itself had gone through several iterations. The protocol went live on TCP/IP version four.

[35] J. Postel, *RFC 801: NCP/TCP Transition Plan* (Network Working Group, 1981).

On January 1, 1983, the ARPAnet switched to TCP/IP as its core networking protocol. Network traffic sent in the obsolete NCP protocol was no longer routed.[36] With the ARPAnet now able to interface with other isolated networks, the event marked the technical birth of the Internet.[37] Each device directly connected to the new network now had a TCP/IP address, a unique number comprised of four sections separated by a period. Each section, called an octet, signified the device's location within the hierarchy of the global network.[38] For example, an IP address might look like this: 128.180.1.16.[39] The different sites on the Internet were now directly accessible without bridging networks, but the numerical addresses were awkward and difficult to remember.

---

[36] "TCP/IP Internet Protocol," Living Internet, https://www.livinginternet.com/i/ii_tcpip.htm, (accessed October 29, 2018).

[37] Although interconnected, the network was still limited to research facilities, and modern Internet interfaces such as the World Wide Web had yet to be invented.

[38] Each octet of an IP address represents an 8-bit binary field, hence the name. An 8-bit field can go between 0 (Binary 00000000) and 255 (Binary 11111111), so each octet can only contain numbers within this range. With TCP/IP, 0 is unusable and 255 is the broadcast address, so each octet represents 254 usable addresses.

[39] The first octet (128) represents the highest level of the Internet hierarchy, a Class A network. The next octet (180) represents the next level down, Class B networks. The Class B in this example (128.180.x.x) is owned by Lehigh University. From this IP address space, the university can theoretically host 65534 addresses. Or, they could create a series of Class C sub-networks. These are represented by the next octet (1). A Class C network can have up to 254 devices. Each device on a TCP/IP network has its own TCP/IP address. Each TCP/IP service hosted on a computer also has its own port number. These ports are standardized and intended to be consistent across all TCP/IP devices regardless of hardware or operating system. For example, the File Transfer Protocol (FTP) uses port 21. A computer looking to connect to FTP on 128.180.1.16 would thusly go to 128.180.1.16:21. The TELNET terminal protocol resides at port 23 and the mail transfer protocol, SMTP, at port 25. TCP/IP addresses can be conceived in a manner similar to postal addresses. The IP address would be akin to a street address, and the port number like an apartment within the building.

The Berkeley Internet Name Domain Server

Where IP addresses are not overly complex, using them to access commonly used network resources is cumbersome and difficult. Originally, sites maintained a host file that translated the network addresses to more friendly names.[40] However, as the network grew, maintaining a local list of network addresses became increasingly difficult. In November 1983, the Network Working Group began to discuss the need for simple, human readable domain names.[41] They outlined specifications for the Internet Domain Name System (DNS) for the Internet.[42]

Within months of the publication of the Domain Name System specifications, the Berkeley Internet Domain Server (BIND), was written and released by Berkeley's Computer Systems Research Group (CSRG) for Berkeley Standard Distribution (BSD) UNIX 4.2.[43] The software performed the role of both name server and resolver and was

[40] Vestigial forms of this approach can still be found on modern computers. On a windows computer, this can be found in C:\Windows\System32\drivers\etc\hosts. On most UNIX-based computers, this resides in /etc/hosts. The invocation of the UNIX (POSIX) path structure on Windows reflects the operating system's implementation of the BSD TCP/IP stack.

[41] The document described a system where a "Domain Name Space" was created. This tree-structured hierarchy would allow for a root domain, subdomains, and hostnames. The Domain Name Space would be hosted by "Name Servers," programs that hold information about the domain's tree structure. These Name Servers would be contacted by "Resolvers," programs that would translate a user request for a domain name into an actual IP address based on the information returned by the Name Servers.

[42] The source uses abstract terms but describes the basic DNS hierarchy seen today. For example, a top-level domain such as .edu could have a Domain Name Space defined for organizations such as Lafayette (lafayette.edu) or Lehigh (lehigh.edu). Each of these Domain Name Spaces may have subdomains, such as mail.lehigh.edu. For example, the resolver for a local user attempting to go to mail.lehigh.edu would go initially to the root level domain server. This would direct them to a Lehigh.edu name server (128.180.2.9), which would in turn, translate the mail.lehigh.edu subdomain to 128.180.3.218. This simple example ignores DNS caching / host files / proxy servers, etc. P. Mockapetris, *RFC 882: DOMAIN NAMES - CONCEPTS and FACILITIES* (Network Working Group, 1983). 3-5.

[43] Douglas Terry et al., *The Berkeley Internet Name Domain Server* (Berkeley, California: The Regents of the University of California, 1984).

freely available with the BSD version of UNIX.[44] The development was funded in part by the Defense Advanced Research Projects Agency, the renamed form of ARPA.[45] The defense agency had begun funding Berkeley's UNIX development as early as 1978 and had grown interested in the TCP/IP work being done by the CSRG team. DARPA had paid Bolt, Beranek and Newman (BBN) to write the original implementation of the TCP/IP protocol, but the software BBN produced was problematic. The BBN prototype was limited to 56KB/sec and utilized 100% of the computer's CPU. Berkeley graduate student Bill Joy re-wrote the code. Joy's version of TCP/IP required minimal CPU usage and increased performance to 700KB/sec. Over BBN's objections, Joy's version was adopted by DARPA.[46] The significant advances being made by the BSD team and the prevalence of their version of the UNIX operating system led DARPA to fund their efforts.

In addition to the BSD TCP/IP stack becoming DARPA's standard for the intranetworking protocol, the Free Software BIND server emerged as the de facto standard Domain Name System program on the Internet. Significant competition did not emerge until the Internet was made public in the early 1990s. For example, Microsoft did not release their own DNS server until 1996.Even then it was as a test component of Windows NT 3.1's Internet Information Services (IIS), which also included the company's initial forays into web hosting, TELNET, and FTP, but with all still running

---

[44] The software was free, released into the public domain, as the innovation predates the creation of the academic BSD license discussed previously. Later releases of the software were covered under free/Open Source licenses.

[45] The Advanced Research Projects Agency (ARPA) was renamed to the Defense Advanced Research Projects Agency (DARPA) in 1972.

[46] Peter H. Salus, *A Quarter Century of UNIX*, 161.

on the BSD TCP/IP stack.[47] According to the Internet Systems Consortium, the organization responsible for maintaining both the DNS standards and the BIND software, BIND has remained the most common DNS server on the Internet since they began surveying DNS server distribution in the 1990s. As of the latest publicly available survey data (2017), BIND accounted for 80.83% of the DNS servers on the Internet. Microsoft comes in at a distant second with 14.73% of the DNS hosting.[48]

The World Wide Web

In the July 1945 issue of *Atlantic* magazine, the Director of the Office of Scientific Research and Development, Dr. Vannevar Bush, published an article entitled "As We May Think." In the piece, Bush discussed emerging technologies such as microfilm, televisions, and facsimile machines. In considering the recording, processing, and storage of this information he determined that the growing amount of knowledge would require future researchers to have a more efficient way to store and associate data. Bush conjectured about the creation of a "Memex," a storage device in which "an individual stores all his books, records, and communications and which is mechanized so that it may be consulted with exceeding speed and flexibility." Rather than a traditional indexed storage like a file cabinet, this desk-like device stored information in a compressed format and linked together information in an associative manner. As a

[47] "Applies To: Internet Information Services 8.0 Microsoft Internet Information Services 8.0 Express," Microsoft.com, October 20, 2017, https://support.microsoft.com/en-us/help/224609/how-to-obtain-versions-of-internet-information-server-iis, (accessed October 29, 2018). Microsoft's use of the BSD TCP/IP stack is easily validated. From command line on a Windows NT or 2000 machine, simply type: findstr Regents c:\winnt\system32\ftp.exe (or nslookup.exe, etc.) and the program will return the BSD Copyright notice, "Copyright (c) 1983 The Regents of the University of California."

[48] "Domain Server Software Distribution," Internet Systems Consortium, February 7, 2017, https://ftp.isc.org/www/survey/reports/2017/01/fpdns.txt, (accessed October 29, 2018).

researcher found material relevant to his studies, he could link it together, making topical connections between the material.[49]

Vannevar Bush's microfilm-based system never came to fruition, but it inspired later innovators. In 1960, college student Ted Nelson envisioned Project Xanadu. Based on Bush's Memex, Xanadu was to be a database of all the world's literature. In Nelson's vision, readers would be able to have access to any material they would like, and the system would handle the royalty transactions. In describing the system's inter-linking between documents, Nelson coined the term *hypertext*, although Nelson's forward-thinking project was never completed.[50]

On the Philippine island of Samar in late 1945, a young Navy radio technician named Douglas Engelbart read Vannevar Bush's "As We May Think." He was awestruck by Bush's description of the Memex. Five years later, as he prepared for graduate school, Engelbart experienced a vision of what computers were to become. He pictured himself before a large computer screen, navigating between different symbols – an electronic workstation for organizing information – Bush's Memex, but rather than mechanical, he pictured it as an electronic computer.[51] Engelbart's electronic interpretation of the Memex informed his creation of the Graphical User Interface at the Stanford Research Institute's Augmentation Research Center in the 1960s, as discussed in Chapter 2.

---

[49] Vannevar Bush, "As We May Think," *The Atlantic*, July 1945, https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/. (accessed October 30, 2018).

[50] Howard Rheingold, *The Virtual Community: Homesteading on the Electronic Frontier*, rev. ed. (Cambridge, Massachusetts.: MIT Press, 2000), 100-101.

[51] Ibid., 7-9.

In 1989, these ideas behind the Memex and Project Xanadu informed the creation of the World Wide Web. That year, Tim Berners-Lee, a consultant at the Conseil Européen pour la Recherche Nucléaire (CERN), had a problem. He needed to facilitate collaboration between the researchers at CERN and their colleagues at scientific institutions throughout the world. To solve the issue, he created his own version of the Memex in a text-based platform with links that scientists around the world could use for distributed research. Berners-Lee and his associates at CERN named the software the World Wide Web, a reflection of the web of connections created by the content.

Berners-Lee's software had three components. The first was a basic command line-mode client that allowed workstations to browse websites through a series of linked menus. The second was the content formatting protocol, the Hyper Text Markup Language (HTML). The software, named after Ted Nelson's Xanadu's hypertext, allowed for formatting and linking text through tags within the content.[52] The third component was the CERN HTTPD (Hyper Text Transfer Protocol Daemon), the web software server.[53] Releases of the software were originally internal to CERN, but on August 6 1991, Berners-Lee announced his project release to the alt.hypertext news group and made the software available via FTP. The announcement described the project as merging "the techniques of information retrieval and hypertext to make an easy but

[52] The use of tags for formatting significantly predated the development of HTML. This approach was used in various forms since the mid-1960s. For example, RUNOFF, a program for CTSS, used .BR for line breaks – similar to the html tag
 used for line breaks. Later derivative programs, such as UNIX's NROFF, TROFF, and Hewlett-Packard's PCL, used similar approaches.

[53] The software was developed on a NeXT workstation running NeXTSTEP, an operating system derived from UNIX. In UNIX, programs that provide services are called *daemons*. These programs are frequently named to reflect this. An FTP server would be FTPD, an HTTP server, HTTPD, and so on.

powerful global information system...The project started with the philosophy that much academic information should be freely available to anyone." [54]

Tim Berners-Lee advocated for the platform to be freely usable by the public and on April 30, 1993, CERN published a statement detailing the release of the software, stating that "CERN relinquishes all intellectual property rights to this code, both source and binary and permission is given to anyone to use, duplicate, modify and distribute it."[55] The following year, Berners-Lee departed CERN for MIT where he created the World Wide Web Consortium, a group intended to provide oversight of web standards. With his departure, François Flückiger took over his technical development team at CERN. By this time, the CERN team had become aware of the risk of public domain software being appropriated by corporations. They discussed Richard Stallman's licensing protections with CERN's legal team and decided to release the next version of the platform as copyright-protected Free Software. Flückiger explained the decision to the web community in November 1994, stating, "The new versions will remain freely available… The only change is that the material distributed will remain copyrighted by CERN…. the rights of the users will be protected, in particular by preventing third parties to turn Free Software into proprietary software.... "[56]

In 1993, a team at the National Center for Supercomputing Applications (NCSA) at the University of Illinois developed their own version of CERN's web technologies. Led by staff member Marc Andreesen, the team created a web browser they named

[54] Tim Berners-Lee. "WorldWideWeb: Summary," August 6 1991. alt.hypertext. (accessed October 30, 2018).

[55] Tim Smith and François Flückiger, "Licensing the Web," CERN, March 12, 2014, http://home.web.cern.ch/topics/birth-web/licensing-web, (accessed October 30, 2018).

[56] Ibid.

Mosaic. The new platform allowed color images to be embedded in web pages and permitted pictures to serve as hyperlinks. Where CERN's software had been purely text based; Mosaic made the World Wide Web graphical. The NCSA's web browser was made freely available to the public in November 1993. By the following spring, over a million copies were in use. That same year, Andreesen and much of his team left the NCSA to launch Netscape, a commercial browser company.[57] The following year, Microsoft licensed the Mosaic technology for use in its Internet Explorer 1.0 product for Windows 95.[58]

The Mosaic browser was just one part of the platform produced by the group at the University of Illinois. NCSA staffer Rob McCool also created an HTTP server that was shared in the public domain. McCool joined Netscape in 1994, and following his departure, the NCSA's development on the HTTP application stalled. This lack of progress became problematic as the software was popular, and by February 1995, the NCSA HTTPD platform was the most common web server on the Internet. To address this lack of support, several webmasters began to write and share their own extensions and patches. Two of these programmers, Brian Behlendorf and Cliff Skolnick, coordinated the development efforts via a mailing list. Using NCSA HTTPD version 1.3 as a departure point, the group assembled their patches and performance enhancements into a Free Software project. They named the program the Apache server, a tongue-in-cheek reference to the patched-together nature of the software, "a patchy server."[59] On

---

[57] Turner, 213-214.

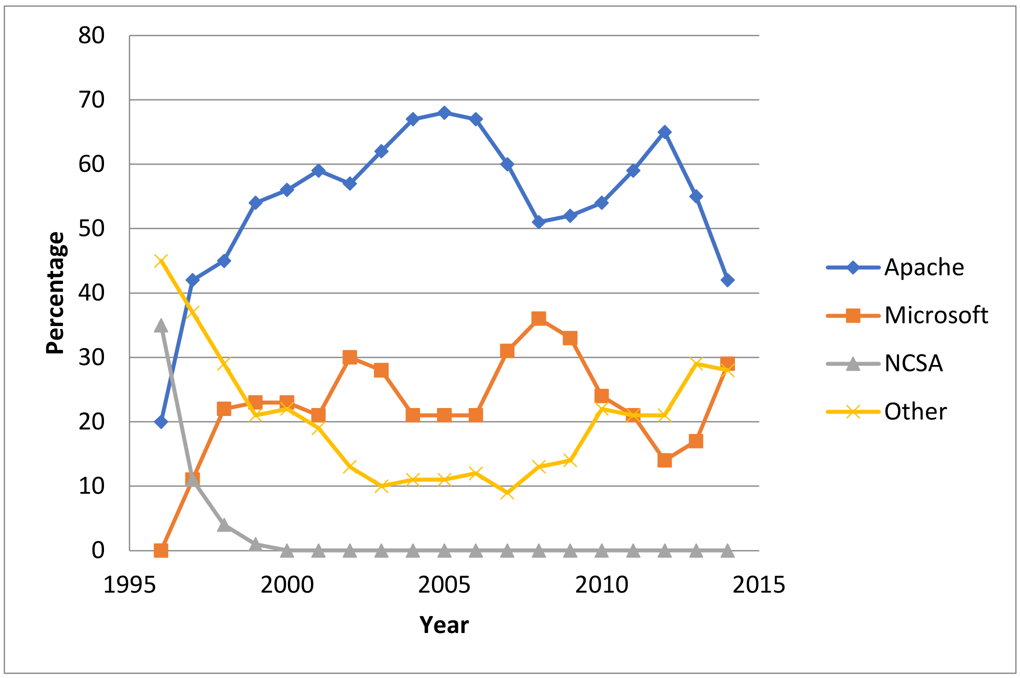
[58] "NCSA Mosaic," National Center for Supercomputing Applications, http://www.ncsa.illinois.edu/enabling/mosaic, (accessed October 30, 2018).

[59] "About Apache," Apache.org, http://httpd.apache.org/ABOUT_APACHE.html, (accessed October 30, 2018).

December 4, 1995, the group announced the availability of Apache 1.0 to the comp.infosystems.www.servers.unix newsgroup.[60] The web server was released under a custom Free Software Apache License derived from the four-clause BSD license discussed in Chapter 4.[61]

In April 1996, Apache surpassed its progenitor and emerged as the primary Internet web server according to statistics gathered and maintained by the Netcraft company. Netcraft, an Internet services company from Bath, England, has collected Internet market share data since 1995. As seen in Table 5, the Free Software Apache server played a seminal role in the growth of the World Wide Web.[62]

Table 5: Web Server Market Share by Year



[60] Randy Terbush. "Announcing: Apache-1.0 Release", December 4, 1995. comp.infosystems.www.servers.unix. (accessed November 05, 2018).

[61] *Apache License, Version 1.0* (The Apache Group, 1995).

[62] "April 2014 Web Server Survey," Netcraft, April 2, 2014, http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html,(accessed March 14, 2015).

Source: Adapted from Netcraft. "April 2014 Web Server Survey." April 2, 2014. Accessed March 14, 2015. http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html. Graph 1.

The Internet Tidal Wave

The Internet dominance of UNIX-based Free Software did not go unnoticed by its commercial competition. On May 26, 1995, Microsoft CEO Bill Gates issued a memo to the company's executive staff and department heads. Entitled *The Internet Tidal Wave*, the document exhorted Microsoft's management to understand the significance of the Internet, stating that "The Internet is the most important single development to come along since the IBM PC was introduced in 1981. It is even more important than the arrival of the graphical user interface (GUI)." Gates foresaw that soon nearly all PCs would be connected to the network and would be dependent on Internet-based resources. The challenge, according to Gates, was that the competition was benefitting from the Internet and Microsoft was not. "The default server is still a UNIX box and not Windows NT."[63] Gates remarked on SUN Microsystem's effectiveness in exploiting the situation to their benefit.[64] The memo notes that BSD UNIX had full Internet server integration and he encouraged his team to understand how to advance Windows NT to compete with this.[65]

Gates' concern with Microsoft's adoption of the Internet was well founded. In the 1980s Microsoft was focused on their DOS and Windows platforms. The company did

---

[63] William Gates, "The Internet Title Wave Memo," May 1995, Department of Justice, http://www.justice.gov/atr/cases/exhibits/20.pdf, (accessed October 30, 2018), 1-3.

[64] SUN Microsystems was founded by Bill Joy and was based on his work with the BSD project. Joy's heavy involvement in developing TCP/IP put SUN in a unique position to embrace the emerging Internet.

[65] Gates, "The Internet Title Wave Memo," 5.

not release Internet-capable networking until the summer of 1993 when Windows NT 3.1 and Windows 3.11 (Windows for Workgroups) provided TCP/IP networking functionality. Prior to this, third-party tools were required for Internet connectivity. Even Microsoft's own website, first launched in August 1994, relied on free web-hosting software, the University of Edinburgh's European Microsoft Windows NT Academic Centre (EMWAC) webserver.[66] At the time the memo was written, Microsoft had yet to release their own browser or web server. Internet Explorer was planned to be included in the Plus! add on for the yet-to-be released Windows 95, and the company's first web server, Internet Information Server (ISS), was still in testing.[67]

In addition to encouraging the adoption of the Internet, the memo also documented Microsoft's drive to undercut competitors, such as Netscape, by commoditizing the web through proprietary technology. In the document, Gates discusses ways to "get people to switch away from Netscape," namely to integrate the web browser within the operating system. "We need to move all of our Internet value added from the Plus pack into Windows 95 itself," Gates wrote. "Over time the shell and the browser will converge and support hierarchical/list/query viewing as well as document with links viewing." For web-based documents, Gates was determined that Microsoft establish proprietary standards based on windows programming protocols. "I am sure the OpenDoc consortium will try and block this," Gates adds.[68]

---

[66] Richard Raucci, *A Windows NT Guide to the Web* (New York, New York: Springer, 1997), 37.

[67] Windows 95 was released in August 1995, and ISS was not made available until its inclusion that November in Windows NT 3.51 Service Pack Three.

[68] Gates, "The Internet Title Wave Memo," 5.

Just as Bill Gates's 1976 *Letter to the Hobbyists* revealed his intention to commoditize software through code scarcity, *The Internet Tidal Wave* reveals Gates's intent to expand Microsoft's dominance into the Internet space through proprietary standards and anti-competitive behavior. It is for this reason that the document eventually became public; in 2001, the memo was an exhibit in the government's anti-trust suit against Microsoft.

Symbiosis

By the time the Internet became commercialized in the mid-1990s, a symbiotic relationship between Free Software and the Internet had already emerged. The ARPAnet evolved using the open schema common to the era. Software was shared, re-written, and redistributed. TCP/IP, the networking protocol that allowed the ARPAnet to transform into the Internet was originally written by BBN – before being re-written and distributed by UCLA's Free Software developer Bill Joy. Joy's Free Software version of TCP/IP was incorporated into many systems, including Microsoft Windows. The BIND DNS server, developed on Berkeley's Free Software UNIX distribution, BSD, has been the dominant provider of Internet name resolution since its introduction. The World Wide Web itself was Free Software, and more often than not, the pages on the web were served by the Free Software web server, Apache. Whether they knew it or not, World Wide Web users were operating Free Software.

Just as Free Software supported the Internet, the network fostered the Free Software Movement. The growing speeds and ubiquity of the Internet allowed for easy collaboration and distribution of programs. It also changed how Free Software projects were performed, allowing for the decentralization of project tasks and the rapid development of alternative technologies. This relationship can be seen in one of the Free Software Movements most well-known products, the Linux operating system.

Linux

Linus Torvalds

The Linux operating system had an unlikely beginning as an educational exercise. Its creator, Linus Torvalds grew up in Rödbergen, a small neighborhood in Helsinki, Finland. Torvalds had an early introduction to programming through his grandfather, Leo Waldemar Tönqvist, who provided the young boy with a Commodore VIC-20 computer. Torvalds learned BASIC on the VIC-20 before moving on to learn assembly language on a Sinclair QL computer. He originally used the machine to play the type-in video games that were commonly shared in the early home computing era.[69] Frustrated with the limitations of what was provided with the device, Torvalds began to write his own software for it.[70]

In the spring of 1990, Linus Torvalds completed his compulsory service in the Finnish military, and he enrolled for courses that fall at the University of Helsinki. In anticipation of his scheduled classes on UNIX and the C programming language, Torvalds began to read *Operating Systems: Design and Implementation* by Andrew S. Tanenbaum. As a faculty member at the Vrije University in Amsterdam, the UCLA-

---

[69] Both computers had read-only operating systems stored on ROMS. These are inherently unchangeable, so to expand the system, such as to add hardware that was not included originally, software would have to be written and the data in the ROMS flashed (over-written.)

[70] Linus Torvalds and David Diamond, *Just for Fun: The Story of an Accidental Revolutionary*, Reprint ed. (New York, New York: Harper Business, 2002), 14-16,40-44.

trained Tanenbaum had written a small UNIX clone named Minix. Tanenbaum's text demonstrated Operating System design through exploration of the Minix software.[71] Fascinated by the elegance inherent to the design of UNIX, Torvalds immersed himself in the textbook and understanding the fundamentals of the operating system.

In early 1991, Torvalds attended a talk given by Richard Stallman at the nearby Polytechnic University of Helsinki. In his semi-biographical text, *Just for Fun: The Story of an Accidental Revolutionary,* Torvalds recalled this as his first exposure to Stallman and the Free Software Movement, "I have to admit that I wasn't much aware of the sociopolitical issues that were – and are – so dear to RMS [Stallman]. I was not really all that aware of the Free Software Foundation, which he founded, and all that it stood for…." Torvalds's interest in attending the talk was driven by the technology, not his understanding of Stallman's ideology, but "something from his speech must have sunk in," Torvalds recalled, since he "ended up using the GPL for Linux."[72]

In the months following the presentation, Linus Torvalds found himself writing software to address Minix's short comings. Andrew Tanenbaum's version of UNIX was intended as an educational tool, not an end-user system, and as a result, it was lacking in key areas.[73] Torvalds's efforts to address the issues soon expanded to using GNU tools to re-write parts of the Minix operating system. By July 1991, Torvalds was seeking information about the POSIX standards, the Institute of Electrical and Electronics

[71] Torvalds and Diamond, 50-54.
[72] Ibid., 58-59
[73] Job controls refers to the ability to set running programs into the background or altering the system processor affinity for the job (called nice/renice within UNIX.) Terminal emulators were programs capable of emulating a hardware terminal that would be connected to a central computer. (E.g. A terminal emulator would allow a PC on a LAN to emulate a Wyse WY-50 terminal that would have been connected to a minicomputer via an RS232 serial port.)

Engineers' guidelines for compatibility between UNIX and other operating systems.[74] Torvalds began to reverse-engineer UNIX system functions by referencing manuals for Sun Microsystems' UNIX.[75] What had begun as a project to address the shortcomings of Tanenbaum's Minix began to grow into a full replacement.

A New Operating System Emerges

On August 25, 1991, Linus Torvalds announced his intentions in the comp.os.inix news group. "I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386 (486) AT clones." Torvalds wrote, "I'd like any feedback on things people like/dislike in Minix, as my OS resembles it somewhat…" Torvalds reported that he had already ported GNU software such as the Bourne-again shell (BASH) and the C compiler (GCC). His operating system was not yet available to the public, but Torvalds said he anticipated having something available within months.[76] He released an early, crippled version of the software, named "Freax," on September 17, 1991.[77] Weeks later on October 5, Torvalds revised the operating system and announced the availability of the software - now named Linux - free and with source code, available via FTP from the University of Helsinki.[78]

A community of enthusiastic programmers began to use the Linux operating system within weeks of its release. They began to test the software, and they shared

---

[74] Linus Torvalds,"Gcc-1.40 and a Posix question.", Email to comp.os.minix mailing list. July 3, 1991.

[75] Torvalds and Diamond, 79.

[76] Linus Torvalds, "What would you like to see most in minix?" Email to comp.os.minix mailing list. August 25, 1991.

[77] Torvalds and Diamond, 87-89.

[78] Linus Torvalds, "Free minix-like kernel sources for 386-AT". Email to comp.os.minix mailing list. October 5, 1991.

issues and functionality requests with Linus Torvalds. As the young student worked to address issues with the core of the operating system, the kernel, the programmers rapidly began to port software to the new platform. Instead of a planned, centralized project like GNU, the group simply created a mailing list, linux-activists@joker.cs.hut.fi, to track their distributed development efforts, and they began to re-write software for the new operating system. In the first two months, six programs were ported.[79] A week later, 15 programs had been migrated to the new platform.[80] As word spread of the new operating system, growing numbers of interested programmers began to join the mailing list and contribute. By January 1992, developers throughout the world were writing and sharing contributions to the new operating system. At Temple University in Philadelphia, programmer Jim Wiegand wrote printer control software.[81] The University of Miami's Yanek Martinson developed the log on and startup processes for Linux.[82] U.S. Navy system administrator Lieutenant Commander Michael Dobson re-wrote the Unix to Unix copying software, UUCP, for the new platform.[83] At Clarkson University in Potsdam, New York, Russ Nelson began work on Linux networking - before even joining the development mailing list.[84] Programmers from academia were particularly fruitful in these early efforts. Developers at the University of Western Australia ported GNU tools,

---

[79] Robert Blum. "The latest TODO list". Email to linux-activists@joker.cs.hut.fi. November 25, 1991.

[80] Robert Blum. "The new TODO file". Email to linux-activists@joker.cs.hut.fi. December 6, 1991.

[81] Jim Wiegand. "lp support & daemonology". Email to linux-activists@joker.cs.hut.fi. January 7, 1992.

[82] Yanek Martinson. "lpdemon, init, shutdown, uucp". Email to linux-activists@joker.cs.hut.fi. January 7, 1992.

[83] Michael Dobson. "Re: lpdemon, init, shutdown, uucp". Email to linux-activists@joker.cs.hut.fi. January 8, 1992.

[84] Russ Nelson. "Networking.". Email to linux-activists@joker.cs.hut.fi. January 9, 1992.

as did the team at Berkeley, who also ported the BSD TCP/IP code. At MIT, they developed file systems for Linux.[85]

Distributions

MIT and other institutions began to set up new FTP sites, called mirrors, to share Linux and the Free Software being created to run on it.[86] For those who did not have ready access to the Internet, MIT's Theodore T'so began to offer a floppy disk-based distribution of Linux. T'so would send copies of the Linux software and associated utilities via the postal service for a distribution fee of $7.50 per diskette U.S. ($10 per diskette international.) The fee covered the disk media, time to produce the copies, and postal fees.[87] Since the operating system had been released under Stallman's GPL license, this type of distribution for profit was possible as long as the source code would be provided upon request. Within months, similar distribution services began to emerge, and by the following year, the first full Linux Distributions or "Distros" -- specific sets of Linux software programs -- began to appear. The first CD ROM based distribution was Yggdrasil Linux, offered by Berkeley California programmer Adam Richter. Introduced on November 25, 1992, Yggdrasil Linux cost $99, although it was complimentary for any of the principal maintainers of the software.[88] The CD-based distribution offered no support, but it helped make Linux available to users outside of the highly-technical news groups and mailing lists where it was being developed. The software was now packaged

---

[85] Robert Blum. "The new TODO file". Email to linux-activists@joker.cs.hut.fi. December 6, 1991.

[86] Robert Blum. "The latest TODO list".. Email to linux-activists@joker.cs.hut.fi. November 25, 1991.

[87] Theodore T'so. "Re: No FTP ... HELP!". Email to linux-activists@joker.cs.hut.fi. January 20, 1992.

[88] Adam Richter. "ANNOUNCEMENT: Alpha release Linux/GNU/X unix clone on CDROM for PC's". Email to alt.os.linux,comp.unix.pc-clone.32bit,comp.unix.wizards mailing lists. November 25, 1992.

and able to be sold in stores like its commercial cousins, providing non-technical consumers access to the software.

With the introduction of Linux distributions, the community began to ponder the Linux project's end goals. In a December 1992 discussion, University of Chicago graduate student Richard Goerwitz pondered, "Can Linux go on forever without some commercial support?" Goerwitz's query went to a fundamental questioning of the tenability of Free Software. "It's hard to see anything as useful that's maintained by folks that aren't doing it for a living…. Is there some way that a useful remuneration path can be arranged?"[89] The responses Goerwitz received provide insight into the ideologies and motivations of the Linux programmers at this time. Jonathan Magid from the University of North Carolina provided a terse response explaining the nature of the GPL licensing and how the remuneration path was the free market, explaining that "Linux (the kernel and most of the main utilities including the compiler, are copy-lefted. Anyone can give them away or sell them...."[90] At Minnesota's St. Olaf University, Michael Johnson detailed his own motivations. "Many of us who work on Linux do so, not with any thought of remuneration, but because it is interesting and rewarding in and of itself. I did not write the improved printer driver, for instance, because someone paid me for the many hours I spent, but because I learned from it."[91] Johnson's perspective mirrored Richard Stallman's intended uses of the software. Peter MacDonald, founder of the Softlanding Linux System (SLS) distribution echoed Stallman's concerns with

[89] Richard Goerwitz. "MONEY + Linux". Email to linux-activists@joker.cs.hut.fi. December 17, 1992.

[90] Jonathan Magid. "Re: MONEY + Linux". Email to linux-activists@joker.cs.hut.fi. December 17, 1992.

[91] Michael Johnson. "Re: MONEY + Linux". Email to linux-activists@joker.cs.hut.fi. December 17, 1992.

commercial software as a driving factor for Linux.[92] “Commercial software typically has one extremely unpleasant side effect,” Macdonald explained, citing the influence of non-technical corporate groups such as marketing, who “come to control so much of the technical direction, that as the organization grows to a size where it can start to do impressive things, it becomes increasingly paralyzed. And the products they deliver become increasingly disjointed.” MacDonald reinforced this with personal experience, asking rhetorically, “Have you ever spent longer trying to get some vendors solution installed and working on your site, than you would have just developing the thing from scratch yourself? MacDonald also makes another observation about the changes occurring in the computing industry at the time, “the growth of Linux is paralleling the rapid growth in the internet. That explains its sudden appearance, and warrents [SIC] that the ‘free support’ aspect can continue, indefinitely.”[93]

Richard Goerwitz’s inquiry was timely; just as he was posing his questions to the Free Software adherents, others were preparing their own models for support and remuneration. At the time of the discussion, programmer Patrick Volkerding was already working on developing his own distribution of Linux, Slackware. Based on MacDonald’s SLS collection, Slackware was the first commercially supported Linux distribution. Version 1 was announced on July 17, 1993.[94] Founded in late 1992, the SuSE Linux company began work on their own variant of Volkerding’s Slackware distribution. In

---

[92] Released in August 1992, Peter MacDonald’s Softlanding Linux System was one of the first Linux distributions, predated only by Owen Le Blancs’ Manchester Computing Center (MCC) release. For the purposes of clarity, the Yggdrasil distribution detailed here is the first CD-based commercial distribution.

[93] Peter MacDonald. "Re: MONEY + Linux". Email to linux-activists@joker.cs.hut.fi. December 18, 1992.

[94] Patrick Volkerding. "ANNOUNCE: Slackware Linux 1.00". Email to linux-activists@joker.cs.hut.fi. July 17, 1993.

1993, Red Hat Linux was founded, and their initial Linux release was the following fall.[95] Other commercial distributions followed, with competitors such as Caldera, Mandrake, and TurboLinux all evolving from Red Hat's version of Linux. The operating system now had many options for software support, but Linux was still not supported by mainstream computer hardware vendors like Dell, IBM, and HP. Users of Linux discovered that if they ran Linux, an unsupported OS, the hardware vendors would refuse to support the hardware. Naturally, capitalism filled this void as well. The VA Research corporation (later renamed VA Linux) was founded in 1993 to provide computers pre-installed with a version of Linux that VA would fully support.[96]

GNU / Linux

Linux emerged at a key time for the Free Software Movement. The personal computer had become prevalent, but there was not yet a Free Software version of UNIX for the platform.[97] The CSRG team had extricated all the AT&T code from BSD and released a free version of the software in 1989, but it did not run on the PC architecture. The GNU project had been working on their own kernel. However, by 1990, they found the development process to be both slow and problematic, so they elected to await the development of the "Mach" kernel, a BSD-derivative being developed at Carnegie

[95] Thomas Chung, "History of Red Hat Linux," Fedora Project, last modified November 6, 2016, https://fedoraproject.org/wiki/History_of_Red_Hat_Linux. (accessed November 7, 2018).

[96] "Larry Augustin," Epic Ventures, http://www.epicvc.com/larry-augustin/, (accessed November 7, 2018). Where it is possible for computers to run third party software without hardware support issues, this does not extend to operating systems, which are responsible for direct interaction with the hardware. As such, hardware vendors provide specific operating system version that they are willing to support. These are OS platforms for which the vendor has composed software "drivers" for their hardware and are able to reliably run hardware diagnostics on. The limitations of support are a technical rather than ideological or intellectual property concerns. In order to take responsibility for a configuration, the hardware must be able to diagnose if a user issue is occurring due to their hardware or due to an OS issue. As a result, hardware manufacturers limit the scope of the operating systems they will support.

[97] There were several corporate versions of the UNIX available for the PC such as Coherent, Microsoft Xenix, and of course, Minix.

Mellon University.[98] The GNU team had also considered using Andrew Tanenbaum's Minix, but Stallman dismissed the idea over concerns that Tanenbaum's software was not powerful enough, and more importantly, the licensing was too restrictive. "We want computer manufacturers to be permitted to distribute the entire GNU system. And even to charge money for doing so (as long as recipients get the right to redistribute). This is not allowed for Minix."[99]

Without a kernel, the GNU project software needed to be run on a commercial UNIX system. For applications like the GNU C compiler (GCC), this afforded a financial benefit to users who did not need to purchase a commercial UNIX C compiler. But for other applications, such as system utilities or shells, the use of the software was purely ideological as "closed" versions of the same programs existed already within UNIX.

Linux changed everything. Now users of PCs could run completely Free Software on a completely free operating system. Linux distributions were also now available both pre-packaged in stores and downloadable off the Internet. Instead of needing to understand the intricacies of computing to use the software, curious home users could just load the CDROM and try out the operating system.[100] This brought the GNU project and other Free Software from the realm of highly-technical ideologues and made it accessible to the general public.

---

[98] Richard Stallman. "GNU kernel status". Email to gnu.announce mailing list. November 11, 1990.

[99] Richard Stallman. "GNU and Minix". Email to comp.os.minix mailing list. October 17, 1990.

[100] Whether acquired in a store or via the Internet, users could use a CDROM or .iso image to create a CDROM and install the operating system. Most of the Linux software installation programs could re-arrange a PC's drive partitions to accommodate dual booting, an approach that allows users to alternate between Linux and the previously installed operating system. By the mid-1990s, PC virtualization programs such as BOCHS began to emerge. These allowed users to examine the alternate operating system without affecting their installed operating system.

The popularity of Linux became so great that Richard Stallman soon grew worried that the ideological objectives of the GNU project were being overshadowed by the operating system. A big part of his unease was in the observation that “Linux” was becoming a synecdoche for Free Software. This oversimplification ignored the decade of programming that preceded the Linux kernel. “The heart of the GNU project is an idea: that software should be free, and that the users' freedom is worth defending,” Stallman explained. “The GNU project's method is that Free Software and the idea of users' freedom support each other. We develop GNU software, and as people encounter GNU programs or the GNU system and start to use them, they also think about the GNU idea.” Stallman continued, “This method was working well--until someone combined the Linux kernel with the GNU system (which still lacked a kernel) and called the combination a ‘Linux system.’” The issue, as Stallman saw it, was that although “adding Linux to the GNU system brought the system to completion,” the practice of referring to the collective system as just Linux undermined the core method of communicating the Free Software ideology.[101]

Although Linux was released under his own GPL license, Stallman was concerned that the growing public embrace of Linux would separate it from the ideas behind Free Software. To address the problem, he recommended several alternative names for the Linux systems using the GNU packages, such as "Linux-based GNU system," "GNU/Linux,” or "Lignux."[102] None of these gained traction outside the most

[101] Leonard Tower. "Linux and the GNU system by Richard Stallman". Email to gnu.misc.discuss,comp.os.linux.misc,comp.os.linux.development.apps,comp.os.linux.advocacy mailing lists. May 31, 1996.

[102] Ibid.

fundamentalist Free Software ideologues. Stallman's intention was to change society through technology. This only worked if the technology stayed on message.

The Community

In addition to adopting the Linux operating system, a growing Free Software community formed around the Internet-based structures they used to support activities such as code sharing, project coordination, and communication. The social interactions that grew around these mechanisms also fostered the further adoption and spread of Free Software. Some of the community creations, like software versioning systems, were merely technical. Others, like Linux user groups and websites such as Slashdot were social and cultural constructs that reinforced the community and its structures.[103]

Software Versioning Systems

With Stallman's GNU project, the planning and communication for the coding efforts were either handled by him, or one of the Free Software Foundation's employees. With earlier projects such as the EMACS editor, Stallman was the ultimate arbiter of the changes submitted for the software. However, with the Internet countless programmers could be working on the source code simultaneously. As a result, efforts to reconcile source code changes for one given program – much less an entire project – could be a herculean task. The solution was Internet-based software versioning systems, source code

---

[103] This section is intended to characterize the Free Software community, context that is required to understand the changes detailed in the coming chapters. This is not intended to be an anthropological assessment of the culture. Anthropologist Chris Kelty provides an in-depth theoretical assessment of the "geeks" in the Free Software community in his monograph*Two Bits: The Cultural Significance of Free Software, Experimental Futures (Durham, North Carolina: Duke University Press, 2008),*Unfortunately, Kelty provides an incomplete account of the history of the Free Software Movement, which he holds did not begin until after 1998. As a result, many of the people and events prior to this point are trivialized within Kelty's analysis.

repositories that allowed developers to work independently on their programming projects and then "check-in" or "commit" their modifications. The software versioning system would maintain a history of the source code changes as well as an audit log of who submitted the software. The first Free Software versioning system was implemented for the GNU project in the early 1980s by team member and Purdue University computer scientist Walter Tichy.[104] His software, the Revision Control System (RCS), was based on similar software, the Source Code Control System (SCCS), which had been developed for UNIX by Bell Labs staff member Marc Rochkind in 1972.[105] Both RCS and SCCS were designed for submissions to be done as local, logged-on users. To adopt the software to the indirect nature of Internet collaboration, Dutch computer scientist Richard Grune invented the Concurrent Versions System (CVS), that essentially acted as a front end for the earlier Revision Control System.[106] The centralized management capabilities of CVS and its derivative projects allowed for Internet-based distributed development of software.

Virtual Communities

Coordination on projects like GNU and Linux initially occurred via email lists and newsgroups. However, the growth of the Internet meant that more networks had perpetual Internet connections. This connectivity gave rise to the adoption of real-time Internet-based discussions in Internet Relay Chat (IRC) "chat rooms." Derived from the

---

[104] Walter Tichy, "RCS—a System for Version Control" (paper presented at the IEEE Computer Society 6th international conference on Software engineering, Los Alamitos, California, September 13-16, 1982). 1-2.

[105] Marc Rochkind, "The Source Code Control System," *IEEE Transactions on Software Engineering* 1, no. 4 (December 1975): 364-70.

[106] Dick Grune, "Concurrent Versions System CVS," VU University Amsterdam, https://dickgrune.com/Programs/CVS.orig/, (accessed November 7, 2018).

topically focused news groups on Bulletin Board Systems (BBS), IRC channels or "rooms" were focused on a single subject, such as kernel development. Massive IRC networks such as EFnet, Undernet, and DALnet emerged to support the global Free Software community's real-time interactions.[107] IRC provided developers with a platform for instantaneous collaboration on projects. For users, the channels provided a venue for real-time technical support. And for the community, Internet Relay Chat provided a virtual town center, a space where the Free Software public could gather to work, socialize, and come together as a virtual community.

The Free Software community also began to gather around various websites. Some were tech-focused counterparts to traditional publications, such as the Linux Journal, which was founded in 1994. Another website, Linux.org served as a low-level starting place to introduce new people to the operating system. The separate, but similarly named Linux.com provided news, user group directories, and higher-level technical content about the Linux operating system. By 1997, one website in particular had emerged as central to the Free Software community. Called Slashdot, the website provided "news for nerds, stuff that matters." The website linked to articles posted throughout the web and featured discussion threads for readers to comment on what they had read. Based on an earlier effort named "Chips and Dips," by Hope College student Rob Malda, the website established a large following immediately upon its release. "At first it had just a few thousand daily readers migrating over from Chips & Dips, but in a

---

[107] Free software communities were resident throughout all major IRC networks, but by the late 1990s, most of the Free Software groups were focused around the Open Projects Net (OPN), which was dedicated to serving these communities. The centralization and introduction of advertising services such as Fresh meat will be discussed in more detail in Chapter 6.

matter of weeks it had grown so fast," Malda recalled of the website's start.[108] In a 2007 Slashdot article detailing the origin of the site, the programmer recalled his surprise as the web page grew in popularity. "My world was rocked over and over again as I watched the domain names... mit.com! ibm.com! redhat.com! Hell, even microsoft.com."[109] Author and Free Software advocate Russell Pavlicek described the website as "the water cooler of geek culture," a website whose animated discussions reveal "the current issues which are important to the community."[110] The site was so popular, and its influence so great, that when Slashdot linked to other websites, the ensuing traffic from Slashdot's readers would frequently overload and crash the linked site. This "Slashdot effect" was such a common occurrence that automatic webpage mirroring technologies were developed to mitigate the issue.[111]

Evangelism

Although later associations were chiefly online, the Free Software community also interacted in person at user group meetings. Formed prior to the commercial Internet, groups known as LUGs (Linux User Groups) convened throughout the world to discuss Linux and other Free Software such as the BSD UNIX derivatives. Group meetings provided a venue for sharing software and helping others learn about it. In May 1993,

---

108 Rob Malda, "A Brief History of Slashdot Part 1, Chips," Slashdot, October 3, 2007, https://news.slashdot.org/story/07/10/02/1553218/a-brief-history-of-slashdot-part-1-chips-dips, (accessed November 7, 2018).

109 Ibid.

110 Russell C. Pavlicek, *Embracing Insanity: Open Source Software Development* (Indianapolis, Indiana: Sams, 2000),119.

111 Solution for Slashdot Effect?, *Wired Magazine*, October 1, 2004, https://www.wired.com/2004/10/solution-for-slashdot-effect/,(accessed November 7, 2018).

Australia's first Linux User Group met in Sydney.[112] A year later, the Amateur Computer Group of New Jersey started their own LUG named LUNICS SIG, (a name reflecting the group's composition of LINUX, and other systems.)[113] In Switzerland, the first LINUX User Group Switzerland (LUGS) meeting occurred on June 16, 1994.[114] That same summer, other Linux user groups began to meet in Washington, D.C., Phoenix, and Salt Lake City.[115] In addition to the monthly LUG meetings, the groups frequently scheduled events called "Install Fests." Part workshop and part tent revival, the events were designed to aid the community's less technical members in the installation and troubleshooting of Free Software such as Linux.

Within the computing industry, some individuals emerged as full-time "evangelists" for Linux and Free Software. For example, in 1997, Compaq Computers appointed Free Software advocate Russell Pavlicek to be the company's "Linux Evangelist."[116] Like their theological counterparts, Free Software evangelists like Pavlicek travelled extensively and spread the good news of Linux and Free Software at client sites, trade conferences, and user group meetings. To be clear, the idea of software evangelism was nothing new. In his popular 1990 memoir, *The Macintosh Way*, author Guy Kawasaki recounted his experience as an evangelist at Apple Computers during the

---

[112] Robert Thomas. "(Australia) Sydney Linux users Group". Email to comp.os.linux.announce mailing list. May 13, 1993.

[113] Peter Fillingham. "NJ computer user's group Linux-related SIG". Email to comp.os.linux.announce mailing list. May 22, 1994.

[114] Phillip Markwalder. "FIRST MEETING LINUX User Group Switzerland (LUGS)". Email to comp.os.linux.announce mailing list. June 10, 1994.

[115] Przemek Klosowski. "First meeting of metropolitan DC Linux User group". Email to comp.os.linux.announce mailing list. June 7, 1994.; Rusty Carruth. "Phoenix Linux User's Group". Email to comp.os.linux.announce mailing list. August 6, 1994.; Vir Lagua. "Salt Lake Linux Users Group". Email to comp.os.linux.announce mailing list. July 17, 1994.

[116] Russell C. Pavlicek, *Embracing Insanity: Open Source Software Development.* About the Author.

1980s.[117] But in Free Software, this role was more than a job; it was about being a fervent believer among a community of the faithful.

The Welcome Wagon

The community, as it was in the early 1990s, is perhaps best understood through the writings of its members. In a 1999 retrospective article entitled "The Welcome Wagon," columnist and community member Emmett Plant discussed the openness and spirit of cooperation that he felt when introduced to the community.[118] The article opened with Plant recalling a childhood memory, the awkwardness of the first day at a new elementary school, and the comfort he felt after another pupil approached him saying, "Hey there! I'm the welcome wagon! What's your name?"[119] Plant associated this encouraging introduction with the inherent diplomacy of the Free Software culture. "The Linux community," Plant wrote," is a really great group of talented individuals…. They know that for Linux to succeed, it will take a strong sense of community and a helpful attitude from its users."[120] Plant then provided rhetorical examples of helpful community members, such as Stallman being willing to aid a neophyte user with disk partitioning.

Beyond the technical assistance freely given among community members, Plant reported a sense of something magical about the Linux community. He told the story of having a beer with Linus Torvalds, and he contrasted the accessibility and sense of

---

[117] Guy Kawasaki, *The Macintosh Way* (Glenview, Illinois.: Scott Foresman Trade, 1990).
[118] Emmett Plant, at the time of this article, was a columnist for Linux Today. This article resonated with the Free Software community and later helped him become a writer for the Slashdot website and following that, editor of Linux.com.
[119] Emmett Plant, "The Welcome Wagon," Linux Today, November 9, 1999, https://www.linuxtoday.com/developer/1999110900105NWSM,(accessed November 7, 2018).
[120] Ibid.

kinship within the community to the corporate computing world. As put by Plant's friend Clyde Williamson, "What are the chances you'll be able to drink beer with Steve Ballmer anytime soon, Emmett?"[121] Plant considered Williamson's question and concluded that even if he could communicate with Microsoft President Steve Ballmer, he would not deign to respond. [122] Plant contrasted this to the Free Software community where members such as developer Tom Christensen responded to community members immediately.[123]

Plant's recollection ended by entreating the community to sustain their traditional values, to continue to have Linux User Group meetings and install-fests, and to enthusiastically greet others who had not yet experienced Free Software and offer to help them join the community. Plant's article was designed to implore the Free Software community to maintain civility and support each other. As such, the article was subjective and deliberately emphasized the community bonds at a point where these bonds were starting to strain.[124] However, Plant's deliberateness in highlighting these aspects of the community did not diminish their validity.

---

[121] Ibid.

[122] In the interest of being forthright, the author noted that he worked with both Emmett Plant and Clyde Williamson on several web initiatives following this period, including work on several Free Software websites such as Linux.com. Williamson and the author also co-created a Douglas Adams memorial project that persists today. (http://www.towelday.org)

[123] Emmett Plant, "The Welcome Wagon."

[124] Plant's article was written in 1999, when the Free Software Movement was struggling to adjust to the growing corporatization within the movement. At the time of the article, the corporate-friendly Open Source re-articulation had recently started a schism within the movement. Plant was reacting to this by encouraging the preservation of the community values as they had existed thus far. This will be discussed in detail in the next chapter.

Debian Social Contract

Where Emmett Plant's "The Welcome Wagon" provided a qualitative recollection of the community, the "Debian Social Contract" was a deliberate statement that defined how the Free Software Debian project would interact with the community and society as a whole. The Debian project launched in 1993 with the intention of creating a new Linux distribution "developed openly in the spirit of Linux and GNU." Programmer Ian Murdock believed the emerging corporate Linux distributions prioritized sales over the quality of their software. Murdock also felt that as corporate interest in Free Software grew, the ideological basis of the movement would be lost, noting in 1994 that already there was a "tendency not to mention that Linux is free nor that it is distributed under the GNU General Public License." Murdock's solution was to create the Debian Linux distribution, designed expressly to be consistent with Free Software's core ideological beliefs. When completed, Debian would be made freely available on the Internet and also distributed on CDROM by the Free Software Foundation.[125]

In 1996, Ian Murdock left the Debian project. The new project leader, Pixar Animation programmer Bruce Perens re-iterated the Debian team's commitment to the Free Software community through the creation of the "Debian Social Contract with the Free Software Community." The contract itself was based on five promises: First, Debian would remain 100% Free Software, so the distribution would only include software released under licenses that the Debian team considered free, such as Richard Stallman's

[125] Ian Murdock, "The Debian Manifesto," Debian.org, last modified January 6, 1994, https://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html,(accessed November 7, 2018). The Debian project name came from combining Ian Murdock's first name with that of his girlfriend, Debra Lynn.

GPL or Berkeley's BSD license. Second, Debian committed to releasing everything produced by the project as Free Software. Third, the Debian project promised technical transparency through the creation of a publicly available issue database. Fourth, the team agreed to prioritize their efforts based on the needs of Debian users in the Free Software community. Finally, to accommodate community members that used programs that did not meet Debian's Free Software criteria, the team promised to provide an FTP site where vendors could share their applications.[126]

The social contract also included the Debian Free Software Guidelines (DFSG.) This section reiterated the definition of Free Software consistent with the GPL license, calling for free redistribution of source code, allowing derivative works and requiring reciprocity for distribution. The section also went further, stipulating that the license of any Debian component could not discriminate against "any person or groups of persons," or specific field of endeavor, such as restricting software from being used at a corporation.[127]

While Emmett Plant had encouraged the Free Software community to re-embrace their cooperative practices in his article, "The Welcome Wagon," Bruce Peren's "Debian Social Contract" was a promise, a recommitment on the part of the Debian team that they would adhere to the ideologies of Free Software. Both documents demonstrate the importance of the community and the desire of its members to stay true to Free Software

[126] Bruce Perens, "Debian Social Contract," Debian.org, July 5, 1997, https://www.debian.org/social_contract.1.0, (accessed November 7, 2018).

[127] Ibid. The latter promise was to keep the project apolitical. The University of California Berkeley had previous created electronic simulation software called SPICE, which they released under a license with a clause that specifically forbid the use of the software by the South African Police forces. This anti-Apartheid license made the software a political tool and, in the process, limited the ideological freedom behind the Free Software Movement.

customs and beliefs. The texts were both distributed on the Internet and they give insight into the vibrant Free Software community that existed on the network at that time.

Summary: Symbiotic Expansion

The ARPAnet arose in the 1960s at a government agency that grew out of Cold War concerns over the U.S. falling behind the Soviets in technological research. In the 1970s the ARPAnet expanded into a global network, and researchers developed internetworking protocols to facilitate this growth. Consistent with the computing culture in this early period, these innovations were achieved through a cooperative approach. The Internet protocols were defined through cooperative standard proposals called Request for Comments (RFCs). TCP/IP, the Internet protocol itself, was also the product of a collaborative planning effort, and the associated software was optimized by the team responsible for the free BSD UNIX distribution. Computing giants such as Microsoft included BSD's TCP/IP software in operating systems such as Windows NT and Windows 2000. The same BSD team wrote and shared the first Domain Name Service (DNS) software, BIND, which remains the dominant Internet DNS application used to this day.

In the early 1990s researchers at CERN developed the hypertext markup language, and the software needed to support it. They freely shared the technology, releasing it first in the public domain, and upon realizing the dangers of corporate co-option, as Free Software. This laid the foundation for the World Wide Web, which evolved into a graphical interface for the Internet thanks to efforts done by the NCSA team at the University of Illinois who had developed their own version of CERN's web software. Their version allowed color images to be embedded within web pages. The NSCA's Mosaic web browser formed the foundation of commercial applications such as Netscape and Internet Explorer, and their abandoned web server was the basis of the Free Software Apache project. Following its release, the Apache project's web server quickly emerged as the dominant web hosting platform on the Internet.

While the Free Software Movement had a key role in developing and supporting the Internet and the World Wide Web, the network also provided the platform for facilitating the growth of the movement. Internet Chat Rooms and websites such as Slashdot provided a digital common where Free Software projects were planned, executed, and maintained. The Internet also supported the collaborative development efforts with source code versioning systems and Free Software distribution methods such as FTP and websites. The result was that as the Internet and the World Wide Web grew supported by Free Software, the Free Software Movement also grew, supported by the Internet and the World Wide Web.

The symbiotic nature of the relationship between Free Software and the Internet was the natural result of the Free Software community and its fundamental drive to create an unrestricted environment in which to work.[128] In his GNU announcement, Stallman made clear that his intention was to create a Free Software environment, "So that I can continue to use computers without violating my principles, I have decided to put together

[128] There is a recursive nature to the Free Software community in that they create the environments in which they prefer to work. Duke University Law Professor James Boyle observed this in his 2003 study of coordination of free and Open Source software projects. Anthropologist Chris Kelty re-articulated Boyle's observations, calling the relationship a "recursive public," a Habermas public "that is constituted by a shared concern for maintaining the means of association through which they come together as a public." Neither Boyle nor Kelty are historians, and as such they do not appear to realize that this is not a coincidence, but the intended outcome from the onset, as seen in historical sources such as Stallman's GNU announcement and GNU manifesto.

a sufficient body of Free Software so that I will be able to get along without any software that is not free."[129]

By 1997 Stallman's stated goal had been met, and there was finally a full Free Software ecosystem available. Although not his planned GNU kernel, Linux provided an operating system environment for the Free Software applications being developed and shared on the Internet. But for some members of the community, the personal freedom inherent in having a complete Free Software ecosystem was not enough – the success of Free Software demanded commercial approval as well.

[129] Richard Stallman. " new Unix implementation,". Email to net.unix-wizards,net.usoft mailing list. September 27, 1983.

CHAPTER 6: FROM FREE SOFTWARE TO OPEN SOURCE

The PC revolution that had begun in the previous decade reached a new stage by the early 1990s as computers were networked. This allowed network resources such as files and printers to be shared. Corporations deployed Local Area Networks (LANs), and access to global networks became increasingly common. By this point, the National Science Foundation's NSFnet backbone supported the pre-commercial Internet. On April 20, 1995, the NSFnet network was decommissioned, and its traffic moved to a backbone provided by commercial service providers. The public Internet was born.

The public increasingly accessed the newly available Internet via the World Wide Web. Programmers at the National Center for Supercomputing Applications (NCSA) and later Netscape, continued to advance the technology by providing a web browser with a fully graphical interface. This enabled individuals who lacked a high-level of technical literacy to access the Internet. Free software played a key role in supporting the Internet and the World Wide Web. At the same time, the network fostered the continued growth of the Free Software Movement. By 1997, a thriving Free Software community existed on the Internet, and the new Free Software operating system, Linux, helped pique interest in the movement.

This chapter explores the emergence of web browsers in the mid-1990s and how the Microsoft corporation utilized proprietary technologies in an attempt to dominate the public's Internet experience. Microsoft's tactics prompted the Netscape corporation to release their web browser as Free Software. Netscape's release announcement was subsequently leveraged by entrepreneurial Free Software enthusiasts to introduce a corporate-friendly reinterpretation of Free Software called "Open Source." This chapter will demonstrate how the Open Source re-branding created a rift within the Free Software Movement as self-appointed community leader Eric Raymond attempted to use the Open Source brand to strip the ideological foundation from Free Software. Raymond's attempts to curry favor within the corporate sphere were ultimately successful. By the end of 1999, Open Source companies enjoyed significant commercial interest and investor support.

The Browser Wars

In early 1993, the World Wide Web was still in its infancy. Its creator, Tim Berners-Lee, had produced a text-based browser that allowed for linking between Internet-based content. On January 29, 1993, the NCSA team at the University of Illinois announced the release of a new web browser. The software, named Mosaic, differed from Berners-Lee's program in that it was designed specifically for use with graphical interfaces. Creator Marc Andreesen announced the availability of the program for UNIX and provided the software binaries as well as the source code via a University of Illinois FTP site. The program was quickly endorsed by Tim Berners-Lee who re-hosted the new application for distribution from CERN.[1] By November of that year, the Mosaic team ported the software to popular home computing operating systems, Microsoft Windows, and Apple Macintosh's Mac OS.[2] [3] As with the UNIX version, the browser software was freely available. Unlike the UNIX-based version, however, the software release for these

[1] Tim Berners-Lee. "New "XMosaic" World-Wide Web browser from NCSA", January 29, 1993. alt.hypertext,comp.infosystems. Email to Newsgroup.

[2] Chris Wilson. "NCSA Mosaic for MS Windows 1.0 release notice (Win31- Winsock)". Email to Newsgroup. November 11, 1993. comp.os.ms-windows.announce.

[3] Kim Stephenson. "NCSA Mosaic for the Macintosh 1.0". Email to Newsgroup. November 11, 1993. comp.sys.mac.comm.

operating systems did not include the source code**.** Mosaic's creators had other plans for their project.

Netscape

By early 1994, over two million people were using the Mosaic web browser. The popularity of the software encouraged its creators to form their own commercial venture, and in May 1994 Marc Andreessen announced the creation of a new company designed to support and expand the technology behind the Mosaic browser. Originally named the Mosaic Communications Corporation, the organization was co-founded by James Clark, a computer scientist who had previously founded Silicon Graphics, a corporation that created high-end graphics and animation workstations. The majority of the Mosaic team joined the new organization. In addition to supporting their previous product, the Mosaic browser, the organization sought to develop and license a new commercial platform that included a web browser, server, and web authoring suite.[4] Citing concerns from the University of Illinois, which owned the rights to the original Mosaic software, on November 14, 1994, the organization announced that it had been renamed to the Netscape Communications Corporation. The change was explained as a way to emphasize the company's identity and mission as being separate from that of the original Mosaic project.[5] The corporation took its new name from the Netscape Network Navigator, a version of the organization's new browser that had been released on the

---

[4] Marc Andreessen. "Introduction to Mosaic Communications Corp.". Email to Newsgroup. May 9, 1994. comp.infosystems.www,comp.infosystems,alt.hypertext.

[5] "Mosaic Communications Changes Name to 'Netscape Communications Corporation,'" (Mountain View, California: Netscape Corporation, 1994).

Internet a month prior. The popular software was free for individual, personal use, but cost $99 for commercial users.[6]

Spyglass

Since Mosaic was developed at the University of Illinois/Urbana-Champaign, the university owned the original Mosaic software. With the core of the Mosaic team having left to form Netscape in June 1994, the university decided to partner with Spyglass, Inc., a data analysis software development company.[7] The university assigned all commercial licensing rights for the Mosaic software to Spyglass, and the corporation was responsible to develop and market enhanced versions of the software. Spyglass immediately prepared an updated release of the Mosaic browser for the Microsoft Windows and Apple Macintosh Mac OS operating systems.[8] By August of that year, Spyglass had established distribution agreements with organizations including IBM.

In December 1994, Spyglass signed a contract that allowed Microsoft to use the Mosaic source code on the Windows operating system. After a failed bid to license the Netscape browser, Microsoft agreed to a contract that gave them rights to use the software on its upcoming Windows 95 operating system. For this privilege, Microsoft would pay Spyglass $2 million and a royalty for each copy of the browser that they sold,

---

[6] "Mosaic Communications Offers New Network Navigator Free On the Internet" (Mountain View, California: Netscape Corporation, 1994).

[7] The University had created Spyglass in 1990 to market and support technologies developed by the NCSA.

[8] Ellis Booker, "Spyglass to Commercialize Future Versions," *ComputerWorld*, August 29,1994, 16.

up to $5 million per year.[9] Microsoft rebranded the software as their own product, Internet Explorer.

Internet Explorer

With the Rolling Stones' hit "Start Me Up," playing in the background, Microsoft CEO Bill Gates joined comedian Jay Leno onstage for the August 25, 1995 launch of Windows 95. In addition to highlighting Windows' dramatically changed interface and innovative "Start" menu, the event also introduced Microsoft's new web browser, Internet Explorer. The software was part of the "Microsoft Internet Jumpstart Kit," an array of tools designed to encourage the adoption of the Microsoft network, MSN. The software was available as part of Microsoft Plus!, an add-on product for Windows 95. It was also available from select Internet providers and hardware vendors.[10]

Even prior to this Windows 95 marketing blitz, Gates intended to place the new Internet browser at the core of the Windows 95 experience. In his confidential May 1995 *Internet Tidal Wave* memo, the CEO predicted a connected world where nearly all personal computers would be linked and using Internet resources. In the document Gates bemoans the lack of Microsoft's proprietary formats on the World Wide Web. "Browsing the Web, you find almost no Microsoft file formats," Gates reported, after spending hours browsing and finding he had not seen "a single Word DOC, AVI file, Windows EXE" or other proprietary Microsoft file format. Gates identified Netscape's current popularity as an advantage for the competitor, stating that since "their browser is dominant with 70%

---

[9] Steve Lohr, "Spyglass, a Pioneer, Learns Hard Lessons About Microsoft," *New York Times*, March 2, 1998, D1.

[10] Sandi Hardmeier, "The History of Internet Explorer," Microsoft.com, August 25, 2005, http://www.microsoft.com/windows/ie/community/columns/historyofie.mspx.

usage share," Netscape could determine the technologies that would "catch on." This was particularly concerning to Gates as Netscape was "pursuing a multi-platform strategy" that allowed them to "commoditize the underlying operating system." In other words, Netscape was operating system agnostic, eliminating any dependence on Microsoft's Windows products. Gates reports the "scary possibility" that Internet users might be able to move to a new technology, "something far less expensive than a PC which is powerful enough for web browsing," and optimized for "the datatypes on the Web." With the specter of a lower-cost PC replacement, Microsoft needed to set their products as central to the user's experience in the newly connected world. This meant moving quickly to integrate the Internet experience with the Windows Operating system to allow Microsoft's proprietary technologies to dominate the market. "We need to establish OLE protocols [Microsoft's proprietary programming model] as the way rich documents are shared on the Internet. I am sure the OpenDoc consortium will try and block this."[11] Simply put, Bill Gates determined that to achieve market dominance, Microsoft would need to undercut competitors. To "get people to switch away from Netscape," the web browser had to be inseparable from the operating system. Within the memo, Gates instructed that the company's Internet software would need to be integrated "into Windows 95 itself," where the operating environment "and the browser will converge."[12] Consistent with this directive, within months of the initial release of Internet Explorer, the corporation introduced Internet Explorer Version 2 as a free download and the software was integrated in the Windows 95 operating service pack 1.

[11] Gates, "The Internet Title Wave Memo," 5. Bill Gates' vision of an inexpensive computer based on web-browsing would eventually come true with the introduction of Google's Chrome-based computers.
[12] Ibid.

Microsoft's strategy to undermine Netscape by positioning the Internet Browser as central to the Windows 95 operating system brought an unintended consequence -- a lawsuit from Spyglass, Inc. Microsoft's decision to place the browser as a free part of the operating system meant that Spyglass would no longer receive its contractually guaranteed royalty payments. Following Microsoft's Internet Explorer 2.0 release, the Mosaic distributor reported a loss of tens of millions of dollars in yearly revenue.[13] The Spyglass suit was resolved in 1997, with Microsoft providing the software distributor with a onetime payment of $7.5 million in cash.[14]

Integrating Internet Explorer was an effective strategy for Microsoft. First, it provided PC users with a built-in Internet browser, eliminating the need for them to find software to use. Next, since Internet Explorer was free of cost for corporate users, Microsoft could undercut competitors like Netscape, which had been charging $99 per user.[15] However, the real threat was that the integration of Internet Explorer would afford Microsoft a monopoly in the browser market. Microsoft could then leverage their market dominance to force their own proprietary formats as the de facto standards for the Internet. This approach -- precisely what Gates had outlined in the confidential *Tidal Wave* memo -- would lock users into using Microsoft technologies and, in doing so, drive competitors like Netscape out of the server market.

Challenged by the possibility of Microsoft having a monopoly lock on their primary market, Netscape began to consider actions to counteract Microsoft's anti-

---

[13] Steve Lohr, "Spyglass, a Pioneer, Learns Hard Lessons About Microsoft," *New York Times*, March 2, 1998, D1.

[14] Michael Mercurio, "Microsoft's $8 Million Goodbye to Spyglass," *Business Week*, January 22, 1997, http:/www.businessweek.com/bwdaily/dnflash/january/new0122d.htm, (accessed December 10, 2018).

[15] For Netscape, this alone would have only a marginal impact: the vendor already gave the browser away to non-corporate clients since most of Netscape's income came from the server market.

competitive maneuvers. One option that was discussed was for Netscape to release both the browser and its source code as Free Software. This would counteract Microsoft's bid for a monopoly by encouraging the proliferation of the Netscape technologies. The action would also encourage Microsoft's competitors to collaborate on the freely available browser platform. The idea of releasing the source code was not new; while at the University of Illinois, the programming team had previously released the original UNIX source to the Mosaic browser. In late 1997, Netscape Systems Engineer Frank Hecker developed a position paper supporting the idea. In the internal document, "Netscape Source Code as a Netscape Product," he argued that source code was not just something used to create a software product but could be considered as a product in itself. Within the document Hecker made a business case for the approach, considering factors such as business models, licensing, and potential competitor reactions. As an example of how a distributed development model could work, Hecker cited "The Cathedral and the Bazaar," a paper written by Free Software programmer Eric Raymond.[16]

The Cathedral and the Bazaar

"The Cathedral and the Bazaar" was originally a speech given by Eric Raymond on May 21, 1997 at a Linux convention in Bavaria. The original lecture was a light-hearted discussion of Free Software, focused on the motivations of Linux contributors. Following the speech, one of the attendees, publisher Tim O'Reilly, asked Raymond to give the speech again at O'Reilly's upcoming conference on the Perl programming

[16] *Revolution OS*, directed by J.T.S. Moore. (Wonderview Productions, 2002), DVD (Wonderview Productions, 2012).

language.[17] Based on the interest in his perspective, Raymond developed the speech into a written paper where he laid out his vision of Free Software as a development model. He detailed the differences between traditional software development processes and the Free Software approach. Raymond documented the distributed method that had evolved in Free Software and provided a pseudo-ethnographic assessment of the Free Software community through their programming model. Through the lens of projects such as Linux, the author detailed the differences between a traditional, monolithic, programming model, "the cathedral," where one organization tightly controlled the development of software, in contrast to the Free Software community's new distributed model, "the bazaar."[18] Raymond's text was remarkable in that it provided a simple, easy to understand account of how software development worked within the Free Software community

The document had its faults: it was largely editorial, factually inaccurate at times, and redolent of self-promotion, but it was significant in that it provided a blueprint for the Free Software process, the distributed approach, which later projects and organizations could follow.[19] At Netscape, the argument set out in "The Cathedral and the Bazaar" was a factor in the organization ultimately deciding to release their web browser as Free Software. The inclusion of the document in Frank Hecker's memo resulted in the organization asking Eric Raymond to participate in discussions as they developed a

[17] Ibid.

[18] Eric Raymond, "The Cathedral and the Bazaar," catb.org, May 21, 1997, http://www.catb.org/esr/writings/cathedral-bazaar/, (accessed March 14, 2015).

[19] Weber, *The Success of Open Source,* 11.

custom Free Software license for the release of the web browser.[20] Raymond eagerly participated in the process. He saw Netscape's yet-to-be-announced move to Free Software as a unique opportunity for the Free Software Movement to emerge into the mainstream. But for this to work, Raymond believed that Stallman's ideological focus on individual freedoms had to be purged.

Eric Raymond

Eric Raymond had been involved with Free Software since its conception. According to Raymond, he had been friends with Stallman since 1979, and he had spoken with him about the GNU project before its launch.[21] This relationship is interesting because Stallman and Raymond were, in many ways, polar opposites: Stallman was highly educated, a product of Harvard and MIT and held over a dozen honorary doctorates.[22] Raymond attended the University of Pennsylvania briefly but dropped out without completing a degree.[23] Stallman was extremely liberal, supporting the Green Party and many environmental causes.[24] Raymond, on the other hand was extremely conservative, an avowed libertarian who frequently wrote screeds against anyone who criticized right-wing political decisions.[25] Eric Raymond's blog, aptly named "Armed and Dangerous," detailed his fetishism with firearms and violent hobbies – martial arts and sword fighting. Both Stallman and Raymond were programmers, but where Stallman

---

[20] Bruce Perens, "The Open Source Definition" in O*pen Sources: Voices from the Open Source Revolution.* Ed.Chris DiBona, Sam Ockman, and Mark Stone. (Beijing: O'Reilly, 1999), 173.

[21] Eric Raymond, email to Chris Campbell, March 1, 2015.

[22] Richard Stallman, "A Serious Bio," Stallman.org, https://stallman.org/biographies.html, (accessed April 20, 2015).

[23] Eric Raymond, "Resume of Eric Steven Raymond," catb.org, last modified January 29, 2003, http://www.catb.org/esr/resume.html. (accessed April 20, 2015).

[24] Richard Stallman "Richard Stallman's Personal Site," Stallman.org, https://stallman.org/, (accessed April 20, 2015).

[25] Eric Raymond, "Why We Fight — an Anti-Idiotarian Manifesto (2.0)," catb.org, December 29, 2003, http://www.catb.org/esr/aim/, (accessed April 20, 2015).

was responsible for the GNU projects and some of the most commonly used Free Software programs, Raymond had written considerably less and chiefly had been a co-contributor or maintainer of existing software. Stallman was the "last of the true hackers," featured as the climax of Stephen Levy's book on the hacker community.[26] Levy neglected to include Raymond in his pantheon of hackers, but Raymond's self-identity as a hacker was so strong that he asserted himself as the authority for hackerdom itself by writing articles such as "How To Learn Hacking," "A Brief History of Hackerdom," and the "Revenge of the Hackers."[27] Raymond even took over the Jargon file, a compendium of MIT-AI lab slang, and published it as *The New Hacker's Dictionary.*[28]

Where the men differed most was in their interpretation of Free Software. Whereas Stallman introduced the movement as a form of consumer activism focused on individual rights, Raymond rejected Stallman's ethical rhetoric.[29] For Raymond, the movement should not have been about ideological freedoms or morals but providing a solution to a problem that had emerged within corporate software development. According to Raymond, the "*management* of programming" had become "separated from the *art* of programming" and in the process created work environments that were "intolerable" for true hackers such as Raymond. In his view, Free Software was a

[26] Levy, *Hackers*, 437.

[27] Eric Raymond, "How to Learn Hacking," catb.org, November 21, 2014, http://www.catb.org/esr/faqs/hacking-howto.html, (accessed April 20, 2015); Eric Raymond, "A Brief History of Hackerdom," catb.org, May 5, 2000, http://www.catb.org/~esr/writings/hacker-history/hacker-history.html, (accessed April 20, 2015); Eric Raymond, "Revenge of the Hackers," catb.org, August 31, 1999, http://www.catb.org/~esr/writings/cathedral-bazaar/hacker-revenge/, (accessed April 20, 2015).

[28] Eric Raymond, "The New Hacker's Dictionary, Third Edition," October 1996, MIT Press, http://mitpress.mit.edu/books/new-hackers-dictionary, (accessed April 20, 2015).

[29] Eric Raymond, email to Chris Campbell, March 1, 2015.

development model that alleviated this issue by giving programmers complete control of their craft.[30] Raymond felt that if he were able to separate Stallman's ideology from the movement, the community would be accepted by the corporate software industry that Richard Stallman had rejected .[31]

Open Source

On February 3, 1998, immediately following Netscape's Free Software release announcement, a group of Free Software entrepreneurs met in the Mountain View, California offices of hardware company V.A. Linux.[32] The group discussed the recent Netscape source code release, which they perceived as an opportunity to gain momentum for the Free Software Movement. Eric Raymond, who had just met with Netscape about their licensing decision, used the occasion to introduce the idea of rebranding Free Software.[33] Raymond argued that Stallman's stated ideologies were holding back adoption of Free Software because "conservative business people were put off by Stallman's freedom pitch."[34] At Raymond's insistence, the group agreed to undertake a marketing campaign to make the movement more palatable to corporations.[35] Concerned with having "credibility with investors," Raymond argued that the name "Free Software" itself was the issue. "If you walk into an executive's office and say, 'Free Software,'" Raymond argued, "If you're lucky, the response you'll get will be…must be cheap,

---

[30] Eric Raymond, "Why I Hate Proprietary Software," http://esr.ibiblio.org, October 1, 2008, http://esr.ibiblio.org/?p=556, (accessed May 13, 2015).

[31] Eric Raymond, "Goodbye, 'Free Software'; Hello, 'Open Source'," Catb.org, February 8, 1998, http://www.catb.org/esr/open-source.html, (accessed May 4, 2016).

[32] *Revolution OS*.

[33] "History of the OSI," The Open Source Initiative, September 2012, accessed March 14, 2015, http://opensource.org/history.

[34] Bruce Perens, "The Open Source Definition" in O*pen Sources,* 173.

[35] Chris DiBona, Sam Ockman, and Mark Stone, "Introduction" in O*pen Sources*, 3.

shoddy, worthless...it's lousy marketing. It's not something that businesses want to hear."[36] Raymond's focus came down to the word 'free.' The Free Software Movement intended that their software be free, as in *libre*, with the occasional result that the software be free, as in *gratis* as well. Raymond was not interested in Stallman's "free" ideology. By rebranding the movement, the confusion over the word "free" would go away and with it the implicit link to Stallman's beliefs. The group settled on a replacement for Free Software, "Open Source," a term recommended by meeting attendee Christine Peterson.[37]

Following the Open Source renaming decision, Eric Raymond presented himself as the reluctant leader of the revised revolution. He rationalized that his role in the Netscape Free Software release decision, "the shot heard 'round the world," cast him as a "Thomas Paine" figure. "The hard part (for me anyway) was accepting what my own role had to be," Raymond wrote of his self-appointment to the leadership of the Open Source Movement. "I knew somebody with very particular characteristics would be needed…." He recalled, "Looking around me, I couldn't see anyone better qualified…but I didn't want the job, because I knew it would cost me my life for many months, maybe for years."[38]

Whatever reservations Eric Raymond may have had about his role in the new business-friendly re-branding of Free Software passed quickly. Within two days of the meeting, Raymond published an article notifying the Free Software community that they

---

[36] *Revolution OS*.
[37] Ibid.
[38] Eric Raymond, "The Revenge of the Hackers" in O*pen Sources*, 210-214.

were now the “Open Source” community. In his article, “Goodbye, ‘Free Software’; hello, ‘Open Source.’” Raymond informed the community that he had given a lot of thought to the movement’s next phase, which he held to be a “serious push to get ‘Free Software’ accepted in the mainstream corporate world,” and in the process, he found that “we have a serious problem with ‘Free Software’ itself.” He argued that the name for the movement was ambiguous and “makes a lot of corporate types nervous.” He informed the community of the February 6 meeting, in which the attendees selected the term “Open Source” as a replacement for Free Software. Raymond then stressed the need for the community to conform everything to this new name, “everywhere we as a culture have previously talked about "Free Software", the label should be changed to "Open Source". Open-source software. The open-source model. The Open Source culture. The Debian Open Source Guidelines.”[39]

Following his declaration of the Open Source Movement, Raymond labored to achieve adoption of the re-branding. In the days that followed his announcement, Raymond revised “The Cathedral and the Bazaar,” purging any mention of Free Software and adding a section to herald Netscape’s embrace of the new “Open Source” movement.[40] Consistent with his new self-appointment as chief Open Source evangelist, Raymond also began to reach out to the media to promote the newly re-labeled movement.

[39] Eric Raymond, “Goodbye, 'Free Software'; Hello, 'Open Source',”

[40] Eric Raymond, “The Cathedral and the Bazaar: Revision History,” catb.org, last modified February 8, 2002, http://www.catb.org/esr/writings/cathedral-bazaar/cathedral-bazaar/index.html#catbmain, (accessed May 4, 2016).

On April 7, 1998, publishing company O'Reilly & Associates held a summit in Palo Alto. The stated agenda for this meeting resembled the February discussion at V.A. Linux -- to consider the implications of Netscape's source code release and to discuss the movement. But in February, the attendees were chiefly Free Software entrepreneurs like V.A. Linux's Larry Augustin and Penguin Computing's Sam Ockman. For the April meeting, the attendees were Free Software community leaders. The Linux operating system's Torvalds joined representatives from many other key Free Software projects, such as Larry Wall and Guido Van Rossum, creators of the Perl and Python programming languages, respectively. The Apache project's Brian Behlendorf was in attendance as well as the Sendmail project's Eric Allman and BIND's Paul Vixie. The creator of the PGP encryption software, Phil Zimmerman, was also present, as was John Gilmore, co-founder of GNU software support company Cygnus.[41]

From Eric Raymond's June 1998 summary of the meeting, it is clear the summit had the unstated goal of selling the Free Software community leaders on the Open Source rebranding of the movement. In his retelling of the event, Raymond started by listing the storied attendees while downplaying his own involvement, "For my semi-accidental role in motivating the Netscape source release with 'The Cathedral and the Bazaar', I also had the honor to be among those invited." He quickly moved on to what he claimed to be "one of the most important purposes of the meeting…to permit everyone to meet face to face… build more personal trust among the chieftains of the major Open Source tribes." In the article, Raymond then detailed aspects of the discussion, such as "different

[41] Eric Raymond, "Open Source Summit," *Linux Journal*, June 1, 1998, https://www.linuxjournal.com/article/2918, (accessed October 2, 2019).

perspectives on the Open Source/Free Software phenomenon," as well as "the vexing issue of labels." For the latter, he reported that the group "agreed to use 'Open Source' as our label." Raymond then went on to detail how the group fielded questions about the re-labeling at a pre-planned press conference at the end of the day. Raymond's meeting summary ended by calling for unity under his new movement, stating that the programs "need to pull together as one community more than we have in the past." This meant that constituent groups such as the Linux community and the Free Software Foundation "need to present one face and speak one language and tell one story to that larger world."[42] There no longer was to be a Linux community or Free Software Movement; everyone was part of the "Open Source" movement now, and everyone had to get with the program.

Supplanting Ideology: Homesteading the Noosphere

As the self-appointed leader of the Open Source Movement, Eric Raymond's stated justifications for rebranding Free Software were typically focused on the goal of mainstream corporate acceptance. The renaming achieved this by addressing Eric Raymond's personal issue with Free Software -- the ideology that had informed Stallman's objectives. In Eric Raymond's mind, Stallman's Free Software Movement overstepped its bounds from its formation, an overreach implicit in its name. According to Raymond, "Richard Stallman ('RMS') tried to give it a name —'Free Software'. But his act of naming was also an act of claiming; he attached ideological baggage to the 'Free Software' label." Raymond argued that most of the hackers never accepted Stallman's ideologies and the Free Software label was "loudly rejected by a substantial

[42] Ibid.

minority of the hacker community" and used with "silent reservations" by the remainder.[43] Raymond's solution was to do the same. Raymond's act of re-naming was also an act of claiming, and just as eagerly as he rejected Stallman's ideological baggage, Raymond injected his own. This can be seen in Raymond's April 1998 treatise, "Homesteading the Noosphere."

Just three days after the O'Reilly summit in Palo Alto, Raymond published "Homesteading the Noosphere," a manifesto that simultaneously minimized the views of the Free Software advocates while imbuing the history and practices of the Free Software community with Raymond's own beliefs. The document evoked summit attendees such as Larry Wall and Guido van Rossum as examples and situated them apart from the traditional beliefs of the Free Software Movement, while presenting the Open Source definition as the true summation of the communities' beliefs.

In the document Eric Raymond revisited the quasi-anthropological approach he employed in the "Cathedral and the Bazaar." He began by analyzing "Open Source" culture ideologies, situating them on a spectrum of corporate acceptance. He cast Stallman and his ideology as being relevant due solely to the Free Software Foundation's vocal partisans, which resulted in the group's view becoming "the closest thing to a hacker ideology, and RMS [Stallman] the closest thing to a leader of the hacker culture." Despite the GNU Manifesto's prolix discussion of business models and Stallman's repeated assurances that he was not anticommercial, Raymond cast him as such because "his program has been so read by most people." Raymond suggested that the majority of

---

[59] Eric Raymond, "How to Become a Hacker," catb.org, October 4, 2007, http://www.catb.org/esr/faqs/hacker-howto.html#history, (accessed October 2, 2019).

programmers shares his pro-corporate views and were more pragmatic. These hackers were moderately anticommercial in such that they were against “the world’s perverse refusal to adopt superior approaches,” and any focused animosity they had is toward the standards bearer of the software establishment -- Microsoft.[44]

Eric Raymond then rearticulated the history of the recent Open Source changes. The author ignored his own involvement in events, retelling the Netscape source release as something that “excited more interest in `Free Software' within the corporate world” which then led to a “call to the hacker culture to exploit this unprecedented opportunity and to re-label its product from `Free Software' to `Open Source.'” Raymond claimed that the corporate friendly re-articulation of Free Software “was met with a level of instant approval that surprised everybody involved,” -- everyone directly involved with the re-branding anyway. Raymond ignored the perspective of the Free Software advocates present at the meeting, minimizing their perspective by devaluing them as “anticommercial purists who found themselves in a minority.” He established this by asserting projects that created custom licenses purposely did so to express “ideological independence.” In Raymond’s calculus, anyone who used a license other than Stallman’s original GPL was not ideologically aligned with Free Software, but rather Open Source.[45]

Following Raymond’s reinterpretation of the history of Open Source, he moved into a more general discussion that touched on many of the themes introduced previously

---

[44] Eric Raymond, “Homesteading the Noosphere,” catb.org, April 10, 1998, http://www.catb.org/~esr/writings/homesteading/homesteading/index.html, (accessed October 2, 2019).
[45] Ibid.

in the “Cathedral and the Bazaar.” He started by addressing project ownership, which he equated to the Lockean theory of property, discussing project ownership in the context of land acquisition. In this, he detailed three primary types: In the first, the eponymous homesteading, the project creator was its owner. The second type was based on transfer of title, where a project owner turned over control to another party. Finally, he discussed the reclamation of abandoned efforts, where new owners could claim control over an existing project.[46]

Following the project ownership discussion, Raymond’s treatise directly revisited many of the ideas introduced “Cathedral and the Bazaar.” In the original document, Raymond proposed that ego was the mechanism that informed the Open Source hacker community; attention was the reward. In this new text, Raymond extrapolated on the idea that the community was a gift culture where the programmers enjoyed programming, but fundamentally, they exchanged the fruits of their labor for a positive reputation in the community, which was itself an ego-driven motivation.

In “Homesteading the Noosphere,” Raymond also reiterated his original theories in an attempt to refute dissenting reactions he previously received to the “Cathedral and the Bazaar.” In the document, he cast the differing opinions about his ideas as problematic for the hacker culture, stating “many hackers resisted the analysis and showed a strong reluctance to admit that their behavior was motivated by a desire for

---

[46] The arguments Eric Raymond presented here were topically relevant to the author himself, who was commonly known to be eager to take over projects written by others while restricting contributions to his own. For example, Raymond’s Fetchmail utility was notoriously insecure and buggy, but he refused contributions to address the problems. See: https://docs.freebsd.org/cgi/getmsg.cgi?fetch=585008+0+archive/2001/freebsd-arch/20010218.freebsd-arch, (accessed October 2, 2019).

peer repute or, as I incautiously labeled it at the time, `ego satisfaction'." Raymond saw individual ego as a good thing, lamenting that the "belief that ego is evil" harms the hacker community in that it renders programmers emotionally unable to understand their own culture.[47] Programmers who disagreed with Raymond's world view were painted as emotionally incapable of grasping the true nature of the community's social dynamics.

Raymond re-emphasized his contention that prestige was the currency of the culture's reward mechanism. He then attempted to understand why this was "largely unadmitted." Having determined that ego is good, Raymond pondered why the community would value humility. Raymond determined that the drive for humility was due to prominent hackers having "fear of becoming the object of a personality cult." To support this claim Raymond recalled an anecdote about Larry Wall, the renowned creator of the Perl programming language, "Once, on a dinner expedition with Larry Wall, I joked 'You're the alpha hacker here—you get to pick the restaurant.'' He flinched noticeably."[48]

Raymond then segued from the joke to the suggestion that Open Source hackers define intellectual territories for their projects in a manner similar to animal territoriality:

> By marking his bounds, and respecting the bounds of others, a wolf diminishes his chances of being in a fight that could weaken or kill him and make him less reproductively successful. Similarly, the function of property in human societies is to prevent inter-human conflict by setting bounds that clearly separate peaceful behavior from aggression.[49]

[47] Eric Raymond, "Homesteading the Noosphere."

[48] Ibid. Larry Wall is well known within the Free Software community as an ardent fundamentalist Christian. Originally trained as a linguist, Wall's reaction could equally be due to discomfort with the sexual and societal implications of Raymond's joke.

[49] Eric Raymond, "Homesteading the Noosphere."

Raymond delved into a deep discussion of interpersonal conflict within the Open Source community, which he relates to issues of property and skill level.[50] Raymond suggested that software projects are property, marked territory, so the challenge comes with larger, complex projects with more than one contributor. By custom, the project has a leader, the originator or most skilled developer, who acts as a "benevolent dictator," and makes decisions. Raymond was dismissive of projects that do not follow such an approach, such as with the Apache web server effort that was led by committee voting. According to Raymond, such democratic approaches without a strict leader, are complicated arrangements that are "widely considered unstable and difficult."[51] To address the instability presented by electoral processes, the voluntary societies that form around community software projects require either a dictatorial leader, or they need to contractually self-regulate. To achieve the latter, Raymond concluded his paper by proposing the creation of a formalized code to adjudicate actions within the new Open Source community. Citing the potential of community growth, Raymond postulated that "it is time for the hacker culture …to develop written codes of good practice for resolving the various sorts of disputes that can arise in connection with open-source projects."

[50] In a later draft of the article, Raymond had a desultory discussion in the context of skill levels acting as shibboleths, in which he discussed feedback connecting the author's observations as re-statements of existing, academic phenomenon. For example, in the *Cathedral and the Bazaar*, Raymond introduced what he calls Linus' law, "With enough eyes all bugs are shallow." This was a programmatic description of peer review. Respondents to the original draft pointed out that many of the customs he was attempting to document where with present in academia and since hacking originated in Universities were likely simple transpositions of the practices. Raymond dismissed these observations, since "hacker custom seems to be readily acquired by intelligent high-schoolers!" Raymond's response to these obvious relationships may have been the result of the author having not completed any secondary education himself.

[51] Eric Raymond, "Homesteading the Noosphere."

Raymond then revealed that he has begun work on such a document of laws, which he has titled the "Malvern Protocol" (named after Raymond's Pennsylvania hometown).[52]

In a single document, produced days after the O'Reilly summit anointed the Open Source re-articulation of Free Software, Eric Raymond minimized the ideological origins of the movement, re-articulated its history, and aligned it with his own views on property and social hierarchies. He used the document to dismiss the concerns of his critics while he introduced the need for a structured Open Source community that would follow rules, such as the ones that Raymond was personally developing. He also identified a common enemy for the rebranded movement -- Microsoft.

The Open Source Initiative

In the months following the introduction of the "Open Source" rebranding, Eric Raymond paired with Debian Linux project leader Bruce Perens to produce the "Open Source Definition." Perens composed the definition based upon his previous work on the "Debian Free Software Guidelines."[53] The Debian Linux project was designed to stay faithful to the purposes of the GNU project, and Perens' guidelines had reiterated this fealty. Perens had worked closely with Stallman and saw his own views as consistent with those of the Free Software Movement's founder. "I don't think we have any philosophical differences," Perens stated. "Richard wants ALL software to be free…that's the only difference."[54] Bruce Perens, a believer in the ideologies of the Free Software Movement, perceived the Open Source rebranding as merely an effort to bring

[52] Ibid.
[53] Perens, "The Open Source Definition" in O*pen Sources,* 171-174.
[54] *Revolution OS.*

the software into the mainstream through commercialization. As such, the very first criteria of the new definition addressed potential corporate concerns by making it clear that Open Source licensing did not restrict the sale of software.[55]

In February 1998, Perens used the Debian project's 503c corporation, Software in the Public Interest (SPI), to obtain the "Open Source" trademark. He also registered, a new domain for the rebranded movement, OpenSource.org.[56] By the fall of that year, Raymond and Perens had formed the Open Source Initiative (OSI) to steward the new movement. Raymond, who was now the organization's President, also introduced the "Open Source Certification," which would be awarded to software vendors who produced software that met Peren's "Open Source" definition.[57] The OSI began to publish a list of licenses deemed to be officially Open Source.[58] Thus the OSI established itself as the standards authority, the official judge of what software was acceptable for the Open Source designation.

Almost immediately, these moves caused strife within the Free Software community. The OSI acting as a self-appointed gateway for the movement was problematic in that Eric Raymond's perception of what licenses were permissible did not accord with what was traditionally considered Free Software. Several the OSI-approved licenses restricted consumer modification and were therefore contradictory to Free Software's ideals. For example, the OSI-approved Sybase Open Watcom Public License

---

[55] Ibid.

[56] Results for: OPENSOURCE.ORG, https://whois.icann.org/en/lookup?name=www.opensource.org, (accessed December 17, 2018.)

[57] Eric Raymond. "The Open Source Initiative needs your help". Email to comp.os.linux.announce Newsgroup. December 3, 1998.

[58] "History of the OSI," The Open Source Initiative, September, 2012, http://opensource.org/history,(accessed March 14, 2015).

Version limited consumer modification and redistribution.[59] Within the Free Software ideology, the ability to modify software and share the modifications was critical to consumer freedom. Within the OSI's interpretation of Open Source, this freedom did not matter. With the approval of licenses that ignored key tenets of the Free Software definition, the OSI demonstrated its rejection of the ideological underpinnings of the Free Software Movement.

Once the OSI was incorporated, the group sought to consolidate control of the Open Source Movement under their new organization. For instance, even though the Open Source trademark had been acquired by the Debian project's SPI corporation, Raymond demanded that the trademark be transferred to the OSI. Raymond's heavy-handed approach evoked concern from the SPI board who believed that the trademark should be publicly managed by the community. The conflict was met with frustration when it spilled over into other groups, such as the BSD community. When asked to forward an update on the conflict to the NetBSD community, mailing list manager Jordan Hubbard chastised both sides of the conflict, stating that, "this kind of in-fighting isn't doing the cause any good," and suggesting that, "Everyone concerned here, from Eric [Raymond] on down, is in serious need of a reality-check."[60] Unfortunately for Hubbard and the rest of the community, the cracks were just beginning to show.

Commercial Competition: Microsoft

---

[59] "Sybase Open Watcom Public License Version 1.0," Openwartchom, March 25. 2016, http://www.openwatcom.org/index.php/Open_Watcom_Public_License, (accessed May 4, 2016).

[60] Jordan Hubbard. "Re: Future of the `Open Source' trademark." Email to undisclosed-recipients. November 24, 1998.

The Open Source Initiative's marketing efforts, riding the coattails of the Netscape source release, brought the movement the intended result -- corporate attention. In April 1998, Computer Associates revealed that it was porting its Ingress II database to Linux.[61] IBM followed suit and released a beta version of the DB2 database for the operating system, and in June they announced that they were adopting Apache as part of the Websphere product line.[62] Then in July, Oracle announced that they had already begun to port their industry-leading database to Linux and indicated that they planned to begin supporting their product line on the new operating system.[63]

However, not all corporate attention was positive. In August 1998, Microsoft Project Manager Vinod Valloppillil composed two detailed memos that analyzed the burgeoning Open Source Movement. The first memo looked at the movement and how it functioned.[64] The second memo was a business analysis of Linux and other Open Source platforms.[65] Where Netscape's decision to release their browser source code was leveraged to market Open Source to corporations, the move suggested that the Free Software community was aligned with Netscape, a direct competitor to Microsoft. Valloppillil's memos provided a concise summary of the history of Free and Open Source software and detailed the differences between the several types of licenses. The memos concluded that Open Source software was a concern to Microsoft since the free

[61] Weber, 120.

[62] "IBM Enhances and Expands Websphere Product Line in Collaboration with Apache and Netobjects," IBM.Com, June 22, 1998, https://www-03.ibm.com/press/us/en/pressrelease/2587.wss, (accessed January 21, 2020).

[63] Weber, 120.

[64] Vinod Valloppillil, *Open Source Software: A (New?) Development Methodology* (Redmond, Washington: Microsoft, 1998), https://www.gnu.org/software/fsfe/projects/ms-vs-eu/halloween1.html#_Toc427495764, (accessed May 4, 2016).

[65] Vinod Valloppillil, *Linux OS Competitive Analysis:The Next Java VM?* (Redmond, Washington: Microsoft, 1998), https://www.gnu.org/software/fsfe/projects/ms-vs-eu/halloween2.html, (accessed July 14, 2020).

projects had become significant in scale and complexity and maintained commercial quality.[66]

Valloppillil's first memo provided a clear sense of how the projects collaborated through email and news groups. Since money was not a primary motivator for Open Source developers, the memo sought to understand their motivations. To do this, the author explored Raymond's "The Cathedral and the Bazaar" and "Homesteading the Noosphere." Valloppillil determined that frequently the motives were pragmatic – programs were produced to solve an immediate technical problem or to complete an academic assignment.[67] In discussing Raymond's "Homesteading the Noosphere," he also considered other possibilities, such as Raymond's ego gratification theory and Microsoft-bashing. The second memo provided a business analysis of Open Source software in general and Linux, Netscape, and Apache in particular.[68] The memo indicated that traditional tactics of spreading "fear, uncertainty, and doubt" (FUD) will not work with the community. Instead, it recommended that Microsoft study the methods of the movement and its ability "to collect and harness the collective IQ of thousands of individuals across the Internet." It also identified the fervency of the community as well as the quickness with which it spread, finding that Open Source "evangelization scales with the size of the Internet much faster than our own evangelization efforts appear to scale." The document also made specific internal recommendations for Microsoft to

[66] Vinod Valloppillil, *Open Source Software: A (New?) Development Methodology*

[67] Valloppillil incorrectly identified Linus Torvald's Linux effort as an educational project at the University of Helsinki. This was not the case.

[68] SWOT refers to Strengths, Weaknesses, Opportunities and Threats. This is a typical analysis undertaken within business for understanding the competition and for self-assessment.

consider, such as embracing distributed development processes and removing internal barriers between programmers. [69]

Unfortunately for Microsoft, the document also openly discussed the company's tactics. Working to "de-commoditize protocols & applications," meant taking common, commodity protocols and extending them with proprietary functions that made them incompatible with the products of other vendors. The document even goes as far as listing examples of the practice, such as requiring proprietary configurations to be added to Internet standard services like DNS in order to support Microsoft products like Active Directory.[70] The memo encourages this approach, as well as recommending the organization focus on installation automation "wizards" and other technical simplification programs that were generally ignored by Open Source programmers.

Raymond received a leaked copy of the memos on Halloween, 1998. He responded by publishing an annotated version of the text on his website. Most of his comments are picayune responses to the commentaries on his writings, but in his editorial, he does get to the crux of the matter: the memos show that Microsoft perceived Open Source software as a threat. This was both affirmation of the work being done by the programmers and confirmation that Microsoft was a foe. The detailed discussion of "fear, uncertainty, and doubt" tactics also confirmed long-held suspicions that Microsoft consciously used fear-based propaganda against competitors. Raymond also pointed to the anti-competitive actions discussed as common practices within the document,

---

[69] Vinod Valloppillil, *Open Source Software: A (New?) Development Methodology*

[70] With Microsoft Active Directory in Windows 2000, Microsoft began to add non-standard resource records to DNS to handle redirects to internal server roles such as LDAP, global catalogue, and primary domain controller servers. These additions caused issues when attempting to replicate with other RFC complaint DNS servers.

connecting them to Microsoft's tactic of "embrace and extend," to establish a monopoly lock through adding proprietary code.[71]

For the community, the documents were affirmation of long-suspected Microsoft practices. The behavior was not a surprise though: corporate malfeasance was the driving force in the creation of the GNU project that informed the Free Software Movement. For Raymond though, the documents helped focus the Open Source community on a common enemy. He subsequently made a point of targeting Microsoft in his rhetoric. For example, in his contribution to the 1998 book *Open Sources: Voices from the Open Source Revolution*, Raymond spends part of his chapter, "Revenge of the Hackers," jeering about the Halloween documents and prognosticating failure for Microsoft's technical endeavors. "Windows 2000 will not ship in a usable form," he predicted, stating that Microsoft's new OS would "either be cancelled or [be] dead on arrival. Either way it will turn into a horrendous train wreck, the worst strategic disaster in Microsoft's history."[72] Raymond's forecasts were wrong; Windows 2000, although not as svelte as its Windows NT 4.0 predecessor, was successful ,and the software formed the basis of Microsoft's subsequent server and desktop operating systems. Despite Raymond's anti-Microsoft rhetoric, the community demonstrated little apparent interest in an organized challenge to the company. Many of the existing popular software projects such as BIND and Apache, already dominated their market segments. There were software projects that acted as direct competitors to Microsoft products, but they existed long before the Open Source rebranding and Raymond's anti-Microsoft writings. More critically, the efforts were not

[71] Eric Raymond, *Open Source Software: A (New?) Development Methodology*.
[72] Eric Raymond, "The Revenge of the Hackers" in O*pen Sources*, 217-218.

designed to challenge Microsoft, but rather to address personal needs in removing dependencies on Microsoft technologies.

SAMBA

In January 1992, programmer Andrew Trigdell announced he was working on a software project that allowed UNIX-based computers to access file shares on Microsoft networks. Trigdell, a student at the Australian National University, had grown tired of having to work around the incompatibilities between the disparate technologies used at the institution.[73] His solution was to write software that provided interoperability with computers running Microsoft's Server Messaging Block (SMB) network protocol.[74] The software was ported to Linux in 1993, and it eventually evolved into the Free Software project, SAMBA (the name being a play on SMB.)[75] By 1997, SAMBA programmers had written software that could decrypt the user password store on Microsoft Windows NT, allowing SAMBA to be set-up to transparently replace an existing Windows file server.[76] By version 2.2, SAMBA was also capable of hosting a Windows Domain, with management done through native Windows NT domain tools such as User Manager for

---

[73] By the early 1990s, local area networks were common, but interoperability was a challenge as network software vendors used proprietary network protocols (e.g. Novell's IPX/SPX, Microsoft's Netbeui) and file sharing protocols (e.g. Novell's NetWare NCP, Microsoft's SMB.) The adoption of the Internet (and with it TCP/IP), helped standardize this, but even that was problematic early on as the network software vendors struggled to force TCP/IP within existing network frameworks (e.g. Novell tunneled TCP/IP packets through IPX/SPX and Microsoft patched together IP support based on the BSD TCP/IP stack and Berkeley socket derivative Winsock.)

[74] Andrew Tridgell. "announcing file services from non-DEC machines." Email to vmsnet.networks.desktop.pathworks newsgroup. January 10, 1992.

[75] Andrew Tridgell. "Netbios for Unix project." Email to comp.protocols.tcp-ip.ibmpc newsgroup. December 5, 1993.

[76] Jeremy Allison. "NT password -> smbpassword dump utility." Email to comp.os.ms-windows.nt.admin.security newsgroup. March 24, 1997.

Domains and Server Manager.[77] Despite having the ability to supplant Windows servers, the project team expressed little interest in packaging the application in a manner that would encourage its adoption by Windows administrators.[78] For example, to achieve functionality with tools such as Microsoft's User Manager, the individual setting up SAMBA would have to write additional Linux scripts that SAMBA would call upon to perform the required functions on Linux (adding a user or computer, resetting a password, etc.) As a result, any Windows administrator interested in moving to SAMBA would first have to learn UNIX/Linux scripting to use it.

Despite being a popular project with the ability to replace key Microsoft network services, the SAMBA effort emerged to address a personal technical issue. Through the life of the project, no deliberate steps were taken to make the software as a drop-in replacement for Microsoft products. The project's developers assumed their audience were users who were already savvy with Linux rather than neophyte Windows users. The skillset required by this assumption limited the software's adoption and suggested that the software was not developed to challenge Microsoft in the marketplace.

WINE

In the early 1990s Sun Microsystems released a product called Wabi. The software was an emulator that allowed Windows programs to be run on Sun Solaris. Programmers in the emerging Linux community were interested in having similar functionality on Linux. Since there was no reason to believe that Wabi would ever be

---

[77] A Windows domain (at that time) was a centrally managed system that performed user and device authentication.

[78] In contrast, with Windows NT Microsoft included tools that were designed to allow their software to replace or migrate workloads from their competitors, including Novell, Apple, and UNIX.

ported to Linux, programmers Bob Amstadt, Alexandre Julliard, and Miguel de Icaza began to write a similar program. Begun in July 1993, the team named their project WINE (Wine Is Not an Emulator). The software was quickly able to support 16-bit windows programs, like Solitaire, but Microsoft had already begun to transition to 32-bit computing. By 1996, the software could run 32-bit programs like Microsoft Office. Some Linux distributions did seek to use the product to run Windows programs, but the support for Windows applications was never full or complete.[79] Although the project was able to support a number of common office programs, the focus of the effort was chiefly to enable Linux users to play Windows-based games, as evidenced by the project's supported software list.[80] As with SAMBA, the project required technical skills for configuration and use. Users of the software had to install WINE and then from within the program, launch the Windows software that they wished to install. The program was then installed within an emulated file structure that resembled Windows. Since the program was not a full Windows implementation, many programs would not work without the user first locating and transferring over Windows library files and placing them in the correct locations. To utilize the software, an average Windows user might find he or she did not possess the requisite skills needed.

As with SAMBA, WINE was a popular project, but it was chiefly employed to address a personal technical issue, such as playing video games. The use of the project was frequently dependent on having a working Windows installation from which

[79] "Wine History," Wine.org, https://wiki.winehq.org/Wine_History, (accessed December 19, 2018).

[80] "Wine Application Database," Wine.org, https://appdb.winehq.org/, (accessed December 19, 2018).

Windows Library files could be retrieved. The project's developers assumed their audience was Linux users who were interested in running Windows software. Thus, the software was not developed to challenge Microsoft, but rather to address issues of program availability and compatibility within Linux.[81]

ReactOS

The project with the clearest targeting of Microsoft was ReactOS. Begun as the FreeWin95 project, the effort had the goal of creating a Free Software version of Windows 95. By 1997, the project, led by programmer Jason Filby, decided to instead focus on re-writing the Windows NT operating system.[82] As of this writing, the project still exists, but has demonstrated little progress, having garnered little interest from the Free Software community. Had the Free Software community been intent to supplant Microsoft, it would be reasonable to expect a project like ReactOS to have been more of a focus. However, over the past two decades, technical progress on the software has been slow, and ReactOS never emerged from test releases.[83]

Windows Refund Day

In the summer of 1998, a few Linux User Groups, such as the Silicon Valley Linux Users Group (SVLUG), began to use Microsoft's publicity events to promote Linux. For example, on June 25, 1998, they visited several retail stores for the midnight

---

[81] The Code Weavers corporation began to sponsor the WINE project in the early 2000s. The organization offers Free Software versions of their own custom implementation of WINE alongside a proprietary version of their software for running Windows programs on the Apple MacOS platform.

[82] "Reactos History," ReactOS.org, last modified June 4, 2012, https://web.archive.org/web/20120604192907/http://www.reactos.org/en/about_history.html, (accessed December 19, 2018).

[83] Colin Fink, "Reactos 0.4.1.0 Released," ReactOS.org, November 6, 2018, https://reactos.org/news, (accessed December 19, 2018).

release of Microsoft's newest operating system, Windows 98. Bearing protest signs, the group engaged waiting customers with flyers about Open Source software accompanied by Linux on CDs.[84] In February 1999, the Open Source community turned their protests toward Microsoft directly, demanding refunds for the unused versions of the Windows Operating Systems that had come preinstalled on their computers. At this time, Microsoft had software distribution agreements with all the major computer manufacturers including IBM, HP and Dell. Microsoft issued a version of its Windows operating system for these companies. This edition of the software was priced lower than retail versions, and in exchange, the licensing agreements resulted in a de-facto monopoly by requiring the hardware vendors to exclusively provide Microsoft's products.

On February 2, 1998, Australian Geoffrey Bennett purchased a Toshiba laptop. He was unable to find a computer without the Windows operating system, so he purchased one with Windows and immediately installed Linux. Bennett then read through Microsoft's End User Licensing Agreement (EULA) and found that it permitted the software to be returned for a refund provided that the consumer never used or copied it. Bennett contacted Toshiba to request his refund but was told "Toshiba is required by Microsoft contract to provide and purchase a valid operating system. Toshiba cannot get a refund from Microsoft." Bennett persisted and after being denied several times, was awarded his $110 refund on August 10, 1998.[85]

---

[84] Sam Ockman, "Svlug Rally," Penguin Computing, http://web.archive.org/web/19990224201149/http:/www.penguincomputing.com/svlug-rally.html, (accessed December 19, 2018).

[85] Joseph Gallivan, "SENDING WINDOWS BACK FOR A REFUND IS NO EASY FEAT," *New York Post*, January 24, 1999, https://nypost.com/1999/01/24/sending-windows-back-for-a-refund-is-no-easy-feat/, (accessed December 19, 2018).

Buoyed by Bennet's success, others began to request refunds. Microsoft rebuffed these attempts, suggesting that the EULA merely contained a legally required opt-out clause, and the required text was not intended to be an invitation for returns. As per Microsoft's license document however, the consumer could "contact PC Manufacturer for instructions on return of the unused product(s) for a refund."[86] Neither Microsoft nor the PC manufacturers had anticipated the customer requests. As such, they had no process for managing refunds. For consumers unable to receive the contractual reimbursement, it appeared that Microsoft was colluding with PC manufacturers to ensure that the contractually stipulated remuneration was impossible to receive. To protest this, Seattle-based webmaster Matt Jensen organized "Windows Refund Day" for Feb. 15, 1999. On this day, throughout the world, consumers that used non-Microsoft operating systems like Linux presented themselves to demand their refunds.

The Silicon Valley Linux Users Group saw one of the largest turnouts for the event, with about 100 members attending. The group met at a restaurant a mile away from the Microsoft offices in Foster City, California, and bearing protest signs, they walked to Microsoft. Raymond attended the event and drew a great deal of attention having dressed as the Star Wars character, Obi-wan Kenobi. Raymond and SVLUG member Chris DiBona attempted to lead the protesters to Microsoft's 9th floor offices but found that the company had closed the facility for the President's day holiday. Despite being closed, Microsoft, aware of the planned protest, set up tables at the top of their

[86] Chris Oakes, "The Non-Refundable Refund," *Wired Magazine*, January 22, 1999, https://www.wired.com/1999/01/the-non-refundable-refund/, (accessed December 19, 2018).

building's parking structure.[87] There the corporation offered the protesters free lemonade and handed out a carefully written note that acknowledge their interest in receiving a refund and informed the protesters that they "should contact their PC maker to address this issue." Microsoft's letter then suggested that the problem fundamentally came down to the consumer's choice in purchasing a computer with Microsoft Windows installed instead of finding a vendor who offered a PC without an operating system.[88]

As their memo suggested, Microsoft did not take the event seriously. Microsoft Windows Group Product Manager Rob Bennett, who distributed Microsoft's memo and their free lemonade, bemoaned the event to IT industry press, calling it a "publicity stunt." He mocked the Open Source advocates, saying they "bought Windows PCs…it's almost like being surprised that the cherry pie you bought has cherries in it."[89] To Microsoft, the personal computer and Windows were one and the same.

Schism

Two days after Raymond and other Open Source supporters protested Microsoft, Open Source Initiative (OSI) co-founder Bruce Perens sent an email to the Debian Linux developers mailing list. Raymond, President of the OSI, had been awarding Open Source certifications to corporations that did not produce software consistent with even the OSI's guidelines. Perens, who had authored the Open Source Definition had objected to Raymond several times, but with no success. Eventually, the OSI co-founder concluded

---

[87] "Windows Refund Day Monday, February 15, 1999," Silicon Valley Linux Users Group, http://www.svlug.org/events/refund/, (accessed December 19, 2018).

[88] *Windows Refund Day Letter* (Foster City, California: Microsoft, 1999), http://marc.merlins.org/linux/refundday/dear-valued-customer.html, (accessed December 19, 2018).

[89] Paul Thurrot, "Windows Refund Day Turns Into Sad Joke," *IT Pro Today*, February 15, 1999, https://www.itprotoday.com/windows-8/windows-refund-day-turns-sad-joke, (accessed March 14, 2015).

that under Raymond's leadership, the Open Source reinterpretation was focused on creating a product that appealed to corporations at the expense of Free Software's ideological focus on individual freedom. He was not alone; an increasing number of community members were concerned with the Open Source rebranding because it was bereft of the ideology that had informed the movement.[90]

"It's time to talk about Free Software again," Perens wrote in his February 17, 1999 email to the Debian mailing list. "Eric Raymond and I founded the Open Source Initiative as a way of introducing the non-hacker world to Free Software. Well, thanks to Eric, the world noticed." It was now time, Perens wrote, to return to Free Software. "Unfortunately," the programmer wrote, "Open Source has de-emphasized the importance of the freedoms involved in Free Software." The focus on Open Source "overshadowed the Free Software Foundation's efforts," resulting in "a schism between the two groups [that] should never have been allowed to develop." Perens said that he had objected to the rift and had worked to bring the camps together, but to no avail. Perens directly implicated the OSI President for the growing division. "Eric Raymond seems to be losing his Free Software focus," Perens wrote. "The Open Source certification mark has already been abused in ways I find unconscionable and that I will not abide. I fear that the Open Source Initiative is drifting away from the Free Sofware [SIC] values with which we originally created it."[91]

[90] Stephen Shankland, "'Open Source' Infighting Grows," *c|net*, February 19, 1999, http://news.cnet.com/Open-source-infighting-grows/2100-1001_3-221918.html, (accessed March 14, 2015).

[91] Bruce Perens. "It's Time to Talk About Free Software Again." Email to debian-devel@lists.debian.org Mailing List. February 17, 1999.

The responses from the Debian developers suggest that they were strongly aligned with the Free Software Movement rather than the Open Source rebranding. Several programmers welcomed Peren's return and many of them expressed concern over Raymond's dominance over the Open Source brand, as well as his self-promoting behavior. Norwegian developer Tor Slettnes criticized Raymond's recent behavior saying that "Open Source[tm]" had become associated with Raymond's publicity stunts and "tasteless anti-Microsoft agenda." Slettnes expressed concerns that Raymond appearing with Windows Refund Day demonstrators, "dressed up in Star Wars costumes," encouraged people to take software, such as Linux less seriously.[92] Another respondent, was particularly pointed in his criticisms of the Open Source leader:

> Eric Raymond is an evangelistic self-promoter who doesn't really care about Free Software, but rather just his own ego and popularity and is willing to sell his soul to Corporate America trying to convince them that he and he alone has the answer to the One True Path to destroy the Evil Empire, even if it means replacing it with another one.[93]

Others exhibited more politesse in their response. "I agree with your assessment of the Free Software/Open Source situation," wrote Michigan State University programmer Ben Pfaff. "I don't like the way that the Open Source trademark is being slung around and diluted these days.… I hope that the community has learned something from what has been going on."[94] These early concerns about the direction of Open Source foreshadowed the growing conflict within the Free Software community. The

[92] Tor Slettnes. "Re: It's Time to Talk About Free Software Again." Email to debian-devel@lists.debian.org Mailing List. February 17, 1999.
[93] Joseph Carter. "Re: It's Time to Talk About Free Software Again." Email to debian-devel@lists.debian.org Mailing List. February 18, 1999.
[94] Ben Pfaff. "Re: It's Time to Talk About Free Software Again." Email to debian-devel@lists.debian.org Mailing List. February 17, 1999.

divisions within the group would soon leave newsgroup discussions and spill out into the public sphere.

The Apple Public Source License

In March 1999, Apple Computers began to release programs under the custom Apple Public Source License (APSL), which had obtained an Open Source certification from the Open Source Institute through direct negotiation with Eric Raymond. The privately awarded certification was problematic as the license blatantly did not accord with the standards set by the Open Source Definition. The software that Apple released under the license was also not entirely their own. It contained Free Software written and released by the University of California, Berkeley and at Carnegie-Mellon University. After learning that the OSI Open Source certification had been awarded to a proprietary software license, several community programmers discussed the issue with Bruce Perens. As the author of the Open Source Definition, Perens contacted his former colleagues at the Open Source Initiative, but his concerns were dismissed. When he inquired about contact information for the license author at Apple, he was told that only Raymond had that information.[95] Unable to voice their concerns through private means, the programmers drafted a public letter to Apple.

In "The Apple Public Source License – Our Concerns," Perens, Debian Project Leader Wichert Akkerman, and the President of Software in the Public Interest, Ian Jackson, welcomed Apple's interest in the Free Software community and presented their concerns with the license. They politely explained that the issues within the license

---

[95] Bruce Perens, "Bruce Perens Replies to ESR's Letter," Free(code), April 6, 1999, http://freshmeat.sourceforge.net/articles/bruce-perens-replies-to-esrs-letter, (accessed December 19, 2018).

disqualified it as “Open Source(TM)” or “Free Software.” They detailed the offending sections of the license, Section 2.2(c) which required Apple to be notified of any modifications to the code, and Section 9.1 which allowed Apple to retroactively terminate the rights in the license. They also emphasized that the material composed at Berkeley and at Carnegie-Mellon University would be governed by their original licenses, rather than the APSL. At the close of the letter, they apologized for Raymond having granted the Open Source designation to a license that did not meet the criteria, and they invited Raymond to join them in requesting the changes.[96]

Speaking for the OSI, Eric Raymond immediately responded in a press release entitled “OSI clarifies the status of the APSL.” The statement dismissed the concerns by Perens et al, blaming the issue on their “erroneous readings of the license and of applicable common and statute law.” Raymond’s statement ignored the issue of Apple re-licensing Berkeley and Carnegie-Mellon University software. Instead, the release cited significant changes that Apple had been previously willing to make to the license before applauding Apple’s vision and lauding it as a model for other companies to follow.[97]

The APSL design essentially took contributions written by individuals within the community and at locations such as Berkeley and re-released them under stricter licensing requirements under Apple’s control. The OSI press release swept the concerns aside in an effort to curry favor with the computing manufacturer. For many in the Free

[96] Bruce Perens, Wichert Akkerman, and Ian Jackson, “The Apple Public Source License - Our Concerns,” *Linux Today*, March 17, 1999, https://www.linuxtoday.com/developer/1999031700205NWLF, (accessed December 19, 2018).

[97] Eric Raymond, “OSI clarifies the status of the APSL,” *Linux Today*, March 19, 1999, https://www.linuxtoday.com/developer/1999031900405NWLF, (accessed December 19, 2018).

Software community, the issue was a clear example of how the Open Source re-branding was trading individual freedom for corporate acceptance.

Within days, Stallman, whose own GPL licenses underwent trenchant legal analysis with each revision, reviewed the license. He deemed that the terms were incompatible with the GPL and other copyleft licenses and that it did not qualify as Free Software for three reasons:

1. Privacy – The license did not allow for modifications for private use without publication.
2. Corporate Control - Any published modifications required review by Apple.
3. Termination – The license allowed Apple to revoke access and forbid use of the software. [98]

In less than a week, the individuals who had defined the criteria for both Free Software and its Open Source reinterpretation had found the license to be inconsistent with both standards. Instead of revisiting the flawed license with Apple, Raymond ignored the issues and subsequently published a screed attacking his critics while extolling his own martyrdom. In "Take My Job, Please!" he complained about his role as the self-appointed leader of the Open Source Movement. He defined the importance of the position, asserting himself as the guard against Microsoft, writing, "Better technology can't win by itself without good propaganda…[;] no matter how clever and good and pure we are, we'll be swamped by the next billion-dollar marketing blitz from Redmond." He

[98] Richard Stallman, "Apple's Non-Free Source License," *Linux Today*, March 22, 1999, https://www.linuxtoday.com/infrastructure/1999032200105NWLF, (accessed December 19, 2018).

moved on to bragging about the position, the travel and luxury accommodation that it entailed and the exposure to famous people. “You're treated like visiting royalty by Linux user groups. You get to have your bio in *Wired* (next month). You get to make history. You even get groupies….”[99]

According to Raymond, the problem with the job was the critics in the community, “the tribe you've sweated blood to serve. Because they'll turn on you.” Raymond reversed the concerns expressed by the community and recast them as personal attacks. He defended his self-promotion as required in a “media-saturated world,” and dismissed his critics as having paranoid fantasies. “They'll attack your methods. They'll impugn your motives…. And it will hurt. Even though you know with the top of your brain that a lot of the hecklers are testosterone-poisoned adolescents acting out at your expense.” Raymond then assailed his critics as not having his “coding creds,” nor the “right personality type” to do his job. He then stated that anyone who wants the job could have it, but they’d have to meet specific criteria, such as to be credible with the community, have excellent speaking skills, an extroverted personality type and a “knowledge base that includes computer science, microeconomics and business practice, and more than a touch of anthropology/sociology/psychology. Bonus points if you can generate substantial research touching all these fields at once.” The job description closed with the rebuff, “If you can't meet these quals, maybe you'd better think a bit about whether you're helping get the job done or hindering it.”

[99] Eric Raymond, “Take My Job, Please!” catb.org, March 29, 1999, http://www.catb.org/esr/writings/take-my-job-please.html, (accessed December 20, 2018).

Raymond's approach received a mixed response in the community. Some took offense at the screed and his characterization of any would-be critics. "It bothers me that you use language in such a way as to produce the impression that anyone who disagrees with you, and says so publicly, is an enemy of the Linux community," wrote Slashdot reader Rob Warren.[100] "You're not the Fearless Leader, Mr. Raymond. The future of Free Software does not hinge on what you say and do, and if the only way to "win" is to fall in line (or "grow up" as you put it) and goosestep behind you, then let's all please declare defeat, go home and get on with our lives."[101]

Others in the community interpreted Raymond's article more sympathetically. Some perceived it as an indication that he had burned out. Discussions began about substituting Raymond with several community evangelists, with the thought that it would represent the movement more democratically than it had been under Raymond's tenure as Open Source's sole self-appointed leader.[102] Raymond immediately shut down any conversation about replacing him in a follow-up posting, "Understand My Job, Please!" where he made clear that he "cannot in good conscience retire without a replacement … ready to pick up my job." He then posited that no one would want the position anyway, since a new incumbent would not want to "worry about satisfying forty thousand

[100] Aleris, "Stereotypes, marytrdom and growing up.," Slashdot, March 31, 1999, https://linux.slashdot.org/story/99/03/31/1832226/understand-my-job-please-esr-explains, (accessed December 20, 2018).
[101] Ibid. Combine 131 and 132.
[102] Bruce Perens, "Do We Need a New Evangelist?" Perens.com, http://www.perens.com/Articles/Evangelist.html, (accessed December 20, 2018).

conflicted idealists nor have to listen to testosterone-poisoned twerps ranting 'who elected you?'"[103]

Raymond then got to the real intent behind his original posting: his rejection of the community criticism of the proprietary Apple license. He stated the event showed two "behaviors our culture needs to learn not to repeat." The first was that "People who should have known better (Perens/Akkerman/Jackson, RMS[Stallman]) threw bombs in public instead of approaching OSI and Apple privately with their concerns." The second was that the community itself reacted as if it was "a plot to sell out the hacker tribe to greedy corporate exploiters." According to Raymond, the behavior of the community and its other leaders made the possibility of renegotiating the APSL with Apple much more difficult.[104]

The issues were not personal, Raymond claimed; "we need to learn not to pull this kind of immature crap in public any more. The stakes are too high now." Open source was playing in the corporate world and had to be seen as capable of defining technical infrastructure. "I believe in the capacity of the hacker culture, coupled with the free market, to make it work. We can win," Raymond declares, before admonishing the community.

> Work on your kindness. Work on your trust. When you see twits going on a rampage, speak up against it without descending to their level. Try to be

[103] Eric Raymond, "Understand My Job, Please!," catb.org, March 31, 1999, http://www.catb.org/esr/writings/understand-my-job-please.html, (accessed December 20, 2018).

[104] Ibid. Raymond did not work with Apple and revise the Apple Public Source License. However, several years later, Apple produced an updated version of their license. APSL Version 2 was significantly re-written and was accepted both as an Open Source and a Free Software license.

> forbearing, not just towards me and your other advocates and leaders... but towards each other as well. [105]

Raymond's own forbearance did not last long. Five days after his article scolding the community for questioning his handling of the APSL license, Raymond responded privately to Perens via a threatening email. In the message, Raymond wrote that he took the license concerns personally and "if you ever again behave like that kind of disruptive asshole in public, insult me, and jeopardize the interests of our entire tribe, I'll take it just as personally -- and I will find a way to make you regret it. Watch your step." Perens shared the threatening correspondence with the Debian developers list, "because I know that Eric is a firearms enthusiast," Perens wrote. "[F]or my own protection, I feel the best strategy is for me to publicize the threat widely."[106]

Two months later, in June 1999, Raymond turned his ire toward Stallman and the Free Software foundation. In an opinion piece entitled, "Shut Up And Show Them The Code," Raymond attacked comments that Stallman had made distancing himself from the Open Source Movement. Raymond states that the difference between the Open Source Institute and Free Software Foundation was one of tactics, saying that the Open Source Movement was largely composed of people who rejected Stallman's rhetoric. The author justifies the Open Source Movement through its success in mainstream adoption. He dismisses Stallman's ideological focus on "computer users' rights" as "a dangerously attractive invitation to us to repeat old failures." Raymond credits the success of the Free Software Movement prior to Open Source as being due solely to the quality of the

---

[105] Eric Raymond, "Understand My Job, Please!" catb.org, March 31, 1999, http://www.catb.org/esr/writings/understand-my-job-please.html, (accessed December 20, 2018).

[106] Bruce Perens, "Email Threat," Email to lists.debian.org mailing list. April 5, 1999.

software produced by Stallman and others in the community. "So the next time RMS, or anybody else, urges you to 'talk about freedom," Raymond directed "…reply 'Shut up and show them the code.'" [107]

As with the other articles in this very public rift, the responses demonstrate the growing division within the community. Several of the comments support Raymond's perspective. Reader Matt Perry wrote, "I understand where RMS [Stallman] is coming from and agree with most of his goals. That doesn't mean that I'm going to preach radical software idealism be it under the name 'Free Software' or 'Open Source.'" One taciturn reader wrote, "I agree, code matters. Talk less and do more and better code. ESR [Raymond] is right on spot this time." Another reader encouraged Raymond, adding "FSF [Free Software Foundation] people seem to miss the point, that most of us want good software - not a revolution. And not everyone is opposed to the idea that sometimes someone may earn a buck or two instead of getting everything according to his needs (as Marks [SIC] proposed)." [108]

Several comments took a more neutral stance. One provided the pragmatic response, "RMS created a vision, he led the charge with some of the coolest code ever compiled…. However, without the buy-in from many large corporations (IBM, Dell, etc.) my corporation would never have taken OSI/FSF/GNU seriously." Reader Himi wrote that with the "Free Software/Open Source thing, I think a lot of people miss the fact that they are aimed at different people…. OSI is aimed at all those people who aren't in the

[107] Eric Raymond, "Shut up and Show Them the Code," *Linux Today*, June 28, 1999, https://www.linuxtoday.com/developer/1999062802310NWSM, (accessed December 20, 2018)..

[108] "Shut up and Show Them the Code Comments," *Linux Today*, June 28, 1999, https://www.linuxtoday.com/developer/1999062802310NWSM#comment_container, (accessed December 20, 2018).

know, who haven't heard of FSF and it's [SIC] philosophy. It's aimed at making those people aware that there is an alternative to the 'traditional' software out there." Reader Art Cancro shared the perspective that "Open Source people see freeness as a means to an end (that end being, as ESR puts it, "a world where software doesn't suck") while Free Software people see the freeness as an end-in-itself." [109]

Most of the responders disagreed with Eric Raymond. Leandro Dutra wrote, "You may be a good propagandist, but you are also a bull in a china store. Have you realized you just commanded RMS to shut up?!?" The comments of John Eikenberry got right to the ideological heart of the differences between Free Software and Open Source. "Eric claims the real difference between the FSF and OSI is not principles, but tactics. This is a load of crap.... [T]he real differences are the goals and the underlying principles driving these goals. The FSF's goal…is to increase freedoms." For other readers. Raymond's attempts to cast himself as the charismatic leader of the Open Source community were inconsistent with the anti-authoritarian ethos of the hacker community. "This is balderdash Eric…," wrote reader Louis Savain. "You just took the opportunity to jump on the coat tails of Linux's and GNU's success.… Long live free (and gratis) software! And I don't care if package prorietary [SIC] software manufacturers don't like the idea." Reader Joe Buck rejected Eric Raymond's approach in general; "you are trying to get us to emulate a corporation. Corporations speak with one voice, because they are dictatorships. Freedom means a cacaphony [SIC]of voices." Another commenter, Christian, shared frustrations with Raymond himself, "I'm seriously getting sick of ESR. Eric's shameless self-promotion, his emotional (and often agressive[SIC])

[109] Ibid.

public outbursts and his corruption of simple (but vital) ideals mean he is bad news for the Free Software community."[110]

Surprised by Wealth

On December 9, 1999, VA Linux, a Virginia-based company that sold free/Open Source-based computer systems, had their initial public offering. Opening at $30 a share, the stock posted a monumental 698 percent gain on its first day of trading. It was the largest NASDAQ IPO at the time.[111] The day after the company's record-setting IPO, Raymond, a VA Linux board member, published "Surprised by Wealth," an article in which he offered an ostentatious display of humility in reaction to his new fortune.[112] In the editorial, he justified his theoretical $36 million windfall and promised that he would be parsimonious with his newfound riches. Raymond also announced that he intended to start charging for speaking engagements, as this would "separate the expensive conferences that attract powerful people from the marginal events where the hacker community would get less leverage from my presence." Raymond made it clear that he would not entertain proposals to share his wealth with the community.[113]

---

[110] Ibid.

[111] Mark Gimein, "Dissecting the VA Linux IPO," *Salon*, December 10, 1999, http://www.salon.com/1999/12/10/va_linux/,(accessed March 14, 2015).

[112] Despite the record-breaking IPO, VA Linux quickly declined. A month after the offering, IBM announced support for Linux on all platforms. Other vendors quickly followed suit. Ironically, corporate acceptance of Open Source software undercut VA, whose core business was filling the gap created by corporations that did not accept Linux or Open Source by providing fully supported Linux-based computers. Within a year of the IPO, the stock was trading at just over $8.00 per share. VA quickly moved out of the hardware business and purchased Andover.net in 2000. At the time, Andover owned Linux.com, Slashdot, and Source forge, three of the websites most central to the Free/Open Source community. The organization also owned a geek-focused online retailer, ThinkGeek. By the mid-2000s, the websites had been sold, leaving VA (by this point, "Geeknet") with ThinkGeek as its only holding. This asset was sold to GameStop in July 2015.

[113] Eric Raymond, "Surprised by Wealth," *Linux Today*, December 10, 1999, http://www.linuxtoday.com/infrastructure/1999121000105NWLF, (accessed March 14, 2015).

Raymond's comments further polarized the Free/Open Source community, which was already divided by the Open Source rebranding. For those who sought corporate adoption, the VA Linux IPO suggested that the rebranding tactic was working. For those who supported Free Software on ideological grounds, these companies were seen as profiting off the fruits of the labor of countless volunteers. Since Free meant *libre*, not *gratis*, would there be remuneration for the efforts of the community? Stallman had formed the non-profit Free Software Foundation to pay programmers to write the GNU software.[114] Raymond's mercenary approach to his new-found wealth formed a sharp contrast to Stallman's community support.

The community division is visible in the responses to Raymond's article. While some commenters congratulated Raymond or agreed with his approach to wealth, the conflict within the community was clear in many of the postings.

> It is a real pity that the person who toiled to give us our most important Free Software, and wrote probably the very development tools used by Eric Raymond, has received little to nothing from the Linux boom…. I have no doubt the reason Mr. Stallman is being left out of the commercial success of Linux is because of his uncompromising beliefs in Free Software…. These vultures much prefer supporting Mr. Raymond's ideology because he does not speak out on crazy software patents and commercial software that traps people. [115]

> Did any of the public Linux companies remember to give a few ten million dollars' worth of shares to the Free Software Foundation? …After all, we can only thank the[m for] best utilities, compilers, glibc and [for] keeping the Free Software Movement alive for so long…. [116]

[114] The Free Software Foundation sold manuals and distributed copies of the GNU project software as a means to pay programmers to focus on developing code for the GNU project.

[115] "Surprised by Wealth Comments," *Linux Today*, December 10, 1999, http://www.linuxtoday.com/infrastructure/1999121000105NWLF#comment_container, (accessed March 14, 2015).

[116] Glibc is the GNU C compiler library. This is the standard C library and an important community project. For example, within Linux the operating system kernel and many of the applications

The responses to Raymond's various missives, Peren's emails, and Stallman's comments reveal a community split by the Open Source re-branding of the movement. Most of the perspectives, including that of Open Source Institute co-founder Perens, tended to identify with Stallman's ideological Free Software objectives. A smaller group identified with the Open Source's pro-corporate goals. These observable divisions in the community were also documented in a 2002 study done by researchers at the University of Maastricht in the Netherlands. Based on a survey of 2784 Open Source/Free Software developers, they found that 83% of the respondents agreed with Free Software's ideological objectives, citing that Free and Open Source Licenses were needed "to protect the freedom that software users should have."[117]

In 1997, the Free Software community had been harmoniously producing software and releasing it under various licenses with little concern for corporate approval or ideological self-identification. Within two years, the Open Source reinterpretation had fractured the movement. Instead of working on the technologies that brought the movement together, the programmers and their de facto leaders were engaged in internecine battles over corporate interests and ideology.

Summary: The Movement Fractures

In 1998, the Free Software Movement enjoyed increasing popularity due to the growth of the Internet and the relatively new Linux operating system. When the Netscape

---

are compiled from C. The explicit mention of the software emphasizes the criticality of the Free Software Foundation to the community.

[117] Rishab Ghosh et al., *Free/libre and Open Source Software: Survey and Study* (Maastricht,Netherlands: International Institute of Infonomics of the University of Maastricht, 2002), http://www.infonomics.nl/FLOSS/report, (Accessed 7/14/2020)

corporation expressed interest in releasing their web browser source code, several entrepreneurs associated with Free Software saw the event as an opportunity to sell the community to a larger audience. The group determined that a need to rebrand and, in the process, eliminate the ideological underpinnings of the movement to make it more palatable to corporations. Raymond took up the mantel of leader for the new Open Source brand and worked with others such as Perens to formalize and certify criteria for Open Source software. Raymond worked diligently to proselytize the rebranded movement to the media and corporations.

The Free Software Movement had been founded on Stallman's GNU project. The programming effort was a form of political activism, focused on ensuring the freedoms Stallman felt that programmers and software consumers should possess. In the GNU project, Stallman sought to create a body of applications apart from the corporate controlled software industry. The Open Source re-interpretation sought to re-brand and market Free Software to make it palatable to the very same corporations that Stallman started the effort to flee. Stallman, intractable in his convictions, distanced himself from the Open Source rebranding. Meanwhile, Raymond rejected Stallman's ethical rhetoric and was weary of the Free Software Movement's "ideological conformity" with Stallman.[118] He sought to use the Open Source re-branding to extirpate Stallman's ideology. He produced multiple treatises designed to minimize the ideological basis of the original movement and supplant it with his own free-market and libertarian beliefs. His rhetoric and behavior cast the software giant Microsoft, as the titular enemy of Open Source, and he began to publicly attack the corporate competitor.

---

[118] Eric Raymond, email to Chris Campbell, March 1, 2015.

Even as the community argued over the ideological deficiencies of Open Source and its self-appointed leader, corporate interest in the group's approach and technologies grew. Driven by the irrational exuberance of the late 1990s .com boom, investors flooded money into corporations that sold or used Open Source technologies. For companies like VA Linux and Google, Open Source software turned out to be the key to their success.

CHAPTER 7: FROM SOFTWARE TO THE STREETS

By the close of the 20th century, Open Source, the corporate-friendly re-articulation of Free Software, had gained popularity in the business world. This reinterpretation of the decades-old Free Software Movement resulted in an unintended side effect: a schism within the community. Eric Raymond, an Open Source advocate and self-appointed leader, deliberately sought to eliminate the ideology of Free Software, which he felt was off-putting to corporations. Instead, Raymond recast Free Software as a development model. This repackaging of Free Software as Open Source was successful, and the rebranded movement attracted corporate use and commercial investment.

The ideas and methods of the Free Software community also began to transition into other forms. Peer-produced efforts protected by copyright licenses emerged in projects ranging from using a distributed approach to legal representation to the creation of a free encyclopedia. At the same time, the commercial acceptance of Open Source and the successful transpositions of Free Software's ideals and methods also brought academic attention to the practices of the community. While anthropologists explored the programming culture, and economists debated its motivations, Harvard Law professor Yochai Benkler dissected the movement and documented how its schema was being imparted to other forms. In the process of defining Free Software's methods, he gave the schema a name: Commons-Based Peer Production.

Not all transpositions were positive. In the early 1990s, the U.S. government struggled to understand dial-up based computer communities. By this point, popular culture conflated cracking, the breaking into a computer system, with hacking, the practice of publishing clever software ‘hacks’ to fix problems. This misidentification also seemed to inform the actions of the federal government, which began seizing computers from individuals who they felt might somehow be involved with illegal activities, regardless of whether or not they had enough evidence to charge the person. To protect civil liberties from such actions, the Electronic Frontier Foundation was formed. The group attempted to guide lawmakers toward technically appropriate on-line legislation. These efforts ultimately failed, as demonstrated by the passage of the Digital Millennium Copyright Act of 1998 (DMCA). This law, which intended to bring the U.S. copyright system into compliance with international treaties, also served to make many Free Software programs illegal. The legislation also allowed programmers and researchers to be censored or arrested at the behest of commercial corporations.

Soon, programmers who developed such software were arrested under the law and corporations used the DMCA to stop scientists from publishing research that revealed security vulnerabilities with commercial products. Publications that provided -- or merely linked to -- material that corporations disliked were censored. Faced with an existential threat to their practices, the Free Software community struck back, boycotting companies, and protesting in the streets. With programmers no longer safe behind their computer terminals, the DMCA brought the Free Software Movement into the political realm.

Part 1: The Movement Succeeds

The Success of Open Source

The timing of the Open Source rebranding of Free Software was fortuitous. In April 1995, the Internet passed to commercial control, and as public awareness of the network grew, so did corporate interest in Internet-based technologies. Although many organizations simply sought to connect, establish email connectivity and perhaps a website, a number of new start-up companies emerged with the goal of leveraging and expanding access to the new network. A series of governmental changes also fueled investment in the burgeoning Internet technology market. The first was the Telecommunications Act of 1996. The law introduced a massive overhaul of the nation's telecommunications, de-regulating the industry to "let anyone enter any communications business -- to let any communications business compete in any market against any other."[1] The second change came a year later with the Taxpayer Relief Act of 1997. The law revised everything from tax credits to IRA limits, broadly reducing taxes. In particular, the law reduced rates for capital gains tax, the duties leveraged on profits made from investing.[2] With a steady demand for internet adoption, de-regulation of the telecommunication industry, and a reduced tax burden on profits made from corporate investing, the stock market experienced significant speculation on Internet-based companies. The success of early Internet companies, such as Netscape, only seemed to increase speculative interest. Netscape's initial public offering occurred on August 9,

---

[1] "Telecommunications Act of 1996," Federal Communications Commission, last modified June 20, 2013, https://www.fcc.gov/general/telecommunications-act-1996, (accessed April 2, 2019).

[2] "Taxpayer Relief Act of 1997," FindLaw.com, https://tax.findlaw.com/federal-taxes/taxpayer-relief-act-of-1997.html, (accessed April 2, 2019).

1995, just over a year after the company's inception. Netscape Navigator had been released only a few months prior to the IPO. The organization had yet to show a profit, but for investors, this did not seem to matter. At the onset, the IPO was set at $28 per share, but the price increased as soon as the market opened and peaked at $74.75 before settling at $58.25 at the closing bell.[3] The market valued the barely established company at nearly $3 billion, and in the process made Netscape's programmers instant millionaires.[4]

Netscape's IPO started the .com boom of the late 1990s, so when the company released their source code in early 1998, investors took notice of the new corporate-friendly Open Source Movement. Like a modern-day gold rush, prospectors took to the California hills to find their fortune. Open source software offered them the opportunity to do that. As detailed previously, the close relationship between the Internet and Free Software meant that any Internet-based company was already employing Free Software at some level, be it for hosting domain names with BIND or for serving web pages with Apache. But by the second half of the 1990s, an entire ready-built platform of tools for Internet hosting had emerged from the community. Commonly referred to by the acronym LAMP, the name described the toolset: Linux, Apache, MySQL, PHP/Python. Combined, the technologies allowed for the creation of dynamic, scalable websites.

By the mid-1990s, the Linux operating system and the Apache web server were already popular. The other two LAMP technologies emerged to fulfill the need for a Free

---

[3] David Streitfeld, "Bits Bits Business, Innovation, Technology, Society Back to the Future: The Netscape and Google I.P.O.'s," *New York Times*, May 17, 2012,https://bits.blogs.nytimes.com/2012/05/17/back-to-the-future-the-netscape-and-Google-i-p-o-s/, (accessed April 2, 2019).

[4] Alex Planes, "The IPO That Inflated the Dot-Com Bubble," *Motley Fool*, August 9, 2013, https://www.fool.com/investing/general/2013/08/09/the-ipo-that-inflated-the-dot-com-bubble.aspx, (accessed April 2, 2019).

Software database and for the creation of webpages on the increasingly popular Internet. MySQL was an Open Source relational database distributed by programmer Michael "Monty" Widenius in 1995.[5] Released in 1994 by Rasmus Lerdorf, PHP was originally "Personal Home Page Tools," written in the C programming language.[6] The alternative to PHP, Python, was a programming language written by Guido van Rossum in the early 1990s and subsequently released under a number of Free and Open Source Software licenses.[7] Both the PHP scripts and Python used database content to dynamically create websites. The platforms used the Common Gateway Interface (CGI), a webserver function that allowed the front-end websites to execute backend programs.[8] Where previous webpages had been manually created static html documents, CGI permitted real-time creation of dynamic sites with context-specific content such as search results.[9] This now allowed web pages to be interactive, with visitors participating in conversations or shopping online. With the free and Open Source LAMP suite, new start-ups instantly had a fully interactive platform on which to build their businesses. What is more, the software was also free of cost, lowering the barrier of entry for pre-capitalized Internet startups.

Existing companies also realized the benefits of adopting Free and Open Source software. With increasing numbers of households going on the Internet, networking

---

[5] Patrycja Dybka, “Mysql: Why ‘my’?” vertabelo.com, August 14, 2014, https://www.vertabelo.com/blog/notes-from-the-lab/mysql-history, (accessed April 2, 2019).

[6] “History of PHP,” Php.net, https://www.php.net/manual/en/history.php.php, (accessed April 2, 2019).

[7] “History and License,” Python.org, last modified April 2, 2019, https://docs.python.org/3/license.html, (accessed April 2, 2019).

[8] Front end, in this context means client facing. This originally would have meant HTML, but following Netscape’s 1995 introduction of JavaScript, this language frequently replaced html due to its flexibility. Microsoft also borrowed Netscape’s JavaScript approach, reverse-engineering the language and releasing it as their own Jscript platform. Both Browsers and webserver front ends featured JavaScript. The later employed worked with CGI platforms such as PHP and Python to serve automatic and interactive content.

[9] The CGI interface originated with the NSCA webserver software and was later documented in RFC 3875 by Apache Project members Ken Coar and David Robinson.

companies like Linksys and Cisco began to create home networking products based on Free Software. By adopting these technologies, companies like Cisco benefitted from a free operating system on their routers as well as additional integration technologies. Because many Free Software community projects focused on compatibility, home networking routers could provide advanced features. For example, devices running Free Software such as SAMBA could also work as file server.[10] Since the software was already written, the costs to add a feature were minimal – all the corporation needed to do was follow the software's licensing stipulations.[11]

Google: A Case Study

The role of Open Source in the success of the .com boom businesses can be seen in one of the juggernauts from the period, Google. The company grew out of a world wide web indexing project begun in 1996 by Stanford University Ph.D. students Larry Page and Sergey Brin. By 1998, the duo had founded Google to support their efforts. The company chose the name to reflect their ambition to create large-scale search engines – a googol is the name for 10^{100} – their ambition for their database size, which at the time of Google's founding consisted of 24 million web pages. [12]

The company begin to assemble servers, which were originally hosted in Larry Page's living room, to serve up their burgeoning platform. They used low-cost commodity personal computer hardware to run a variety of Open Source software, to

---

[10] Network Attached Storage or NAS, is centralized, network-based file storage made available to computers on the network via file sharing protocols such as SMB, NFS, FTP, and CIFS.

[11] Jack Schofield, "FSF Sues Cisco Over Linksys Open Source Code," *Guardian*, December 12, 2008, https://www.theguardian.com/technology/blog/2008/dec/12/cisco-fsf-opensource, (accessed April 2, 2019).

[12] Ken Auletta, *Googled: The End of the World as We Know It* (New York, New York: Penguin Press, 2009), 38-40.

which they began to make proprietary changes. Google ran modified versions of the Debian and Ubuntu Linux distributions.[13] The Google website was primarily JavaScript served via the Google Web Server (GWS), a modified version of Apache.[14] The Google search engine and the web-crawling spider that informed it used the Python programming language.[15] The databases used the Open Source MySQL platform.[16] The low cost and extendable nature of the Open Source platform allowed Google to scale without incurring considerable expense.[17]

The core of Google's technology was their ability to index the sites on the world wide web and rank their relevance. To do the latter, Page and Brin wrote an algorithm that went beyond the key word searches common on the Internet at the time of Google's incorporation. The Google search engine analyzed the number of links connecting to a given site as well as its outbound links to other sites to create a page ranking. In addition, the frequency of user visits to a site also increased its ranking in the search results. This design relied on visitor behavior to determine relevance to the search. In 2000, Google began to monetize its search engine with the introduction of Google Ad-Words, which allowed advertisers to pay for their site to be presented alongside Google's search results. The company introduced the idea of Cost-Per-Click, where advertisers only paid if their content was accessed. They also automated the entire system, eliminating Google's

[13] Steven Vaughan-Nichols, "The Truth About Goobuntu: Google's In-House Desktop Ubuntu Linux," Ziff Davis, August 29, 2012, https://www.zdnet.com/article/the-truth-about-goobuntu-Googles-in-house-desktop-ubuntu-linux/, (accessed April 2, 2019).

[14] "Site Report for Www.Google.com," Netcraft, last modified March 31, 2019, https://toolbar.netcraft.com/site_report?url=http://www.Google.com, (accessed April 2, 2019).

[15] Andrew Sangster, "Organizations Using Python," Python.org, July 2, 2013, https://wiki.python.org/moin/OrganizationsUsingPython, (accessed April 2, 2019).

[16] Brian Proffitt, "Google Waves Goodbye to Mysql in Favor of Mariadb," ReadWrite.com, September 14, 2013, https://readwrite.com/2013/09/14/Google-waves-goodbye-to-mysql-in-favor-of-mariadb/#awesm=~omBpxaeC3cblHt, (accessed April 2, 2019).

[17] Auletta, 321-324.

overhead in managing their advertising clients. In 2003, the company introduced Google Ad Sense. Ad Sense was a system where websites could host Google generated banner ads for products. The advertising was context aware: an individual who searched Google looking for hiking boots in the morning would later see hiking equipment in banner ads in the afternoon at unrelated sites. This was due to "behavioral targeting," based on data that Google collected and stored on its users to target them for precision advertising. In April 2004, Google expanded their surveillance advertising approach with the introduction of their Gmail Email service. The service provided users with free email, including 1 GB of storage. The application discouraged the deletion of old emails, so much so that the service originally lacked any means to delete messages. Having this content available afforded Google more information for data collection and ad targeting. Ad sense could now analyze the user's correspondence and present topically relevant ads to what they were discussing.[18]

In many ways, Google epitomized the role of Open Source in the late 1990s. It heavily relied on Free Software, and it freely made changes to the software to suit their needs. They also worked the with community; prior to their foray into online advertising, one of their first contracts was with an Open Source company, Red Hat Linux.[19] Harvard Law professor Yochai Benkler also identified the influence of Open Source in the company's page ranking software, which "employs peer production of ranking" in that the web activity of previous site visitors increase the relevancy given to a website.[20] "By doing this," Benkler observed, "Google harnessed the distributed judgments of many

[18] Ibid., 38-39,63,89,187,99..

[20] Yochai Benkler. "Coase's Penguin, Or, Linux And The Nature of The Firm," *The Yale Law Journal* 112, no. 3 (December 2002): 392.

users…each judgment created as a by-product of making his own site useful." Like the peer-based construction of Open Source projects, Google employed the web's users in the peer production of information. Through their use of Google's search software, users constructed Google's ranking of the page's relevance. At the same time, by utilizing Google's search engine and Gmail, the users also created a portfolio of their activities for Google to leverage for advertising revenue.

Google also demonstrated the difference between Open Source and Free Software. The company employed Open Source software and the production schema used by both Open Source and Free Software. But the organization lacked any ideological relationship with Free Software. Where Google used and modified Open Source programs, the result was private, proprietary software, such as the Google Web Server, which was not shared with the community. Google was far from transparent in other areas as well; in fact, Google's model reified the concerns expressed by Stallman in the early 1980s. The company's business model was based on the surveillance of user activities. Just as Symbolics had done to Stallman, Google essentially spied on their users. Google analyzed the emails and searches performed by users and later leveraged the information garnered from this surveillance to provide advertising clients with the ability to target ads. Stallman's invocation of Kantian ethics resounded with the consideration of Google's practices. Immanuel Kant's categorical imperatives required that individuals needed to treat humanity, "never merely as a means to an end but always at the same time as an end." With Google, the users were the product. The organization manipulated its consumers into creating the user data that buttressed Google's search and advertising algorithms. According to Kant, this sort of manipulation of individuals to achieve a goal was immoral.[21] Google was an Open Source company, but it lacked the ethical grounding and ideology inherent to Free Software.

Academic Interest in Open Source

By 2000, the "irrational exuberance" that had fueled market speculation in Internet technology companies had begun to subside.[22] On April 4, 2000, Microsoft was found guilty of violating anti-trust laws and of exhibiting predatory behavior by attempting to monopolize the web browser market through "unlawfully tying its web browser to its operating system."[23] Although no remedy was announced at that time, the decision concerned tech investors in the market. Combined with the Federal Reserve's decision to increase interest rates and unexpected tech stock selloffs, investors became jittery.[24] The tech-stock heavy Nasdaq index, which had peaked at 5048 just weeks before on March 10, 2000 began to fall as investors shed technical investments. The index reached its nadir at 1423 the following year when the markets re-opened after the September 11, 2001 terrorist attacks on the United States.[25] Investor confidence was eroded further the following month with the spectacular failure of the Enron corporation

---

[21] Immanuel Kant, *Ethical Philosophy: The Complete Texts of Grounding for the Metaphysics of Morals, and Metaphysical Principles of Virtue,* 2nd ed. (Indianapolis, Illinois: Hackett Pub. Co., 1994), 32-26.

[22] "20 Years Already? Alan Greenspan and the 'irrational Exuberance' Flop," Market Watch, December 6, 2016, https://www.marketwatch.com/story/20-years-already-alan-greenspan-and-the-irrational-exuberance-flop-2016-12-05, (accessed April 2, 2019).

[23] Joel Brinkley, "U.S. Vs. Microsoft: The Overview; U.S. Judge Says Microsoft Violated Antitrust Laws with Predatory Behavior," *New York Times*, April 4, 2000. A1.

[24] "Greenspan Warns On Rates," CNN, February 17, 2000, https://money.cnn.com/2000/02/17/economy/greenspan/, (accessed April 2, 2019).

[25] Jake Ulick, "Not Your Father's Nasdaq," CNN, March 19, 2002, https://money.cnn.com/2002/03/19/markets/nasdaq/index.htm, (accessed April 2, 2019).

and the crimes that were subsequently revealed.[26] The corporate accounting scandal was the first of several that undermined confidence in the stock market, resulting in a prolonged recession.

Despite the end of the artificially favorable market conditions, some Open Source-based corporations such as Google still experienced growth. In 2001, during the tech market slump, Google's revenue was $86 million. By 2003, it increased to $1.5 billion and was on target to double by 2004. In the spring of that year, Google announced its planned IPO. On August 19, 2004, Google's stock was offered at $85 per share. By the next day it was trading at $108.31 and by the next quarter, the shares were above $200.[27] Google proved that it was possible to base a business on Open Source technologies and be successful.

Before early 2000s, most of what was written about Free and Open Source software appeared in either tech journals or industry publications. Journalists such as Glyn Moody, who had written for *Computer Weekly*, *Linux Journal*, and *Wired*, reported positively on the emerging movement in books including *Rebel Code*, which was Moody's hagiographic account of Linux and other rising Open Source projects.[28] Other industry considerations of the movement were less positive. Writing in the Institute of Electrical and Electronics Engineers (IEEE) software magazine, programming industry veteran Robert Glass offered an enthusiastic opposition to the Open Source Movement. Open source, he argued, would be ephemeral, "an interesting chapter in software's

---

[26] C. William Thomas, "The Rise and Fall of Enron," *Journal of Accountancy*, April 1, 2002, https://www.journalofaccountancy.com/issues/2002/apr/theriseandfallofenron.html, (accessed April 2, 2019).

[27] Auletta, 108-109.

[28] Glyn Moody, *The Rebel Code: The Inside Story of Linux and the Open Source Revolution* (Cambridge, Massachusetts.: Perseus Publishing, 2002).

history, but hardly a ground-rules changing one." Any changes brought by the movement, Glass argued would "be limited to one or a few cults emerging from a niche culture of the software field." Glass's assessment was based on his experience in the industry:

> I cannot imagine that the software field will experience a groundswell change toward the maintenance, reading, and study of software for no financial reward, away from the traditional importance placed on development and money. [29]

Robert Glass' disbelief was founded on the misperception that software had always been a marketable commodity. Glass's comment offered little insight about Open Source; instead it revealed that he believed that corporate dominance of software was complete and immutable. Glass's personal perspective may be debatable, but his comment showed why Open Source was beginning to pique so much interest – it opposed the ordinary understanding of how software should work. By the early 2000s this apparent inconsistency brought Open Source to the attention of economists, political scientists, and anthropologists – all of whom sought to understand the phenomenon unfolding in the software industry.

In their 2002 paper, "Some Simple Economics of Open Source," Harvard Business school economist Josh Lerner and Nobel Prize-winning French Economist Jean Tirole explored the emerging Open Source Movement through a survey of four projects: Apache, Linux, Perl, and Sendmail. In their exploration of the projects, the researchers sought to determine who contributed and how they benefited from the activities. They opined that the burden on Open Source developers who contributed was in the

---

[29] Robert Glass, "The Sociology of Opensource: Of Cults and Cultures," *IEEE Software* 17 (May - June 2000), 105.

opportunity cost -- the time spent contributing could have been spent doing something else. The authors revealed that the practice provided immediate benefits: the coding may have been done to address issues that an employer had, and the practice might have been enjoyable for the programmer if the work was being done on a "cool" project. They also highlighted the possibility of delayed rewards such as shares in a commercial Open Source Software company. They explored Raymond's theory of ego gratification, but where they discussed peer recognition, they also dismissed the role of the community, stating that identity with those working on Open Source projects did not seem to apply since the developers would have had an equal opportunity to "be part of the team" at work. The researchers also considered the nature of Open Source projects, determining that in order to be successful efforts must be divisible into distinct components and to pique interest in participating, a project leader must first produce and share functional code. For corporate interactions with Open Source, they identified three models. Corporations could live symbiotically off a project, as evidenced by Red Hat Linux. Alternately, a company could elect to release existing software as Open Source, as Netscape did. Finally, corporations could create a niche as an intermediary that certified and supported existing Open Source software. They closed their paper with the considerations that remained in the industry, ending with the poignant question – could the ideas behind Open Source be transposed to other industries?[30]

Many of the topics raised in Lerner and Tirole's paper were further explored by University of California, Berkeley Political Scientist Steven Weber. The Berkeley professor issued a series of papers that were eventually incorporated into his 2005 book,

---

[30] Josh Lerner and Jean Tirole, "Some Simple Economics of Open Source," *The Journal of Industrial Economics,* 50, no. 2 (June 2002). 213-220, 224-230.

the *Success of Open Source.* As one might expect from a Berkeley researcher, Weber provided a fine history of UNIX that highlighted Berkeley's many contributions in the BSD distribution. However, the author declared that his purpose was not to provide a history, community ethnography, or attempt to discuss philosophies regarding intellectual property or copyrights. Instead, he worked heavily with Raymond's writings on Open Source as a software development process and he discussed its success, which he defined as its efficacy as a process and the abilities of businesses to leverage this model. This being the case, he sought to account for economic logic, social structures, and individual motivations involved with this process. Consistent with his business-centric focus, the author dismissed the role of ideology or ethics. In his discussion of individual motivations, for example, he stated that the motivation for "many Open Source developers" was doing battle with a "joint enemy," which he suggested was Microsoft. He referenced a Boston Consulting Group Survey that found 11.3% of respondents indicated this Microsoft response was the primary motivation for their work. However, he privileged this answer over other responses to the same survey question. Direct examination of the survey results revealed that 34.2% of respondents reported an ideological motivation, in that "code should be open."[31] The author acknowledged that this reflected an ideological commitment, but he dismissed this data as inconsistent with his "observed practices of most Open Source users." Not surprisingly, Weber's conclusions mirrored the writings of Raymond stating that the eponymous success of Open Source was in meeting Raymond's objective: corporate acceptance.

[31] Weber. 225, 139.

Open source elicited interest from academics in the humanities as well. Although entirely too recent for historical analysis, anthropologists began to explore the movement. In *Two Bits: The Cultural Significance of Free Software*, anthropologist Christopher Kelty explored the Open Source culture as it was in the early 2000s. [32] The manuscript comingled history with cultural analysis and his ethnographic field work experiences throughout the world.[33] The conclusion of the globe-trotting anthropologist was that the Free Software/Open Source Movements were "an international community of geeks brought together by their shared interest in the Internet." Kelty determined that the movements formed a recursive public, "a collective independent of other forms of constituted power." However, in contrast to Jürgen Habermas' conception of a public sphere, the author suggested that these "geeks" were not concerned with societal transformation.[34] Instead, the "geeks" simply came together to "build, control, modify, and maintain the infrastructure that allows them to come into being in the first place."[35]

Kelty's historical context was based strongly on the writings of Eric Raymond, acting as the anthropologist's "principal informant," as well as established secondary sources: Janet Abbate's Internet history, Peter Salus' UNIX history, and Glyn Moody's Rebel Code. Unlike Weber, Kelty's account was not satisfied to leave Open Source as a development model, but his examination was quick to downplay the relevance of

[32] Kelty, 30-31.

[33] Kelty's research provides a sense of the aesthetic associated with hackers in the late 1990s and early 2000s. By this point, the concept of hacking (understood commonly more to mean the cracking into a system), had been celebrated in movies such as the 1995 film *Hackers*, and the 1999 film *The Matrix*, the latter being a blend of the philosophies of Jean Baudrillard and a typecast of hackers as ultra-modern techno enthusiasts. Kelty's interactions are as much a statement on the conventional cultural conception of hackers as they are one on the individuals themselves.

[34] Kelty uses the term "geek" throughout his text. He seems to be unaware of the pejorative use of the term, which suggests the subject to be unfashionable and/or socially inept. This choice seems particularly questionable considering that Kelty is anthropologist; it would be akin to studying a racial group and referring to its members by a racist epithet.

[35] Kelty, 3-7.

Stallman and Free Software, with the author concluding that "before 1998, there was no movement." The author refuted Stallman's 1983 concerns with the emerging software industry, determining that "Free software is not an ethical stance" but a solution to problems with "legitimacy of the means of circulation of knowledge." According to the author, the progenitor to Stallman's GNU efforts, UNIX, "was never outside the mainstream" software market – a curious conclusion since when the software emerged it was not sold in said market – but discretely licensed without support. Likewise, he claimed that UNIX was "never free in any sense," rather it set the stage for what "free" would become in the 1980s and 1990s.[36]

Kelty recalled the schism that had occurred with the Open Source rebranding. "Stallman was denounced as a kook, a communist, an idealist, and a dogmatic holding back the successful adoption of Free Software by business," Kelty explained, where Raymond and his Open Source adherents "were charged with selling out the ideals of freedom and autonomy." However, the anthropologist was an apologist for the later prospective, pointing to the VA Linux IPO as proof of the reinterpretation's success while dismissing Free Software ideologues as "kindling flames of worry over intellectual-property expansion," in their resistance to the 1988 Digital Millennium Copyright Act.[37] However, despite his alignment with Raymond's conception of history, by the time of Kelty's field research in 2002, the anthropologist perceived Raymond as representing the movement's past.[38] The author recounted a dinner at Rice University where he planned to correct Raymond's anthropological misconceptions. Raymond

[36] Ibid., 19,99,306-308.

[37] Ibid., 116, 99.

[38] It is important to note that Kelty tends to use the term "Free Software" indeterminately and even when referring to the Open Source reinterpretation of the movement.

snubbed the researcher and instead lavished his attentions on the only female present. This left Kelty to converse with Richard Baraniuk and Brent Hendricks, fellow researchers who were crafting the Connexions project, an effort to create a repository of open content licensed educational materials. Kelty, who believed Free Software to be "an experimental system, a practice that changes with the results of new experiments" had an epiphany. "All at once, Raymond looked like the past of Free Software, arguing the same arguments, using the same rhetoric of his online publications," Kelty concluded, "while Baraniuk and Hendricks looked like it's [sic] future."[39] Kelty was interested in the Connexions project because it represented a transposition of the schema used by Free Software to a project outside software production.

Kelty's research divorced the movement from its ideological origins, due to his belief that the movement began with the advent of Open Source. At the core of Kelty's reasoning was the belief that since the adherents held differing perspectives, it was their practices, the schema of software production, which mattered – and not their ideological differences. As a result, the anthropologist focused on the movement's procedural descendants and ignored the incipient transposition of the Free Software ideology to other aspects of society.

By the time of the publication of *Two Bits*, the transposition of the Free Software schema that Kelty was keen on observing had already been given a name – Commons Based Peer Production. In his 2002 article, "Coase's Penguin, or, Linux and 'The Nature of the Firm,'" Harvard Law professor Yochai Benkler sought to reconcile the practices of the Free Software/Open Source community with traditional economic models. For his

[39] Kelty. 254,239,244.

analysis, Benkler chose a methodological template based on the work of Nobel Prize winning economist Ronald Coase. In 1937, *Economica*, a general economics journal, published Coase's "The Nature of the Firm," a paper that explained why firms (defined by Coase as "clusters of resources and agents that interact through managerial command systems rather than markets") emerged and grew. Coase argued that there were transaction costs associated with determining and enforcing contract and property rights within the marketplace. According to the economist, the appearance and longevity of a firm was relative to the transaction costs. Managed firms come into existence when the costs of organizing production through the market were higher than the transaction costs in creating a managed firm. Once created, an individual firm could grow until its costs exceeded the organizational costs of a smaller firm.[40] Simply put, if we assume the efficiencies of the market (low costs for a product due to competition, established production, etc.) it makes sense to acquire goods and services on the market. However, once the transaction costs (composing and enforcing contracts, protecting trade secrets, patents, etc.) increase the costs of goods or services to the extent that it would be less expensive to produce the goods (or fulfill the services) directly, it made more sense to pursue an entrepreneurial solution and create a firm.[41] The newly created firm would grow until its costs exceed that of a smaller firm, which, would then make a market-based solution more economical.

For Benkler, Free Software/Open Source practices represented a puzzle, since the software developers participated in the project outside of the market/firm-based model. The projects did not rely on the market or hierarchical management to organize

[40] Benkler, 372-373.
[41] Or, assumedly in the case of an established firm, pursue vertical integration of production.

production, and the programmers did not perform work because they were directed to by a boss. Rarely did the developers participate on a project because of any financial remuneration, and the decisions made on the projects did not seem to accord to any known models.[42] To understand the framework employed by the community without getting caught up in the specifics of software development, the economist stepped away from the act of programming itself and instead looked at medium and large scale collaborations associated with the community.

Benkler's analysis began with the broader context in which these collaborations had emerged. He argued that the declining costs of data storage and communication had "inverted the capital structure of information and cultural production." Previous technologies such as printing presses, radio, records, television, and film required a high expenditure of capital to organize, produce, and disseminate cultural artifacts. Advances in technology eliminated the overhead of this capital-heavy industrial approach, resulting in a situation where the "primary remaining scarce resource is human creativity." This led the economist to consider the motivations of participants. Here, he made two determinations. First, since human beings are diverse, there are a myriad of motivations for human actions beyond the acquisition of money. "There exist ranges of human experience in which the presence of monetary rewards is inversely related to the presence of other, social-psychological rewards. The interaction between money, love, and sex offers an obvious and stark example." He determined that there will be conditions under which the projects might offer social-psychological rewards beyond money that would attract certain individuals where the promise of monetary rewards would not. Second,

[42] Benkler, 372-373.

Benkler suggested that when a project was broken into small enough segments, the effort would be much smaller, and as such "the motivation to get any given individual to contribute need only be very small."[43]

Having established the environment and considered motives, Benkler then articulated a framework he refers to as "Commons Based Peer Production." He identifies several commonalities within the efforts he observed employing this method of production. First, projects were modular and each of the elements within are able to be produced independent of each other. This allowed for asynchronous, incremental advancement, which could combine the skills of "different people, with different capabilities, who are available at different times." Second, the more modular the effort was, the larger the number of contributors and the more likely the effort was to see individual participation. This was consistent with the second proposition for motivations to contribute: the smaller the effort required, the smaller the motivation needed to participate. Third, peer production efforts needed methods for low cost integration. On the technical front, this would be a "mechanism for integrating the contributions into the finished product," an arrangement such as a software repositories or content versioning system. However, Benkler determined that these methods had to also include the ability to "defend itself from incompetent or malicious contributions." This included quality control and the ability for the project to protect itself. For quality oversight, the economist reasons that the size of the project would correct for any issues of incompetence: the larger the project the more likely that poor contributions would be statistical outliers.

[43] Benkler, 372-373, 378-379.

Benkler discussed both the social norms within the community and the importance of having the protections of a structure like the GNU Public License (GPL).[44]

To support the framework, Benkler then looked at practices of the user community outside of the development of the Free and Open Source programs. He considered corporations that used Open Source software, such as Google and Amazon, as well as community websites such as Slashdot, and demonstrated how peer production factored into their practices.[45] The corporations integrated peer production structures into their operations; peer data production afforded Google its search accuracy and targeted marketing. For Amazon, peer behavior on the site informed the algorithm that recommended associated products. With Slashdot, the content was peer submitted, and the core of the site comprised of peer interaction.

Benkler's research was important for several reasons. First, he provided a framework divorced from the practice of software development. Next, he looked at the consumers of the software and perceived that the schema extended beyond the composition of software and was extrapolated to different forms within the community. Finally, he provided an answer for the timing of the phenomenon. Previously the dissemination of creative endeavors was locked into governmental or commercial structures with the capital necessary to organize, produce, and disseminate ideas through print, radio, television, and film. The required infrastructure presented barriers to entry for the individual and afforded a monopoly of control to those who could produce and disseminate media. The production of software was not capital intensive, and recent

---

[44] Ibid., 379-380.

[45] For a more detailed conception of Google's use of these constructs, review the Google section from earlier in this chapter.

advances in technology, such as the Internet, facilitated the distribution of content. This eliminated the control that traditional power structures had over the dissemination over individual expression.

The Success of Free Software: Transposition

The events that led to Stallman's GNU manifesto and the birth of the Free Software Movement are easier to comprehend when viewed through Professor Benkler's framework. Leveraging changes to copyright and to the cultural norms within computing, the commercial software industry developed a hegemony over software production. Stallman perceived this to be accompanied by a growing pattern of corporate abrogation of the rights of the individual. But where corporations required active coordination of many people and resources to create a structure of production and dissemination, Stallman worked alone to produce and propagate an alternative to corporate control. Technology was democratizing: the subaltern programmer now had the same opportunities afforded to the hegemonic corporations. Stallman developed a corpus of programs as well as a licensing scheme that used copyright law to protect his software and any derivative works. The GNU software itself was intended as a method of transmitting the ideas that informed its creation.[46] This is why Stallman had spent so much of the GNU manifesto discussing ways to make money with his software. The object of the effort was not to just oppose capitalism – it was to propagate a world view. Interpreted correctly, Free Software acted as a clarion call for the likeminded to protect their civil liberties in the digital realm by creating structures separate from mainstream

---

[46] Leonard Tower. "Linux and the GNU system by Richard Stallman." Email to gnu.misc.discuss,comp.os.linux.misc,comp.os.linux.development.apps,comp.os.linux.advocacy mailing lists. May 31, 1996.

commercial control. The intention was the creation of a public sphere, societal transformation through technology.

The Open Source re-interpretation of Free Software sought to remove this ideological payload by narrowly defining the community's activities as a software development process and not an instrument with which to propagate civil liberties. For Raymond, his motivation with Open Source was to allow programmers to seize control of production from the "incompetent or evil bosses" who managed development. "If it takes seizing control of the craft of programming back from the suits, that's what it takes. If it takes blowing the *entire system of production* to smithereens…well, then it's long past time."[47] Raymond's comments, aside from being ironically redolent of Marxism, demonstrated the limitations of Open Source: since it is only a *software* development model, it does not readily transpose to other forms or mediums. In the Afterword to the printed edition of Raymond's *The Cathedral and the Bazaar*, the author considered the potential spread of the development model to other areas such as music, books, or hardware. Raymond conjectured that it would be possible for Open Source software development model to be adapted to other arenas, but it would offer little benefit as "music and most books are not like software, because they don't generally need to be debugged or maintained. Without that requirement, the utility of peer review is much lower, and the rational incentives for some equivalent of open-sourcing therefore nearly vanish."[48] Despite Raymond's reservations, the ideas and practices behind Free

[47] Eric Raymond, "Why I Hate Proprietary Software," esr.ibiblio.org, October 1, 2008, http://esr.ibiblio.org/?p=556, (accessed April 10, 2019).

[48] Eric Raymond, *The Cathedral and the Bazaar*, catb.org, August 16, 1999, http://www.catb.org/~esr/writings/cathedral-bazaar/afterword/, (accessed April 10, 2019).

Software had already begun a process of transposition to other areas: education materials, legal defenses, and publications.

Open Content License (OPL)

In 1998, David Wiley, a doctoral candidate at Brigham Young University, sought to apply the Free Software approach to written materials. Wiley had developed teaching materials that he desired to be freely circulated under protections like those of Stallman's GNU Public License. To achieve this, the graduate student engaged Stallman, and they worked together to modify the GNU software license to apply it to written content. The result was the Open Content License (OPL). Wiley's license could be used for any manner of content: text, photos, movies, or music. Use of the license quickly spread for use with academic texts and software documentation. Although Wiley discussed his transposition of Stallman's Free Software license to the OPL with Raymond in 1998, this was nonetheless absent from the author's discussion at the end of *The Cathedral and the Bazaar.*[49]

The transposition of a Free Software license to written materials is unsurprising in that the hacker community conceived their practices as freedom of speech. Naturally, the protections afforded to one form of expression would easily translate to another. The difference was that other forms of expression – music, books, and films – never collectively had a time in the modern era where sharing was the norm, as it had been with software. Most other forms of expression had been covered under copyright in the United States since the nation's inception. Software was a product of twentieth century

[49] Lev Grossman, "New Free License to Cover Content Online," *Time Digital*, September 8, 1998, 1.

electronic computing, and its protection under copyright required a complete reversal of legal precedent.

The Creative Commons

By the 1990s, the concept of the public domain was imperiled. The issues stemmed from the 1976 Copyright Act, which had eliminated the requirement for copyrighted material to be registered by the government. Before this, registered works were granted copyright and unregistered works were considered public domain. The 1976 Act granted copyright immediately upon the production of a work, inadvertently creating a situation where there was no explicit way to place something in the public domain. Under the new rules, all intellectual works were property, owned by an individual. As such there was no longer a clear path to creating something for common use.

The public domain was further constrained with the passing of the Sonny Bono Copyright Term Extension Act in October 1988. The legislation amended the 1976 Copyright Act and significantly delayed the entry of works into the public domain. When the United States was founded, copyrights protected material for 28 years.[50] By the middle of the twentieth century, this period was extended to a maximum period of fifty-six years. The 1976 law expanded this dramatically, attaching the timeline to the lifespan of the creator, protecting the material for the life of the author, and then an additional fifty years following their death. The 1988 legislation extended the timelines yet again, extending the post-mortem protection period to seventy years for individual authorship,

[50] The following is a concise summary of changes to copyright law and is intended to show the pattern of copyright terms being lengthened. Although not detailed here, these terms were increased eleven times in the second half of the twentieth century. See OPENLAW: Eldred v. Ashcroft. Wendy Seltzer, “Openlaw:Eldred vs. Ashcroft,” Harvard Berkman Center, https://cyber.harvard.edu/openlaw/eldredvashcroft/, (accessed July 15, 2020).

and to one hundred and twenty years after creation for corporate materials.[51] The law was championed by entertainment companies such as Disney, which benefited from the extended profitability afforded by the law.[52]

<u>OpenLaw</u>

In 1998, Eric Eldred, the founder of Eldritch Press, a website dedicated to scanning and sharing public domain literature, brought a suit against the U.S. government (represented by Attorney General John Ashcroft) over the Sonny Bono Copyright Term Extension Act. In the case, <u>Eldred v. Ashcroft</u>, organizations, such as small publishers and classical music distributers took issue with law as it offered a material impact to their activities by limiting their ability to disseminate materials from the public domain. ,.[53] In support of the case, Harvard Professor Lawrence Lessig and Duke University Professor James Boyle created OpenLaw, a project designed to apply Free Software's commons-based peer production to legal representation. Hosted by the Berkman Center for Internet and Society at Harvard University, OpenLaw's website provided case details and sought assistance to "develop arguments, draft pleadings, and edit briefs in public, online." The project sought participation on the case, but also hoped to use technology to allow "the

[51] Timothy Lee, "15 Years Ago, Congress Kept Mickey Mouse Out of the Public domain. Will They Do It Again?" *Washington Post*, October 25, 2013, https://www.washingtonpost.com/news/the-switch/wp/2013/10/25/15-years-ago-congress-kept-mickey-mouse-out-of-the-public-domain-will-they-do-it-again/?utm_term=.3cf2d16b1692, (accessed April 10, 2019).

[52] The desire for the law is easy to see for Disney. The company, a vocal advocate for the legislation, had enjoyed extended protection for their characters each time the law was advanced. For example, the character of Mickey Mouse was created in 1926. The copyright on the character was set to expire in 1982, but the Copyright Act of 1976 afforded it additional protection. Under the updated terms, the character would be protected for fifty years following Walt Disney's death in 1966. This meant that the character would pass into the public domain in 2016. The 1988 law changed this again, extending the copyright to 2036. The Disney corporation owns many characters, a number of which are older than the organization's figurehead mouse.

[53] "Eldritch Press," Eldritch Press, http://www.ibiblio.org/eldritch/, (accessed April 10, 2019).

public interest to speak as loudly as the interests of corporations."[54] By the time the case reached the Supreme Court, the amicus briefs in opposition to their case underscored this point: the supporters of the law were a veritable who's who of corporate media production: the Motion Picture Association of America (MPAA), Recording Industry Association of America (RIAA), and AOL Time Warner. Eldred's supporting amici were more representative of intellectual interests: fifty-three intellectual law professors, seventeen economists, five constitutional law professors, fifteen library associations, the National Writers Union, and the Free Software Foundation.[55]

Creative Commons

In 2001 while preparing for Eldred v. Ashcroft, OpenLaw's founders, Lawrence Lessig and James Boyle also launched another endeavor: The Creative Commons. According to anthropologist Christopher Kelty, the Creative Commons was a fallback measure in case they lost the Eldred v. Ashcroft case.[56] The non-profit organization was focused on enriching the public domain by adopting the licensing approach pioneered by Stallman and others in the Free Software Movement. Expanding on Wiley's Free Software-based Open Content License, the Creative Commons sought to protect and provide a wide range of content types – from musical composition to film and written works. And just as Stallman had with Free Software, the group saw the transformative potential in the approach. Even if the government won the Eldred v. Ashcroft case, the Creative Commons would have provided tools for people to work around the law,

[54] Wendy Seltzer, "Openlaw," Harvard Berkman Center, https://cyber.harvard.edu/openlaw/, (accessed April 10, 2019).

[55] "Cert Granted: The Supreme Court will hear Eldred v. Ashcroft in its fall term.," Harvard Berkman Center Openlaw, https://cyber.harvard.edu/eldredvreno/legal.html, (accessed April 10, 2019).

[56] And lose the case they did. On January 15, 2003, in a 7-2 decision, the Supreme Court held the copyright law to be constitutional.

creating a "direct route to the transformation of the legal structure of intellectual property." To do this, the organization created a series of variations on the Free Software licenses, offering differing degrees of content ownership, running the gamut between "all rights reserved" to "some rights reserved," and even "no rights reserved."[57] The new licenses provided a clear legal path to the public domain and provided the flexibility to be extended to any form of content ownership.[58]

Free and Open Source Encyclopedias: GNUpedia, Wikipedia

On December 18, 2000, Stallman announced his plans to create GNUpedia, a "free universal encyclopedia and learning resource."[59] The previous year Stallman had introduced an open content license for the GNU project, the GNU Free Documentation License (GFDL).[60] The encyclopedia was to a great extent a progression from this earlier effort. Having created a content license agreeable to Stallman's sociopolitical goals, the encyclopedia offered a way to further spread the ideas.

In his project announcement, Stallman provided justifications for the effort, rationalizations which were informed by corporate distrust. "The World Wide Web has the potential to develop into a universal encyclopedia covering all areas of knowledge," Stallman wrote, "…but corporations are mobilizing now to direct the future down a different track -- one in which they control and restrict access to learning materials." Stallman argued against the corporate control of knowledge. Citing the Free Software

---

[57] Kelty, 260,265.

[58] Lessig and others at the Creative Commons worked with the Debian software developers on some derivative licenses, but due to the wide variability of licenses that the group provides, not all of their protections meet the criteria needed for Free Software or the derivative Debian guidelines based licenses.

[59] Richard Stallman. "Re: Evaluation of Gcompris." Email to vtamara@gnu.org. December 18, 2000.

[60] Richard Stallman. "New Documentation License--Comments Requested." Email to Newsgroup gnu.announce,comp.os.linux.misc,gnu.misc.discuss. September 12, 1999.

Movement which had created a world of software apart from corporate control, he urged for the creation of a universal free encyclopedia, "an alternative to the restricted ones that media corporations will write."[61]

Stallman than described his vision for the encyclopedia effort. It had to be universal and decentralized; the content would be fully accessible, translated into different languages and mirrored throughout the world. Ideally, the authors would have expertise in their given fields and content would grow organically from the contributors. There would be no central control or governance of the content, but peer review and professional endorsements would be encouraged. Most importantly, the encyclopedia needed to provide the same freedoms as Free Software. The website had to permit modified versions of content, including text, pictures, and videos. Direct quotations required attribution, and any subject matter linked to from the site had to have a license that permitted its use. Naturally, any programs involved within the project would need to be Free Software.[62]

Within weeks, the project effort ran into a sizeable road block: the intended domain name, gnupedia.org, had already been taken. The domain name brought visitors to another encyclopedia – Nupedia. Then owner of the domain, Internet entrepreneur Jim Wales, explained his decision to purchase the domain; "I registered gnupedia.com/net/org a long time ago… to try to prevent this kind of confusion from happening."[63] Wales, co-owner of Bomis, an adult web portal, had started a similar sounding Gnupedia project the previous year.

The Nupedia project was started and financed by Jim Wales and his partners at Bomis. The intention was to create a reliable, peer-reviewed, on-line encyclopedia. Wales hired a recent graduate, Dr. Larry Sanger, to lead the effort. Sanger and Wales assembled

---

[61] Stallman. "Re: Evaluation of Gcompris."
[62] Ibid.
[63] Jimmy Wales. "Re: [Bug-gnupedia] gnupedia.org resolves to nupedia." Email to Newsgroup Bug-gnupedia. January 21, 2001.

an advisory board of doctoral level experts in various fields for the project. The Nupedia framework anticipated public contributions to create content: article drafts, and topic proposals were accepted from anyone with familiarity on a given topic. The content they created would then be vetted by experts on the advisory board via a seven-step review process. This process, designed by the advisory board members, was intended to lend credibility to the encyclopedia by providing oversight from bona fide experts.

The Nupedia project expanded rapidly, but the production of content progressed slowly. In less than two years, the project had attracted almost two thousand contributors, or "Nupedians." However, in the same time, the effort had only produced about twenty articles. Sanger determined that the issue was with the complexity of the review process. The efforts were initially handled through exchanges on email lists. The team eventually introduced a basic webpage to facilitate the effort, but it was cumbersome and saw low adoption. Sanger discussed his concerns with the process with his friend, Ben Kovitz. Kovitz, a computer programmer who had been working on Ward Cunningham's WikiWikiWeb, recommended that Sanger consider implementing a wiki, which would allow for the rapid production of public content.[64]

Cunningham had designed the first Wiki website in 1995. The idea behind the project was that it would allow contributors to come together and create a site using basic collaborative software. Named Wiki, after the Hawaiian word for quick, the web scripts were designed to create a low barrier to contribution; any user could add or revise

[64] Larry Singer, "The Early History of Nupedia and Wikipedia: A Memoir," in *Open Sources 2.0: The Continuing Evolution* (Beijing: O'Reilly, 2006), 308-315.

content. Cunningham created the WikiWikiWeb site and made both the website and software free to the public.[65]

Following the meeting with Kovitz, Sanger set up a wiki for the encyclopedists at Nupedia, but the website met resistance from the advisory board since the format was unlike that of a traditional encyclopedia. In reaction to the editorial opposition to the site, Sanger and Wales decided to launch the wiki as an independent project. On January 15, 2001, they revealed the Wikipedia website, with the intention that the site would generate content that might be fed into the Nupedia effort. Sanger, still editor-in-chief at Nupedia, refocused his efforts on promoting the new website.[66]

The emphasis with the new Wikipedia website was to encourage collaboration. Sanger eschewed references to his doctorate or role as editor-in-chief in an effort to make contributors comfortable with submitting content. Likewise, the site had a hands-off management policy that was extremely tolerant of all but the most extreme behavior from the contributors. This approach mirrored Stallman's calls for a fully accessible, decentralized knowledge repository. The extent of governance on the site was a set of precepts to guide the approach: the site should be collaborative, with unedited, unapproved content available for review and update. The subject matter had to focus on being encyclopedic rather than to have free-form contributions or overly-concise snippets.[67] The material also had to retain a neutral tone, addressing topics in wording

[65] Ward Cunningham, "Wiki History," C2.com, last modified December 22, 2014, http://wiki.c2.com/?WikiHistory, (accessed April 13, 2019).

[66] Singer, 315-317.

[67] At the time that the site was introduced, other, similar, online knowledge bases had begun to emerge. For example, the Slashdot related site, Everything^2 had been released in 1998, but the site lacked a formal focus for content. Contributions were welcome on anything and licensed individually. Wikipedia

that was as objective as possible.[68] The site was licensed under the GNU Free Documentation License (GFDL), meeting Stallman's ideological intentions for the site content.[69]

Wikipedia grew rapidly. By July 2001, the site already had six thousand articles, and it began to grow by about two thousand a month. By October, the site hosted over thirteen thousand topics. Part of this rapid growth was due to Google. Each time the search engine surveyed the site, it found an increasing number of topics and links. This brought more traffic to the site, and with it more contributors. Likewise, links to the budding encyclopedia from community websites such as Slashdot brought the site increased visibility, and with it contributions.[70]

The rapid growth of the Wikipedia had a downside, however. The unvetted, public encyclopedia soon eclipsed its peer-reviewed progenitor, Nupedia. The extreme permissiveness inherent to Wikipedia's policies valued lay contributions on the same level as the work done by professional academics. This led to conflict between contributors, but the organization's hands-off approach prevented them from intervening in all but the most egregious situations. According to Sanger, this fostered a culture of anti-elitism that ultimately resulted in the departure of the academic contributors from both Nupedia and its descendant, Wikipedia.[71] What had begun as an effort to create a

---

differed in that it was first an encyclopedia, and its structures were intended to maintain this tone and objective.

[68] Singer, 321-323.

[69] The GNU free Encyclopedia Project, originally Gnupedia, later GNE (GNE's Not an Encyclopedia) eventually ceased their effort in favor of Wikipedia, which met the effort's original objectives. See Singer 323.

[70] Singer, 322-324.

[71] Ibid., 324.

centrally managed, peer-reviewed online compendium morphed into a commons-based, peer-produced encyclopedia.

Other Transpositions

The transpositions of the Free Software ideology and commons-based peer production model were not limited to the Open Content License, OCL, OpenLaw, The Creative Commons, or Wikipedia. The connections between these efforts constituted just one of many threads where the schema and/or ideology informed the creation of new efforts. Other notable examples included the SETI@Home project and WordPress.

SETI@Home

The SETI@Home project performed astronomical spectroscopy analysis in search of extra-terrestrial signals. The project began in 1995, when Berkeley researcher David Gedye proposed the creation of a peer-based virtual supercomputer composed of distributed Internet-connected workstations.[72] He developed the SETI@Home software platform, which he released under the GPL Free Software license in May 1999.[73] It is not apparent if the SETI@Home project embraced the ideology of Free Software, but it utilized Stallman's license, and it demonstrated a creative transposition of the schema from collaborative software development to distributed computing.[74]

WordPress

[72] "About Seti@home," SETI@home, https://setiathome.berkeley.edu/sah_about.php, (accessed April 13, 2019).

[73] "Porting SETI@home," SETI@home, https://setiathome.berkeley.edu/sah_porting.php, (accessed April 13, 2019).

[74] The software was superseded by the LGPL-licensed Berkeley Open Infrastructure for Network Computing (BOINC) platform in 2005.

WordPress was an effort that sought to "democratize publishing."[75] The effort came out of a Free Software weblog project named "B2." The original software had been maintained by programmer Michel Valdrighi, but by the start of 2003, his engagement on the project had ceased. This resulted in the project undergoing a "fork," where the effort bifurcated, and development of the source code proceeded down two different paths.[76] The new developers, Matt Mullenweg and Mike Little, released their version of the software as WordPress in May 2003. In addition to releasing their software under the GPL license, they also launched wordpress.org, a LAMP-based webservice for all to freely use.[77] The site was based on their software, which allowed non-technical individuals to easily create a publishing website.

The WordPress project is very clear in its ideological alignment with Stallman and Free Software. On their website, they detail their goal of democratizing publishing. They explain that the software they have created is published under Stallman's GPL license, and then they connect their own "WordPress Bill of Rights" to the Free Software Foundation's four freedoms: to use a program for any purpose, to study and alter the program as desired, and to distribute modified versions to others so that they might enjoy the same freedoms.[78] Their site even goes as far to discuss the Free/Open Source

[75] "Democratize Publishing," Wordpress, https://wordpress.org/about/, (accessed April 13, 2019).

[76] This term comes form UNIX where the command 'fork' can be used to spawn additional processes from a running parent process. Project forks are strongly resisted within Free Software as it results in a duplication of effort. Nonetheless, there have been a number of famous project forks. For example, the BSD variant of UNIX would be considered a fork. The Microsoft interoperability project SAMBA also forked, with a secondary project, SAMBA-TNG focusing on full replication of services rather than emulation.

[77] Siobahn McKeon, "Wordpress," WordPress History, Chapter 3, 2-6, https://wordpress.org/about/history/chapter3.pdf, (accessed April 13, 2019).

[78] "Democratize Publishing," Wordpress, https://wordpress.org/about/, (accessed April 13, 2019).

Software community and, based on its example, expounds on the etiquette that is expected within the WordPress community.[79]

Part 2: Into the Streets

The second part of this chapter reviews how the U.S. government's inability to differentiate between hacking and cracking resulted in a heavy-handed approach to policing early on-line communities and prompted the creation of the Electronic Frontier Foundation. This section will also demonstrate how new revisions to U.S. copyright law criminalized certain types of programming, turning hackers into crackers, and how the subsequent enforcement of the law resulted in the arrest of programmers and censorship of researchers. These actions brought the Free Software community to the streets in protest.

From Hackers to Crackers

In November 1984, counter-culture publisher Stewart Brand organized an invitation-only Hacker's Conference in Marin County, California. Inspired by Stephen Levy's book, *Hackers*, the event was designed to be a meeting of the individuals detailed within the book. During the meeting at Fort Cronkhite, Brand endeavored to have the group explore the unspoken Hacker ethic that Levy had postulated in his book. Seeking to determine if the behaviors were a "precursor to a larger culture," Brand asked the attendees to articulate their perspective on the alleged ethic.[80] A number of the participants rejected Levy's portrayal of their culture as tied to a singular ethic. Hacker

[79] Despite the explicit re-iteration of the Free Software Foundation's ideals, Word Press refers to the community by the more well-known "Open Source Appellation."

[80] Turner, 135-136.

Steve Witham, a programmer working on Xanadu, argued that it is less of an ethic and more like an instinct.[81] Apple co-founder Steve Wozniak characterized the practices as something closer to a drive like the intellectual curiosity that motivates children. For the hackers, Wozniak reasoned, the "motivation is what's different…the challenge of solving the puzzle was the only reward."[82] Stallman, speaking to concerns that Levy's book portrayed hackers as against intellectual property, explained how the hacker perspective was more accurately against obstacles. Hackers tended to "not put up with someone who wants to deliberately obstruct them," and when they encounter prohibitions, they view the impediment as bureaucratic challenge. MIT Hacker and Lisp Machines Incorporated founder Richard Greenblatt echoed Stallman's perspective, "I think it's very fundamental that source codes be made available."[83] Wozniak agreed, citing the educational benefits of looking "at code, like operating system, listings, and the like, to learn how it was done before them. Source should be made available." Stewart Brand distilled the Hacker perspective down to a paradox, holding that "Information wants to be free," and yet, "information wants to be expensive because it is so valuable."[84]

The Hacker Convention was created to explore Levy's 1984 text, *Hackers*, and his articulation of the "hacker ethic" – the principles later enshrined in Stallman's' GNU project and the GNU Public License. The meeting had nothing at all to do with breaking into computer systems – and yet several of the meeting's participants were themselves

[81] Xanadu was a project intended to create a worldwide database and text sharing system, similar in concept to what later emerged as the World Wide Web.
[82] Stewart Brand, ed., "Discussions for the Hacker's Conference, 1984," *Whole Earth Review*, May 1985. 46.
[83] Ibid. 48.
[84] Ibid. 49.

accustomed to cracking – a few of the West Coast computer hobbyists present at the meeting were known phone system crackers, or *phreakers*.[85]

Phreaking

Since the 1950s, experimentation with the phone networks had been a frequent practice among some electronic hobbyists. In a 1971 *Esquire* magazine article, journalist Ron Rosenbaum presented a colorful account of the practice of cracking the phone system, and in the process revealed to the world a phreaker named John Draper, who went by the nom de plume "Cap'n Crunch." Draper, who would later be a Hacker Conference attendee, had discovered that the toy bosun whistle included as a prize in boxes of Cap'n Crunch cereal emitted a 2600 hertz tone, which was the same frequency that AT&T's signaling system used to open a line for routing a new call.[86] The result was that for the cost of a box of sugar-laden cereal, long distance calls could be made throughout the country. The article also detailed the creation of a blue box, a simple electronic device designed to generate the tones required to control the phone system. After reading the *Esquire* article Wozniak, and his future business partner Steve Jobs, began to design and sell digital blue boxes.[87] By the time the duo began to produce and sell Apple computers, the technique no longer worked consistently.[88]

---

[85] A phreaker is someone who cracks telephony systems.

[86] Ron Rosenbaum, "Secrets of the Little Blue Box," *Esquire*, October 1971, 120-122.

[87] Bill Chappell, "Steve Jobs Dishes On the Tech Business in 'lost interview' from 1995," National Public Radio, November 16, 2011, https://www.npr.org/sections/alltechconsidered/2011/11/16/142373973/steve-jobs-dishes-on-the-tech-business-in-lost-interview-from-1995, (accessed April 13, 2019).

[88] The adoption of the new type of signaling was progressive. Depending on time and location, blue boxes would still have worked over the following decades as the regional operating companies progressively switched over to the new system. This is also not intended to suggest that all phreaking techniques ended with the implementation of signaling system seven. Techniques such as beige boxing (essentially creating a lineman's handset and using it to connect into a local telephony bridgehead) still

Cracking

Security on computer systems evolved almost as an afterthought. The invention of the first computer password occurred in the mid-1960s and is credited to MIT researcher Fernando Corbató.[89] The MIT researchers were developing a time-sharing operating system which would be used by multiple users simultaneously. This is not to say that this password was secure; it was stored on the computer in clear text format – anyone who knew where to look could read what the users' passwords were. This issue was resolved by the UNIX team at Bell Labs when Researcher Robert Morris wrote the Crypt function, which encrypted the passwords stored on the UNIX server.[90] Morris and UNIX co-creator Dennis Ritchie presented the encryption functionality along with a thorough analysis of potential cracking techniques in 1979; however, the functionality was not widely available until the release of System III UNIX in 1981.[91] Even then, the introduction of the security levels were frequently met with resistance. For example, when the first password system was installed in the MIT AI lab, many hackers resisted using them. Stallman was particularly vocal, citing an ethical objection to the security, which he felt infringed on the lab's culture of "intellectual openness and trust."[92] The

---

work today, but there is little incentive to perform calls in this way since long-distance communications can now be inexpensively done via the Internet.

[89] Robert McMillan, "The World's First Computer Password? It Was Useless Too," *Wired Magazine*, January 27, 2012, https://www.wired.com/2012/01/computer-password/, (accessed April 13, 2019).

[90] Even with this added security, password security was still problematic. Early Internet protocols such as Telnet and FTP sent the username and password information over the network in clear text, which meant that the credentials could theoretically be discerned by intercepting and reviewing the network packets.

[91] Robert Morris and Dennis Richie, "Password Security: A Case History," *Communications of the ACM*, November, 1979.

[92] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software* (Sebastopol, California: O'Reilly, 2002), 53-54.

programmer and just under a quarter of the lab's other hackers elected to use a blank password, which defeated the purpose of the system.[93]

With utilities such as Crypt to conceal a system's authentication information, the encryption now would have to be "broken" or "cracked." However, this did not necessarily mean a system was secure. Any computer with a modem or network connection was a potential target. A malicious user with a computer could use a program called a *war dialer* to have their modem sequentially call all the phone numbers in a given area. If a modem answered, they could attempt to *brute force* the remote computer, repeatedly trying passwords until the correct one was found. Frequently, hacking was not even this technical – if a potential cracker knew the organization that owned a particular dial-up line, they could call the information and perform *social engineering*, an approach where they convince unsuspecting employees to share their computer credentials by pretending to be affiliated with the company.

By the early 1980s, computer cracking was becoming more common. Within popular culture there arose an interest in these practices, along with a tendency to conflate *hackers* – individuals focused on producing clever technical solutions for problems -- with *crackers,* individuals who broke into computer systems. This can be seen in the 1983 film, *WarGames*, where a teenage character uses the cracking practices described above to break into remote computer systems. The film's protagonist, David Lightman, was attempting to play new video games when he accidentally triggered a

[93] This resistance to centralized authority predated the AI Lab's crisis over the commercialization of the LISP machine. It also provides an easy answer as to how Symbolics was able to access Stallman's account in order to install software that spied on his activities.

military computer into running a real-time simulation of a global thermonuclear war.[94] In the movie, the white suburban teenager and his "hacker" activities inadvertently presented an existential threat to humanity.

Aside from the scope of the plot, the techniques displayed in the film were technically accurate, and the picture portrayed the young "hacker" in a sympathetic light. Played by Matthew Broderick, Lightman was depicted as a misunderstood, bored genius, who inadvertently set a potential apocalypse in motion. By the end of the film, Lightman has helped resolve the crisis and is seemingly free from any consequences for his part in the affair.[95]

Following the film's release, the Milwaukee, Wisconsin-based "414" group was arrested for cracking computer systems at locations including Los Alamos National Laboratory, Security Pacific National Bank in Los Angeles, and New York's Memorial Sloan-Kettering Cancer Center.[96] The following year saw the creation of similar teenager hacking groups with comic-like names such as the "Legion of Doom" and "Masters of Deception." In 1984, New York City computer enthusiast Eric Corley began writing articles on hacking under the pseudonym Emmanuel Goldstein.[97] His publication, *2600: The Hacker Quarterly,* featured articles on cracking and phreaking.[98] The same year, Canadian author William Gibson published *Neuromancer*, a science fiction novel whose

---

[94] *WarGames*, directed by John Badham. (United Artists,1983) DVD. (MGM/UA Entertainment, 2008).
[95] Ibid.
[96] William Marbach et al., "Beware: Hackers at Play," *Newsweek*, September 5, 1983, 42-43.
[97] Universal City v. Reimerdes, 2000 U.S. 2nd Circuit Court of Appeals.
[98] The magazine took its title from the 2600 hertz tone produced by the Cap'n Crunch whistle. Corley's nom de plume is equally referential, connecting his activities with the anti-government antagonist in George Orwell's novel *1984*.

protagonist, Case, was a "hacker" navigating a bleak dystopian "Cyberpunk" future that was focused on a large-scale virtual reality called the Matrix.[99]

The electronic reality portrayed in *Neuromancer* was a fictional re-imaging of the robust on-line community that had emerged by the mid-1980s. The Internet had yet to become public, but Bulletin Board Systems (BBS) supported community forums in news groups and facilitated file sharing. Public adoption of computers and online communities was just beginning, but early hacking-focused, popular culture efforts conflated hacking and cracking and, in the process, fueled public interest in the illicit activity. At the same time, the U.S. government struggled to fully understand the burgeoning online communities and how and when to intervene on computer-based activities. In 1984, President Ronald Reagan signed the Comprehensive Crime Control Act. The law was updated in 1986 with the Computer Fraud and Abuse Act. Both laws sought to provide a legal response to computer-based crimes. In 1988, DARPA formed the Computer Emergency Response Team (CERT) Coordination center. Located at Pittsburgh's Carnegie Mellon University CERT endeavored to provide centralized management for security issues on the DARPA's networks.[100]

In November of the same year, the government's new CERT center was put to the test when Cornell graduate student Robert Tappan Morris released a computer program that inadvertently took down most of the computers on the burgeoning Internet. Morris, whose father was the aforementioned Bell Laboratory cryptographer, had discovered several vulnerabilities in the UNIX operating system. The student wrote software that

---

[99] William Gibson and Neil Gaiman, *Neuromancer*, Penguin Galaxy (New York, New York: Penguin Books, 2016), 2-7.

[100] Timeline: The U.S. Government and Cybersecurity, *Washington Post*, May 16, 2003, http://www.washingtonpost.com/wp-dyn/articles/A50606-2002Jun26.html, (accessed April 13, 2019).

would traverse the network and locate computers with the security hole. The program was not designed to harm any of the computers. Instead, it was intended to replicate itself and spread to new computers. Morris made a miscalculation in his replication code. Upon its release on November 2, 1988, Robert Tappan Morris' software infected and overwhelmed over 600,000 computers. The software was the Internet's first malware, a "worm," which was a program without a destructive payload that replicates like a virus. The graduate student was charged under the Computer Fraud and Abuse Act, sentenced to three years' probation, and ordered to pay a fine of $10,000 and perform 400 hours of community service.[101]

Despite the legislation aimed at computer crimes and the CERT center, the U.S. government seemed unsure of the new digital world and its role in it. This can be seen in Clifford Stoll's non-fiction text, *The Cuckoo's Egg,* in which the author provided his account of detecting an intrusion at his workplace, California's Lawrence Berkeley National Laboratory (LBNL). Stoll's narrative detailed his own efforts to trace the intruder and document his activities. In the process, the author revealed how federal law enforcement was unprepared to handle such investigation. The break-in was eventually traced to Markus Hess, a German "hacker" who penetrated U.S. facilities in the service of the KGB. However, after discovering the intruder, Stoll attempted to engage U.S. government agencies, only to find them unclear of their jurisdiction and unaware of the technologies involved: After detecting the intruder, Stoll contacted the FBI. The agent he spoke with was disinterested and suggested the agency would need a search warrant to

[101] Carolyn Marsan, "Where Is Robert Morris Now?" *NetworkWorld*, October 30, 2008, https://www.networkworld.com/article/2268914/where-is-robert-morris-now-.html, (accessed May 8, 2019).

get involved. Since the LBNL was funded by the Department of Energy, the researcher next contacted Rick Carr, the department's manager of computer security. Carr suggested that Stoll call the National Computer Security Center at the NSA. After describing the issues to operatives at the NSA, the researcher was told the NSA could not get involved, as their efforts could be interpreted as domestic monitoring, which was outside of their jurisdiction. Stoll then contacted the CIA, who advised him to not block the intruder, but to continue monitoring the intrusive activities with the assumption that "the FBI will eventually wake up."[102]

The Electronic Frontier Foundation

As Clifford Stoll's book demonstrated, there were legitimate concerns over computer intrusions, but the U.S. government struggled to understand and address the threat. In the early months of 1990, the FBI did "wake up," and the government agency began to visit alleged "hackers" and seize their computers and equipment. These seizures culminated in a highly publicized raid on May 8, 1990. Dubbed "Operation Sun Devil," the anti-hacker campaign featured 150 federal agents working with hundreds of state and local police in 14 cities throughout the nation. Billed as the culmination of a two-year investigation into credit card and computer fraud, the highly visible offensive against alleged computer crime netted few results. By 1992, only three individuals were charged

[102] Clifford Stoll, *The Cuckoo's Egg: Tracking a Spy through the Maze of Computer Espionage* (New York, New York: Doubleday, 1989), 106-108, 82.

– two of whom were immediately sentenced to probation, but the government kept the 40 computers and 23,000 discs that it had seized from other uncharged citizens.[103]

One of the hacking investigations leading up to the 1990 Operation was a search for a hacker group called the NuPrometheus League, a group that had claimed responsibility for stealing Apple Computers' QuickDraw graphics library source code. In pursuit of the cyber criminals, FBI Special Agent Richard Baxter Jr. visited John Perry Barlow, former lyricist for the Grateful Dead, and a regular visitor to the on-line community, the Whole Earth 'Lectronic Link (WELL.) Founded by Whole Earth Catalogue publisher Stewart Brand, the WELL was a computer network frequented by the Bay Area's computer counterculture. In the process of questioning Barlow, the agent revealed how his investigation brought him to the lyricist. The agent had determined that the Hacker's Conference, by now an annual convention, was actually "an underground organization of computer outlaws with likely connections to, and almost certainly sympathy with, the NuPrometheus League."[104] Since Barlow had previously been known to attend the conference, it was assumed he was aligned with the group. Furthermore, the agent incorrectly determined that John Draper (the aforementioned Cap'n Crunch) was the CEO and President of Autodesk, Inc. Autodesk, creator of drafting software such as AutoCad, was a concern to the agent because the organization, "had many top-secret contracts with the government to supply Star Wars graphics imaging and 'hyperspace' technology." The agency believed that Draper may have had

[103] Dan Charles, "'Innocent' Hackers Want Their Computers Back," *New Scientist*, May 9, 1992. https://www.newscientist.com/article/mg13418201-400-innocent-hackers-want-their-computers-back-/, (Accessed July 15,2020.)
[104] John Perry Barlow, "Crime and Puzzlement," Electronic Frontier Foundation, June 8, 1990, https://www.eff.org/pages/crime-and-puzzlement, (accessed May 8, 2019).

contacts within the Soviet Union. According to his account of the visit, Barlow attempted to explain the flaws in the conclusions that had been reached. For example, Barlow pointed out that Apple founder Steve Wozniak had been attending the Hacker's Convention since its inception. Agent Baxter was incredulous at Barlow's claim of Wozniak's participation, since his contact at Apple characterized the group as Anti-Apple, "due to the Hacker belief that software should be free combined with festering resentment of Apple's commercial success." The agent then began to question Barlow about other WELL users, such as Mitch Kapor, the retired creator of the Lotus 1-2-3 spreadsheet. Concerned with intent and inexactitude of the misinformed agent, Barlow posted an account of his visit with Agent Baxter on the WELL immediately after his departure.

Several days after the meeting John Perry Barlow met with Mitch Kapor. Having read Barlow's account of the conversation, Kapor had grown concerned with "the inappropriateness of leaving our civil liberties to be defined by the technologically benighted." Following their discussion, they saw the need for civil liberty protections in the digital sphere. Working with John Gilmore, founder of the Free Software support company Cygnus, Barlow and Kapor founded the Electronic Frontier Foundation (EFF.)[105] The organization was intended to "raise and disburse funds for education, lobbying, and litigation in the areas relating to digital speech and the extension of the

[105] The group was originally intended to be named the "Computer Liberty Foundation," but was later renamed that fall during a dinner meeting with individuals that the trio hoped to enlist to serve on the board. See Turner, 172.

Constitution into Cyberspace."[106] Upon learning of the new group, Wozniak offered to help fund the groups education and lobbying efforts.

In *From Counterculture to Cyberculture*, Stanford University Professor Fred Turner recounted the lobbying efforts of the Electronic Frontier Foundation during the 1990s. Through John Perry Barlow and Stewart Brand, Turner connected the EFF efforts in lobbying Congress with the ideals of the 1960s counterculture. The author situated the efforts as supportive of a new, global economy based on networked production, contract employment, outsourcing and de-regulation. According to Turner, their lobbying brought them in line with the New Right movement led by Newt Gingrich, who interpreted their efforts as support for a free-market system.[107] Turner's connections relied chiefly on Esther Dyson, one of the later EFF directors, who co-authored the August 1994 futurist treatise, "Cyberspace and the American Dream: A Magna Carta for the Knowledge Age."[108] Newt Gingrich joined Dyson for a *Wired* magazine interview in 1995, where the two discussed how the global network could fundamentally change production and the nature of employment.[109] Even with the political connections demonstrated by Turner, the reality is that the Electronic Frontier Foundation's lobbying efforts in the early 1990s were fruitless. Under the guidance of House Majority leader Newt Gingrich, the 105th Congress of the United States passed legislation that limited the rights of consumers in the digital space and criminalized certain types of programming.

---

[106] Barlow, "Crime and Puzzlement."

[107] Turner, 173-175,228.

[108] Esther Dyson et al., "Cyberspace and the American Dream: A Magna Carta for the Knowledge Age," The Progress and Freedom Foundation, accessed May 8, 2019, http://www.pff.org/issues-pubs/futureinsights/fi1.2magnacarta.html, (accessed May 8, 2019).

[109] Turner, 232.

The Digital Millennium Copyright Act of 1998 (DMCA)

On October 28, 1998, President Clinton signed the Digital Millennium Copyright Act (DMCA) into law. The legislation was intended to update United States copyright law to make it compatible with two international intellectual property agreements signed in 1996, the World Intellectual Property Organization (WIPO) Copyright Treaty and the WIPO Performances and Phonograms Treaty. The law was split into five separate titles. The first title implemented the WIPO treaties. Title II provided protections for online service providers in the case of copyright infringement. Titles III, IV and V provided various exceptions to the law, such as for distance education providers, webcasting, and making backup copies of computer programs.[110]

Title I of the DMCA updated the nation's copyright laws (Title 17 of the U.S. Code) by adding a new chapter 12. In this section, the law met the obligations of the WIPO Copyright Treaty by providing legal protections against the circumvention of any technological tools used by copyright owners to protect their creations. This section divided these measures into two distinct types: The first type were barriers intended to prevent unauthorized access to the work. The second were impediments against the duplication of the copyrighted material. The fair use doctrine within existing copyright law permitted owners to reproduce works for the purposes of backup. However, the DMCA changed this: regardless of the circumstance, the new law did not permit "the act of circumventing a technological measure in order to gain access."[111] In other words,

[110] *The Digital Millennium Copyright Act of 1998 U.S. Copyright Office Summary* (Washington, DC: U.S. Copyright Office, December 1998). 1.
[111] Ibid., 3-4.

under the DMCA, a consumer who owned a copyrighted work did not necessarily have the right to use it or make backups copies.

The DMCA's new chapter 12 did grant some exemptions for the new prohibitions. In section 1201 D of the law, archives, educational institutions, and libraries were permitted to circumvent the technological measures in a limited set of circumstances. Reverse engineering to achieve interoperability for computer programs was allowed in section 1201 F. Encryption research was permitted in section 1201 G, allowing researchers to circumvent encryption to "identify flaws and vulnerabilities of encryption technologies."[112]

Title II of the DMCA also added a new section to the Copyright Act. The new rules limited liability on entities that qualified as a "service provider," an organization that provided transitory communications or the storage (or caching) of user information on systems/ networks. Internet service, webhosting, and search engine companies were all protected under the new law. The protected organizations needed to implement a policy whereupon they would cooperate with the effort of copyright owners to identify and protect their works and, if needed, terminate the accounts of repeat copyright offenders. Under the new section of the copyright law, the DMCA created procedures for enforcement. In Section 512 (c)(3), the law described the notice and takedown process, wherein a copyright owner submitted a notice of ownership of specific content to the service provider's designated agent (DMCA Takedown Notice). The service provider was then obligated to promptly remove or block access to the copyrighted material. If

[112] Ibid., 6.

they followed the process, the service provider would be exempt from any monetary liability relative to the specific copyright infringement claim.[113]

The 1998 Digital Millennium Copyright Act was intended to update United States copyright law to make it compatible with international intellectual property agreements. In practice, the law radically changed conceptions of consumer ownership. For example, under traditional copyright laws, once a consumer purchased a VHS tape, the individual owned a copy of the movie and they could do what they liked with it short of illegally disseminating the content. The DMCA changed this, restricting the consumer's personal use of the media. If a consumer purchased a DVD, they were not afforded complete control over the disc they purchased. Title I of the DMCA limited the mechanisms by which the consumer could access the movie on the disk, allowing manufacturers to specify the circumstances in which the media could be used. They could restrict the DVD to play only on certain types of media players or to restrict the playback of the DVD based on geographical regions. No longer did a consumer truly own a DVD; they simply had purchased the right to watch the movie in extremely specific vendor-approved scenarios. And this was not limited to just digital media. Anyone making any purchase with any digital component fell under the law, from students listening to a CD they purchased to consumers attempting to repair a modern car. This presented a radical erosion to the concept of private property; just as the Commission on New Technological Uses of Copyrighted Works' changes had done with software licensing, the DMCA made it so that consumers no longer fully owned anything. Now, as a consumer, you were a licensee and you had to use your purchase according to the license terms, which could

[113] Ibid., 8-12.

spell out anything from the region where the material could be used to from whom you needed to purchase your service or consumables such as printer toner.

Title II of the Digital Millennium Copyright Act offered a more chilling potential. The legislation was designed to protect service providers in the new digitally focused world. However, the DMCA takedown process had the potential to be used as a tool of censorship against criticism. An organization that owned a copyrighted product, such as a software program, could use the law to have negative discussions of the product taken down under the guise of copyright infringement. Since the takedown process existed exclusively between the copyright holder and the service provider, there was no recourse for the individual being censored by the takedown process. At the same time, if a service provider failed to follow the process, they would be exposing themselves to liability in any potential case. As a result, the service provider was disincentivized to intervene at all on the behalf of the censored individual. With the new law, copyright instantly trumped the freedom of speech in the digital sphere.

As mentioned earlier in the chapter, in *Two Bits: The Cultural Significance of Free Software*, anthropologist Christopher Kelty dismissed Free Software's ideologues as fanning "kindling flames of worry over intellectual-property expansion."[114] However, the law, and how it came to be used by corporations in the early 2000s represented an existential threat to the Free Software Movement and its focus on civil liberties in the digital world. As the law was applied against individuals and organizations, the Free Software community became increasingly vocal in their resistance to the DMCA. Their

[114] Kelty, 99.

defiance to the legislation demonstrated how they conceived software as a form of individual expression, and their support for those targeted by the law brought the programmers out from behind their computer terminals and into the streets in protest.

DVDCSS

In the mid-1990s the film industry developed a process to release movies in digital form. They established the Digital Video Disc (DVD) format and they developed a 40-bit encryption standard, the Content Scramble System (CSS) to protect the DVD's contents. The material on the DVD was not accessible unless the disk was accessed via a player that had a set of digital "keys" that were required to de-crypt the media.[115] The studios formed the non-profit DVD Copy Control Association to manage the sale of CSS Licenses to manufactures approved to produce DVD players. The centralized registration process and encryption scheme allowed the industry to classify the nations of the world into different regions, with licensees in each region only allowed to play DVDs created for that part of the world.[116] For example, DVDs created for Africa (Region 5) were not playable in North America (Region 1). This afforded the film studios the ability to regionally control the pricing and distribution of their films.[117] By 1996, the CSS software was ready, and the DVD Copy Control Association was issuing keys to manufacturers. The following year, the film studios began to release movies on the new platform.[118]

---

[115] Universal City v. Reimerdes, 2000 U.S. 2nd Circuit Court of Appeals.

[116] "Content Scramble System (CSS)," DVD Copy Control Association, accessed April 30, 2019, http://www.dvdcca.org/css.aspx.

[117] This also gave the studios the ability to have different release dates in different markets. On the technical side, the regions also helped accommodate historical incompatibilities between regional broadcasting systems (NTSC/PAL/SECAM, etc.)

[118] Universal City v. Reimerdes, 2000 U.S. 2nd Circuit Court of Appeals.

The DVD format became popular following its 1997 release. For the Free Software community, the discs were problematic. Where there were commercial DVD player programs for closed source platforms such as Windows, no such programs were available for Free Software operating systems such as Linux or BSD. Naturally, programmers within the community solved the problem by writing their own.

Collaborating over the Internet, three Free Software programmers reverse engineered a Microsoft Windows based player. After determining the commercial player's decryption keys, the trio created a program capable of reversing the DVD's CSS encryption, DeCSS. The programmer who wrote the GUI interface for the software, Norwegian teenager John Lech Johansen, released a binary copy of the program on his website in September 1999. On September 23, 1999, Johansen posted to the Linux video software development (livid-dev) mailing list, announcing the software. Two weeks later the programmer accidentally posted the source code to his website. Upon review of the posting, the livid-dev programmers took issue with Johansen's software, as some believed it was likely code purloined from a corporation. In the debate about the software, the group soon realized that a program like DeCSS was not even required. On October 27, 1999 programmer Frank Stevenson announced that he had found flaws in the CSS algorithm, and he posted source code that decrypted CSS without needing a key. Within a week, various websites such as Slashdot, began to report that the CSS encryption had been broken. As news of the circumvention became known, movie industry groups like the Motion Picture Association of America began to issue DMCA take down notices and send cease and desist letters to websites that provided any code that performed CSS decryption.[119]

In November 1999, *2600* magazine publisher Eric Corley composed an article about DeCSS which he shared on his website, *2600*.org. His article detailed DeCSS and how CSS was cracked. He also covered the movie industry's desperation to shut down web sites posting the decryption software. At the end of the online commentary, Corley posted copies of DeCSS, providing both binaries and source code. According to the author, writing about the program without including the source code would have been "analogous to printing a story about a picture and not printing the picture."[120]

By January 2000, the response to the DeCSS publication had taken a serious turn. In Norway, the police arrested and questioned John Lech Johansen. In the United States, the movie studios filed suit against Eric Corley and *2600*. Invoking Title, I of the DMCA, the movie studio sought to stop the publication of Corley's article with the DeCSS code. On January 20, 2000, an injunction was granted by Southern District of New York, and *2600* was forced to remove the DeCSS source code from their website. Corley complied with the ruling, but in an act of what he termed "electronic civil disobedience," the publisher provided links to other web sites that were hosting the DeCSS source code. Corley encouraged other webmasters to post the source code as well.

Within weeks, Carnegie Mellon University computer science professor David S. Touretzky began collecting a gallery showing how the DeCSS code was becoming widely disseminated in differing forms of expression. He shared academic mathematical

---

[119] "Openlaw: Informal DeCSS History Timeline," Harvard Berkman Center Openlaw, accessed April 30, 2019, https://cyber.harvard.edu/openlaw/DVD/research/chronology.html.

[120] Universal City v. Reimerdes, 2000 U.S. 2nd Circuit Court of Appeals.

proofs of the decryption process, pictures of the source code, movies featuring the source code, a dramatic reading of the source, tee shirts and ties bearing the source code, and even the program set as haiku. According to Touretzky, his collection was intended to challenge the *2600* ruling where the judge determined that "code that can be directly compiled and executed may be suppressed under the DMCA..., but a textual description of the same algorithm may not be suppressed." [121] Professor Touretzky's gallery demonstrated the absurdity of the ruling and posited, "Where exactly should the line be drawn?"

The Protests Begin

Professor Touretzky's gallery reified the basic perception held by the Free Software community: source code was free speech. In the case against *2600*, the DMCA was used to suppress the publication of code, an intellectual product. The injunction against the publisher demonstrated that freedom of expression was second to copyright law under the DMCA. If corporations could censor the writing and sharing of code using the law, what else could they restrict? What happened if other Free Software efforts ran afoul of corporations? Based on the *2600* injunction, corporations could now launch criminal suits against individuals whose work conflicted with corporate interests.

Coordinating through community structures such as Linux User Groups (LUG) and Slashdot, Free Software supporters began to protest the new law and its perceived overreach.[122] At 9:00 am on Tuesday, March 28, 2000, a group of East Coast Free

[121] David Touretzky, "Gallery of CSS Descramblers," Carnegie Mellon University, March 13, 2000, http://www.cs.cmu.edu/~dst/DeCSS/Gallery/.

[122] It is interesting to note that although Christopher Kelty's account attributes the concerns over the copyright law solely to the Free Software ideologues, Open Source leader Eric Raymond also participated in these protest efforts.

Software supporters assembled and began a march on the U.S. Capitol.[123] The LUG structures and evangelistic nature of the Free Software community had been designed for distributed messaging about the movement. Now, instead of simply spreading the good news of Free Software, the structures were used to resist the new copyright law.

On May 2, 2000, the LUGS coordinated another demonstration in Washington, D.C. Instead of protesting the Capital as they had previously, the group marched outside the Library of Congress where the U.S. Copyright Office was considering testimony before implementing further clauses of the Digital Millennium Copyright Act.[124] Although the rally was relatively small, the two dozen picketers carrying anti-DMCA signs and photos of Johansen attracted more media attention than they had previously. The second march – and the protester's concerns – were covered in the *Wall Street Journal*, the *Nation*, and the *Register*.[125]

Two weeks later, a third demonstration was held, this time on the West Coast at Stanford University. At the University's law school, the U.S. Copyright Office was continuing the process of considering testimony before implementing the final clauses of the new copyright law. In addition to the California user groups, the May 18, 2000 protest at Stanford was also supported by *2600* and the Electronic Frontier Foundation. The latter held a press conference to voice opposition to the DMCA and its anti-circumvention provisions, which they believed to be incompatible with fair use of legally obtained media. Since the copying and distribution of copyrighted material was illegal before the

---

[123] Jim Gleason. "Linux Users to Protest New Copyright Law in Washington, D.C. " Email to nylug-talk@nylug.org Mailing List. March 21, 2000.

[124] Michael Smith. "DC DMCA Protest." Email to ruben@sruben.dental.nyu.edu,nylug-talk@nylug.org Mailing List. May 1, 2000

[125] Jim Gleason. "press coverage of the DMCA protest." Email to nylug-talk@nylug.org Mailing List. May 4, 2000.

passage of the Digital Millennium Copyright Act, the protesters argued that the new restrictions were in place to control the legitimate uses of copyrighted works.[126]

On July 17, 2000, the case brought by the motion picture studios against *2600* began in the Southern District of New York. As with the Washington, D.C. demonstrations, the regional user groups coordinated a protest in support of Corley and *2600*. As with the Stanford protest, the Electronic Frontier Foundation was present, this time to file a motion against Judge Lewis Kaplan who had previously advised Time Warner on DVD related issues. Stallman addressed the questions posited by the press -- *CNN, MSNBC, Reuters, Salon*, and *Wired* magazine. Even Brooklyn congressman Gerald Nadler engaged the protestors, concerned about how the new law could be stopping Linux users from watching the DVDs they purchased.[127] The case, which had prompted the computer users to begin demonstrations, was now helping bring widespread attention to their cause.

The Secure Digital Music Initiative Challenge

While the Motion Picture Association of America (MPAA) and a bevy of movie studios used the DCMA to pursue *2600* magazine in New York, the Recording Industry Association of America (RIAA) and the Secure Digital Music Initiative (SDMI) began to invoke the new copyright law to stop Princeton University professor Edward Felten from publishing academic research. By the late 1990s the recording industry had become concerned with the growing distribution of music in digital form. The small size of

---

[126] "Digital Millennium Copyright Act Protest at Stanford University," LWN.net, last modified May 12, 2000, http://lwn.net/2000/features/DMCA/announcement.html, (accessed April 30, 2019).

[127] Jim Gleason, "July 17, 2000," New York linux Users Group, http://www.nylug.org/7_17_00_page.html, (accessed April 30, 2019).

individual songs and the increasing availability of Internet connections gave rise to peer-to-peer sharing services such as Napster. Here, individuals could make digital copies of tapes, records, or CD's and easily share them on the global network. To address the problem recording industry groups such as the RIAA and the American Society of Composers, Authors and Publishers (ASCAP) formed the Secure Digital Music Initiative (SDMI) group to develop a method to deter music piracy. The SDMI developed a system that would transparently overlay copyright information on music files. Their intention was to use this "watermarking" in conjunction with approved digital music players to prevent the playing or copying of unapproved music. By September 2000, the SDMI had developed four potential watermarking standards, and they issued a challenge to help them choose which technology to implement. During the three-week public challenge, researchers would download watermarked samples for each potential standard. The participants would then attempt to remove the protections from the music files. Anyone who was successful in removing the watermarks would be eligible for a $10,000 prize.[128]

In November 2000, a team of researchers from Princeton and Rice successfully removed all four proposed watermarks. Led by Princeton's Felten, the team notified the SDMI that they had completed the challenge and that they intended to present a paper describing their results at the Fourth International Information Hiding Workshop scheduled for April 25-27, 2001. Since the researchers elected to forgo the SDMI's $10,000 cash prize in order to retain the right to publish their results, the SDMI

[128] Edward Felten, et al, "Statement Regarding the SDMI Challenge," Princeton University, http://sip.cs.princeton.edu/sdmi/announcement.html, (accessed April 30, 2019).

disqualified them as entrants in the contest.[129] This determination was seemingly at odds with the terms of the challenge which allowed contestants to retain their results if they did not receive compensation.

> You may, of course, elect not to receive compensation, in which event you will not be required to sign a separate document or assign any of your intellectual property rights, although you are still encouraged to submit details of your attack.[130]

Despite the contest's contractual wording, the secretary of the SDMI Foundation, RIAA attorney Matthew Oppenheim, sent Felten a threatening letter on April 9, 2001, weeks before the scheduled conference. The letter suggested that the team could not share the results of their research as it would violate the contest agreement and subject the individuals on the team to "actions under the Digital Millennium Copyright Act." The letter then attempted to re-articulate the challenge itself as a form of academic publication, suggesting that the sharing of the research at the conference would be actions "outside the peer review process established by the Public Challenge and setup by engineers and other experts to ensure the academic integrity of this project."[131] Concerned with the threat, the researchers elected to not present their paper, "Reading Between the Lines: Lessons from the SDMI Challenge," at the Pittsburgh workshop. Instead, Felten attended and read a statement explaining how the law had been leveraged by the RIAA, SDMI foundation and others to halt the publication of their paper. Citing these threats made against "the authors, against the conference organizers, and against their respective employers," the researchers elected to not present to avoid potential

---

[129] Edward Felten, "Response to SDMI Press Release, November 8, 2000," Princeton University, https://sip.cs.princeton.edu/sdmi/nov8.html, (accessed April 30, 2019).

[130] "Click-Through Agreement for the SDMI Public Challenge," HackSDMI.org, last modified October 8, 2000, https://sip.cs.princeton.edu/sdmi/clickthru.pdf, (accessed April 30, 2019).

[131] Matthew J. Oppenheim, Esq., "RIAA/SGMI Letter, April 9, 2001," Princeton University, April 9, 2001, https://sip.cs.princeton.edu/sdmi/riaaletter.html, (accessed April 30, 2019).

litigation but vowed to fight for the right to publish the research.[132] And fight they did; on June 6, 2001, with the assistance of the Electronic Frontier Foundation, Felten and his fellow researchers filed suit against the RIAA. Two months later, on August 15, 2001, the researchers decided to proceed with the publication of their research. Despite the RIAA's threats, they presented their paper at the 10th USENIX Security Symposium in Washington, D.C.[133]

Dmitri Sklyarov

On a warm Monday morning in July 2001, 27-year-old programmer Dmitri Sklyarov was packing his suitcase in preparation for his flight from Las Vegas back to Moscow. The graduate student was visiting the United States to present at a conference. His talk was "E-Book Security: Theory and Practice," a seminar based on the dissertation that he was completing at Moscow State Technical University. In addition to attending the university, the graduate student supported his wife and two young children by also working for ElcomSoft, a small, 20-person software company located in Russia's capital. Amid preparations for his long flight home, the young programmer was interrupted with a surprise visit from the FBI. The agents arrested Sklyarov and detained the young programmer without bail, pending transfer to a federal facility where Sklyarov would face charges for violating the Digital Millennium Copyright Act. Sklyarov's crime was working for ElcomSoft, helping them develop original software – an eBook reader. [134]

---

[132] Edward Felten, "The following statement was read by Edward W. Felten at the Fourth International Information Hiding Workshop, in Pittsburgh, on April 26, 2001," Princeton University, April 26, 2001, https://sip.cs.princeton.edu/sdmi/sdmimessage.txt, (accessed April 30, 2019).

[133] Edward Felten et al, "Reading between the Lines: Lessons from the SDMI Challenge," (paper presented at the 10th annual meeting USENIX Security Symposium, Washington, D.C., April 15, 2001).

[134] Jennifer Lee, "Technology; U.S. Arrests Russian Cryptographer as Copyright Violator," *New York Times*, July 18, 2001, C8.

ElcomSoft's core business revolved around a product line of password recovery tools for various applications. The company's Advanced eBook Processor product allowed the purchasers of e-books to read their books on a variety of e-book readers. This made proprietary e-book formats, such as Adobe's PDF and eBook Reader, compatible with the consumer's preferred book reading software. The software also allowed the books to be backed up, a right afforded to consumers by copyright law and established jurisprudence. To achieve this compatibility, however, the program circumvented the digital protections that had been added to the e-book files. Believing that the program violated the Digital Millennium Copyright Act, the Adobe corporation contacted the FBI. The company informed the agency that one of ElcomSoft's employees, Dmitri Sklyarov, would be attending a conference in July 2001. Acting at the corporation's behest, the FBI took the programmer into custody and charged him with the criminal violation of the DMCA.[135]

The detention of Sklyarov resulted in protests throughout the world. The initial focus was on Adobe, which was subject to a boycott and was the focus of a letter-writing campaign. The Electronic Frontier Foundation scheduled a demonstration at Adobe's San Jose, California headquarters but postponed the event when the vendor agreed to meet with them.[136] Following their meeting with the EFF, on July 23, 2001, Adobe publicly called for the release of the Russian programmer while re-iterating their commitment to the new copyright law, stating:

[135] Steven Levy, "Busted by the Copyright Cops," *Newsweek*, August 26, 2001, https://www.newsweek.com/busted-copyright-cops-151703, (accessed April 30, 2019).

[136] Kim Zetter, "Hacker's Arrest Prompts Protest Against Adobe," *PC World*, July 23, 2001, https://www.pcworld.idg.com.au/article/43670/hacker_arrest_prompts_protest_against_adobe/, (May 1, 2019).

> We strongly support the DMCA and the enforcement of copyright protection of digital content…; however, the prosecution of this individual in this particular case is not conducive to the best interests of any of the parties involved or the industry. [137]

Adobe's support for the release failed to affect the programmer's incarceration, and on July 30, 2001, the EFF joined a rally in San Francisco, while other similar demonstrations were held the same day in Los Angeles, Boston, Chicago, and Minneapolis. In the weeks that followed, protests were held throughout Europe and Russia, but they had no effect.[138] On August 28, 2001, Dmitri Sklyarov and his employer, ElcomSoft were indicted on five counts by a grand jury. The charges included "aiding and abetting circumvention offenses under the DMCA and a charge of conspiracy to traffic in a circumvention program." Based on the charges, the young programmer faced a fine of up to $2,350,000 and 25 years in prison.[139] The announcement prompted Free Software community members, who were in San Francisco for the Linux World Expo, to march to the U.S. Attorney's office in a rally against the indictments.[140]

The software industry lauded the charges the day following the grand jury's indictments. The software manufacturing trade group, the Business Software Alliance (BSA), whose members included Apple, IBM, and Microsoft, issued a press release about the first arrest made under the law, asserting the legality of the action, by saying the indictment "under the DMCA is consistent with the plain reading of the law and with Congress's intention when the law was drafted and enacted." The notice also commended the recently completed U.S. Copyright Office's study, which, despite

---

[137] "EFF Effector," July 27, 2001, https://www.eff.org/effector/14/16, (accessed May 1, 2019).

[138] Gabriella Coleman, "Code Is Speech," *Reason*, https://reason.com/2013/03/21/code-is-speech, (accessed May 1, 2019).

[139] "Us v. Elcomsoft Sklyarov," Electronic Frontier Foundation, https://www.eff.org/cases/us-v-elcomsoft-sklyarov, (May 1, 2019).

[140] Coleman, "Code Is Speech."

protests and vocal opposition at the public hearings, determined there was "no need or justification for changing the anti-circumvention provisions of the DMCA."[141]

Rulings on the DMCA

On May 1, 2001, Eric Corley's *2600* freedom of speech case, Universal City Studios, Inc. v. Reimerdes was decided by the U.S. Court of Appeals for the Second Circuit. The decision acknowledged software as a form of speech but held that it was constitutional to limit his right to free speech due to the nature of the content. Corley elected not to appeal to the Supreme Court. The MPAA's efforts to stifle the spread of the software in question was fruitless, however, as both the DeCSS code and knowledge of how to circumvent the 40-bit encryption on DVD discs quickly spread through the Internet. Despite this, the MPAA also pursued John Lech Johansen for his role in disseminating the code. The youth went to trial several times in Norway but was ultimately acquitted of all charges as his behavior had been legal under Norway's own fair use laws.[142]

Edward Felten's case against the RIAA for their attempt to censor his SDMI research was dismissed by a New Jersey Federal court in November 2001. In February 2002, the researcher announced he would not appeal the ruling, citing "assurances from

[141] Patrick Mellody, "Statement of Business Software Alliance President and CEO Robert Holleyman Concerning the First Indictment and Arraignment under the DMCA," Business Software Alliance, August 30, 2001, http://web.archive.org/web/20010917030022/http://www.bsa.org/usa/press/newsreleases/2001-08-30.692.phtml, (accessed May 1, 2019).

[142] Timothy O'Brien, "Technology; Norwegian Hacker, 19, Is Acquitted in Dvd Piracy Case," *New York Times*, January 8, 2003, C4. https://www.nytimes.com/2003/01/08/business/technology-norwegian-hacker-19-is-acquitted-in-dvd-piracy-case.html, (accessed May 8, 2019).

the government, the recording industry, and a federal court that the threats against his research team were ill-conceived and will not be repeated."[143]

On New Year's Eve 2001, after nearly six months in custody, Dmitry Sklyarov was released and allowed to return home to his family in Russia. The charges against the programmer were dropped on the condition that the programmer would testify against his employer. The government's effort was fruitless as a jury acquitted ElcomSoft of all charges on December 16, 2002.[144]

The cases that brought the Free Software community into the streets in protest were just the start of a larger trend of corporations using the DMCA to censor criticism or hobble competitors. Examples abound: In 2002, Blizzard Entertainment, a computer gaming company brought a suit against the Free Software project, Bnetd, to stop the programmers from writing software that interacted with their video games.[145] Lexmark, a printer manufacturer, brought a DMCA suit against competitors in 2004 in an attempt to forbid them from manufacturing printer consumables such as ink cartridges.[146] In 2005, researchers discovered that Sony Music CDs were deliberately installing malware on consumer's computers. Sony's XCP software automatically installed itself without the user's permission and enabled the company? to gain control of the consumer's computer

---

[143] "Felten, et al., v. RIAA, et al.," Electronic Frontier Foundation, https://www.eff.org/cases/felten-et-al-v-riaa-et-al, (accessed May 8, 2019).

[144] Matt Berger, "Jury: Elcomsoft Not Guilty," *NetworkWorld*, December 17, 2002, https://www.networkworld.com/article/2339223/jury--elcomsoft-not-guilty.html, (accessed May 8, 2019).

[145] Kenneth Hwang, "Blizzard Versus Bnetd: A Looming Ice Age for Free Software Development?" *Cornell Law Review* 92 (February 2013), 1044-1046.

[146] "Lexmark international, Inc. V. Static control components, Inc.," *Harvard Law Review* 128, no. 321 (November, 2014),321-330, https://harvardlawreview.org/2014/11/lexmark-international-inc-v-static-control-components-inc/, (accessed May 8, 2019).

system. Princeton's Felten found himself once again facing DMCA censorship efforts when he endeavored to make the security risk publicly known.[147]

Summary: Toward Activism

The corporate friendly Open Source re-articulation succeeded in encouraging corporate adoption of the software. Oriented toward compatibility with commercial systems, the low cost provided the programming frameworks for many promising web-based corporations. Academic interest in the corporate adoption of the software helped to define the movement's schema as Commons-based Peer Production. Meanwhile, the practices and ideals began to evolve away from software, informing a myriad of projects including a free, universal encyclopedia, a cooperative legal support system, and a democratized web publishing platform. The transformations that occurred were not limited to Free Software. As the U.S. government struggled to understand the new online communities and began to clamp down on anything redolent of cracking. Then, with the passage of the Digital Millennium Copyright Act (DMCA) in 1998, the government criminalized the interoperability-focused programs commonly written and shared in the Free Software community. The new law served as a litigious cudgel that could be leveraged by corporations to censor publications they objected to and to seek the arrest of programmers. Outraged at what they perceived to be attacks on their first amendment rights and fellow programmers, members of the Free Software community throughout the nation took to the streets in protest.

[147] Deirdre Mulligan and Aaron Perzanowski, "The Magnificence of the Disaster: Reconstructing the Sony Bmg Rootkit Incident," *Berkeley Technology Law Journal* 22, no. 3 (Summer 2007), 1998-1200.

CHAPTER 8: FROM ACTIVISM TO ANARCHY

By the end of the 20th century, Open Source, the corporate-friendly re-articulation of Free Software succeeded in encouraging commercial interest in their programs. The low-cost software was oriented toward networking and interoperability.[1] It also provided the Internet platform for many emerging web-based corporations. Academic interest in the corporate adoption of the software helped to define the movement's schema as Commons-based Peer Production. Meanwhile, the practices and ideals began to evolve away from software, influencing the creation of projects including Wikipedia, the Creative Commons, and WordPress. The Free Software ideology was also evolving into new types of activism: In Texas, a group of Free Software hackers formed a group with the purpose of composing software, such as tools for computer-based activists to perform on-line activism, or as they named it, "Hacktivism." The ideas put forth by this cadre of hackers were soon weaponized by an anonymous group of Internet users who found themselves censored by the Digital Millennium Copyright Act. Meanwhile, at Cygnus, a Free Software support company, a group of ideologues began to compose Free Software that was expressly designed to change how society itself functioned. Just as Richard Stallman had used his skills to create a sphere outside of the control of corporations, this group's programming focused on creating a new digital realm outside the control of the world's governmental structures. Their handiwork soon challenged global law enforcement, banking, and geopolitical structures.

[1] Interoperability in this context refers to the ability of disparate computers to work in conjunction with each other. As discussed in the previous chapter, many popular Free Software projects like SAMBA grew out of individual needs to accommodate proprietary technologies (in the case of SAMBA, Microsoft's proprietary file and print sharing services.

Part 1: From Activism to Hacktivism

The DMCA brought the Free Software community to protest in the street. The law was used by corporations to stifle freedom of speech and to justify the arrest of computer programmers. The same month that arrests under the DMCA began, the community also introduced a new concept: Hacktivism. This movement was based on the idea that software could be written that would assist activists in combating issues, such as human rights violations, on the Internet. Topics which would normally be resisted through direct protests, could now be actively opposed within the digital sphere using Free Software designed specifically for that purpose. The origin of this new type of activism can be traced to the mid-1980s formation of a hacking group, The Cult of the Dead Cow.

The *Cult of the Dead Cow*

In 1983, Kevin Wheeler, a junior high school student in Lubbock, Texas, began a dial-up Bulletin Board System (BBS) entitled the "Pan-Galactic Entropy Association." Inspired by the hacking seen in the film *War Games*, Wheeler and a visitor to his BBS, Franklin Gibe, decided to start their own cracking group. As with other hacking crews such as the New York City-based Masters of Deception, the boys adopted pseudonyms -- Wheeler became "Swamp Ratte" (later "Grandmaster Ratte") and Gibe styled himself "Sid Vicious." They named the group "The Cult of the Dead Cow," (CDC) in reference to an abandoned slaughterhouse that became a common hangout for the teens in Lubbock.[2] Initially, the group was more social than technical. The youths published online magazines in text files. The topics covered everything from UNIX commands to

[2] Joseph Menn, *Cult of the Dead Cow: How the Original Hacking Supergroup Might Just Save the World* (New York, New York: Public Affairs, 2019), 9-15.

lyrics for songs by the punk band The Dead Milkmen.[3] As the CDC group grew, the topics became more varied. For example, in a 1987 text file, group member Beto O'Rourke (aka "Psychedelic Warlord") postulated about the societal changes required to have a world without money.[4] As skilled new members joined, the group grew in prominence, and began to present at cracker conferences, such as the yearly DEF CON convention.[5] By the late 1990s, the group was using the annual gathering to announce new software. Where other cracking groups focused on compromising specific corporations or networks, the CDC skirted legal concerns by creating tools designed to exploit security holes in commercial products. Most famously, in 1999, the CDC introduced "Back Orifice 2000," a Free Software package that used security holes within Microsoft's Windows operating system to provide control over a remote computer.[6] The group presented the program as a remote management tool for Windows administrators since it offered full access to the computer's desktop and was able to remotely alter system files, including editing the computer registry.[7] Once installed on a target computer, the software communicated outbound to the Internet, providing access to systems even if they were protected by a network firewall.[8] The software also contained functionality commonly found in a cracker's toolbox – it could record user keystrokes

[3] The group's logo, a text-based representation of a deceased bull, bears a strong resemblance to the logo of the band, suggesting that band name may also have been a factor in the group's naming decision.

[4] Beto O'Rourke, "A Feature On Money - Today's Monster," Cult of the Dead Cow, http://textfiles.com/groups/CDC/cDc-0031.txt, (accessed July 3, 2019).

[5] Menn, 62.

[6] Back Orifice 2000 was released as Free Software under Richard Stallman's GNU Public License.

[7] Back Orifice was a play on Microsoft's "Back Office" product, a server software suite that allowed Windows NT to act as a server for databases, email, and the world wide web,

[8] Firewalls were programs (on local computers) or devices (on networks) that can restrict certain types of network traffic. These are typically used to restrict network traffic coming into a local network to secure the local network from Internet-based threats. Although best practice, it was uncommon to see local computers running firewalls or to see outbound (egress) network filtering. The need for the latter is demonstrated by the way in which this software communicates.

(which crackers used to learn logon names and passwords), while hiding its presence on the system from the users (same as viruses, rootkits, and other forms of malware.)[9]

The Cult of the Dead Cow's Back Orifice brought the group notoriety; following their presentation of the software, the group members were interviewed by a number of major news outlets, including *Businessweek*, CNN, NPR, and the BBC.[10] It also set them apart within the cracking community – instead of cracking, they created tools which others could use to crack. At the same time, they justified their offerings through their legitimate use. The "Back Orifice" software could be used by crackers to compromise systems – or it could be legitimately employed by system administrators to maintain their network. At the very least, the program demonstrated threats inherent in Microsoft's insecure operating systems. Had Microsoft's product been secure to start with, the CDC would not have been able to compose the exploit.

The group's rising exposure propelled some members to expand their horizons. CDC member Peiter Zatko (aka "Mudge") and others within the group formed L0pht Heavy Industries, a security consulting group based out of Boston. According to Zatko, L0pht was intended to raise awareness of security issues in a manner like "*Consumer Reports* and Rachel Carson and Ralph Nader." Other members, such as Misha Kubeka (aka "Omega"), moved to California in the early 1990s and by the latter part of the decade served to welcome other CDC members migrating to Silicon Valley as part of the

[9] Menn, 65-72.
[10] Ibid., 67.

.com boom.[11] Almost immediately upon his arrival, Kubeka became involved with the hacking counter culture in the Bay Area. He befriended Jude Milhon (aka "St. Jude"), a 1970s hacker who helped co-found the Community Memory Project, the first public computer Bulletin Board System (BBS), as detailed in Stephen Levy's book *Hackers*.[12] Milhon later wrote Free Software for BSD UNIX and by this time, had become an editor of the counter-culture technology magazine *Mondo 2000*.[13] Milhon helped Kubeka get published by the periodical shortly after he moved to the area in 1992.

By 1998, the Cult of the Dead Cow had established themselves as celebrity hackers. Some of the group's members saw the exposure as an opportunity. Member Laird Brown (aka "Oxblood Ruffin") posited this to the group: "You have a little window of fame right now, what do you want to do with it?" Brown, who had previously worked for the United Nations, was passionately opposed to political oppression throughout the world. He encouraged the group to expand beyond just computer security and work to advance human rights. Misha Kubeka helped publish a story about the plight of a political dissident group in China. Kubeka also introduced a term for what the group was endeavoring to do: Hacktivism, a portmanteau of hacking and activism, was the approach of using technology and electronic media to advance activist issues, including human rights. The CDC tested the idea by publishing a press release encouraging democracy advocates in China to use their Back Orifice software to attack businesses

[11] Ibid., 73, 65. In May 1998, seven of the L0pht programmers testified before Congress about computer security concerns. Peiter "Mudge" Zatko acted as the de facto spokesperson and later ended up being hired by the government to manage aspects of cybersecurity.

[12] Levy, *Hackers*, 283.

[13] Jude Milhon also named the Cypherpunks, a group for which she was also a founding member. The Cypherpunks will be discussed in detail later in this chapter. Milhon brought Kubeka into this sphere as well, introducing him to her then-boyfriend, Eric Hughes a seminal member of the Cypherpunk group.

associated with the Chinese government's oppressive regime.[14] It is not clear what effect, if any, that the statement had on supporting the Chinese democracy movements, but the announcement was just the start of the group's activism. The CDC soon began a deliberate focus on hacktivism, through the creation of a subgroup dedicated to the activity: Hacktivismo.

Hacktivismo

On July 4, 2001, the CDC announced an offshoot project, Hacktivismo. The project was intended to create tools to assist political activists communicate securely and evade censorship. The effort was announced via the "Hacktivismo Declaration." Written in a format similar to a U.N declaration, the document conveyed the group's intentions. "We recognize the right of governments to forbid the publication of properly categorized state secrets, child pornography, and matters related to personal privacy," the document read, "but we oppose the use of state power to control access to the works of critics, intellectuals, artists, or religious figures." State control of the Internet was "a serious form of organized systemic violence against citizens," which fostered "xenophobia and destabilization of international order." As such, the group declared that the "international hacking community has a moral imperative to act" and that they would meet this obligation by developing ways to circumvent state sponsored censorship and would "implement technologies to challenge information rights violations."[15]

---

[14] Menn, 65.88-91, 96-97.

[15] "The Hacktivismo Declaration," Hacktivismo, July 4, 2001, http://www.hacktivismo.com/public/declarations/en.php, (accessed July 3, 2019).

Within two weeks of publishing the declaration, several CDC members presented a panel on hacktivism at the 2001 DEF CON computer security conference in Las Vegas. The group explained what hacktivism was, and Laird Brown read the Hacktivismo declaration. The presentation had so many registered attendees that the convention's organizers were forced to host the talk in a tent pitched on the hotel roof; there were no convention facilities large enough to accommodate the crowd.[16] With Hacktivismo, the CDC was no longer just producing software that exploited security holes – they were now developing programs expressly designed to circumvent governmental oversight. For their tools to be effective, they intended to distribute this Free Software to dissident groups throughout the world. To accomplish this, they had to find a way to license the software consistent with their goals.

The Hacktivismo Enhanced-Source Software License Agreement

The CDC's previous software releases had been Free Software. Back Orifice 2000, for example, had been released under Stallman's GNU Public License (GPL). The problem was that the GPL and other Free Software licenses were insufficient for Hacktivismo's political goals. There were no free (or Open Source) software licenses that qualified the intent or prerogative of any licensee. As was Stallman's original intention, users of Free Software could do anything they would like with it, which was problematic in Hacktivismo's perspective, as this freedom could be abused by oppressive governments. For example, a repressive regime could take a Free Software program and could theoretically modify it to include a virus or use it for surveillance of dissident groups. No Free Software license agreement could "in any way contain (as a license

[16] Menn, 101.

term) any restriction on the use of software."[17] For the Hacktivismo programmers, this represented a problem, as they enthusiastically endorsed and supported the goals of the Free Software Movement. Their solution: create an "enhanced" Free Software license.

The Hacktivismo Enhanced-Source Software License Agreement (HESSLA) was released on November 25, 2002. Heavily influenced by the GNU public license, the document made clear that it ensured the same rights, specifically the benefits of freedom to distribute copies, access to the source code, and freedom to make changes – so long as the use and changes did not "subvert or infringe the freedoms of end-users." The license specifically endorses the use of Free Software and even links to the Free Software Foundation page describing the movement's ideology. But the core of the license is focused on the augmentations that the hacktivists felt needed to be added to Free Software. The document emphasizes the requirement that the software "cannot be used to violate human rights or forbidding the insertion of "spy-ware" or surveillance mechanisms into derivative works." To eliminate any ambiguity with regards to the freedoms it sought to protect, the license detailed them: Freedom of expression, collective action, thought, conscience, sexuality, religion, and privacy. To underscore their ambitions, the authors also referred to relative articles within the Universal Declaration of Human Rights.[18]

Where Free and Open Source licenses were designed to afford the licensee absolute freedoms to use the software, the HESSLA afforded the same freedoms – provided that the use accorded with the international standards for human rights. And just

---

[17] "The Hacktivismo Enhanced-Source Software License Agreement," Hacktivismo, November 25, 2002, http://www.hacktivismo.com/about/hessla.php, (accessed July 3, 2019).

[18] Ibid.

as Free Software was intended to create a software ecosystem outside of the control of corporations, the HESSLA sought to create software outside of the control of repressive governments– tools with which rebels could challenge the regimes themselves.

Waging Peace on the Internet

The motivations behind Hacktivismo and the license can be best understood in concert with "Waging Peace on the Internet," a treatise written by CDC/Hacktivismo member Laird Brown several months prior to the release of the HESSLA license. Brown provided examples of oppressive Internet censorship in which online activities resulted in real world consequences. In the Ukraine, for example, the founder of an on-line newspaper critical of the government was located, abducted, and de-capitated by the authorities. Brown described hacktivism as the group interpreted it, "using technology to advance human rights through electronic media." The author explained the importance of open code within the movement. Stepping beyond the existing arguments for Free Software, which could already be garnered from "the demotic juggernaut of the Free Software Foundation, to the debate laden pages of Slashdot," Brown instead rationalized Free Software as a unifying component for hacktivists, because it is demonstrative:

> Open code, like the open and inclusive nature of democratic discourse itself, will prove to be the lingua franca of hacktivism. And perhaps more importantly, it will demonstrate that hacktivists are waging peace, not war.[19]

Brown went on to cite Marshall McLuhan's 1970 text *Culture Is Our Business*, where the communications scholar pronounced that "World War Three will be a guerilla

[19] Laird Brown, "Thoughts On Hacktivism from the CDC," The Register, April 19, 2002, https://www.theregister.co.uk/2002/04/19/waging_peace_on_the_internet/.

information war with no division between military and civilian participation."[20] Brown suggested that McLuhan's prediction had come to pass, and the opening battles of the information war had been joined. Hacktivismo was the Cult of the Dead Cow's foray into the guerilla conflict; the CDC was intent to not let the "Internet's lights dim simply because liberal democracies are asleep at the switch." The transparency and freedom of using open code – be it Free Software or Open Source, would differentiate the hacktivists from the oppressive governments which sought to use the Internet as a thought control tool by employing extensive censorship and national propaganda.[21]

In the treatise, Brown not only supported Free Software, but also its ideology. He made clear that for hacktivism to be legitimate it had to support the type of freedoms that underlay the Free Software Movement, such as the free flow of information. Online activists who defaced websites or launched any form of digital attack performed "nothing more than hi-tech vandalism…an assault on free speech."[22] Just as McLuhan predicted a global information war, Brown was prescient in describing the tactics employed in the next phase of the conflict. Within a year of Brown's publication, a loose affiliation of anonymous activists began to employ Free Software programs to deface websites and disable access to them through Distributed Denial-Of-Service (DDOS) attacks.[23]

Anonymous

Where Cult of the Dead Cow and Hacktivismo embraced Free Software's ideology and schema and extended it to geopolitical issues with oppressive governments,

---

[20] Marshall McLuhan, *Culture Is Our Business* (Eugene, Oregon: Wipf and Stock, 2015), 66.
[21] Laird Brown, "Thoughts on Hacktivism from the CDC."
[22] Ibid.
[23] A Distributed Denial-of-Service attack is when an attacker uses software to overwhelm a server or network with a flood of traffic in an attempt to disrupt the normal activities of that server or network.

the online activist group, Anonymous, adopted the software and the schema but acted in opposition to the movement's ideology. In 2006, Anonymous, began using Free Software to suppress the online activities of the group's targets – everything from social media websites to the Church of Scientology.

The concept of anonymous posting predates the Internet. There were anonymous chatrooms and posting groups back on the pre-Internet online community, USENET. On the Internet, anonymous communities emerged on sites such as 2chan, 711Chan, and 4Chan. The latter was a popular image channel that emerged as a meeting place for anonymous users who frequently posted materials banned from other sites. Almost from the channel's inception in 2003, the anonymous members performed various forms of "trolling," Internet-based pranks and attacks on others. The anonymous denizens of 4Chan initially attacked individuals. They began coordinated "raids" in July 2006 when the group began to harass participants on a Finnish online platform called the Habbo Hotel, a virtual meeting space for teenagers. Visitors explored the hotel as avatars representing themselves. They could also create custom guest rooms in the Habbo Hotel, complete with "furni," virtual furniture. On July 6, 2006, members of the newly formed Anonymous group began to log on to the site. They created matching avatars, "black men in gray suits with prominent afros." The anonymous interlopers assembled their on-line characters into human formations resembling swastikas. The also formed a picket line to prevent the site's legitimate adolescent visitors from visiting the hotel's pool. Any Habbo visitor unfortunate enough to inquire about Anonymous's actions received the cryptic answer that the pool was closed “due to fail [SIC] and AIDS.”[24] The motives behind the collective action against the youth side are unclear, but it did not matter – Anonymous was just getting started.

Project Chanology

In 2008, a video of Tom Cruise gushing over Scientology was leaked to the Internet. Scientologists moved to have the video taken down and attacked the anonymous anti-Scientologist Internet commenters as being paid shills employed by the “psychiatry industry.” The Scientologists issued DMCA takedown notices to remove the Tom Cruise video.[25] In response, the anti-Scientologists styled themselves “Anonymous” and launched into what they considered a “cyber-sit in,” a distributed denial of service attack in protest of the Scientologist’s exploitation of the DMCA as censorship mechanism.[26]

The Anonymous activists used Free Software tools to perform their attacks. Initially, they used Apache J-Meter, software developed by the Apache web server project. J-Meter was intended to flood a website with test traffic to assess its ability to handle high levels of visitors prior to being put into production. The group soon moved on to another software package that performed a similar function, the Lo-Orbit Ion Cannon. As the Anonymous group expanded the scope of their targets, they adopted additional distributed attack software such as Havij, H.O.I.C., and PyLoris. Each of the tools allowed the Anonymous attackers to flood Scientology websites with web traffic

[24]Gabriella Coleman, *Hacker, Hoaxer, Whistleblower, Spy: The Many Faces of Anonymous* (London: Verso, 2014), 44-46,4-5.

[25] For a more complete understanding of Anonymous actions regarding the Digital Millennium Copyright Act, see Marcia K. Wilbur’s survey, *A Decade of the DMCA* (Middletown, Delaware: lulu.com, 2009).

[26] Ibid., 250-253.

that overwhelmed the web server, causing it to crash and take the organization's content off-line.

The on-line attacks were soon joined by real life protests. On February 10, 2008, Anonymous protesters throughout the world took to the streets. Donning Guy Fawkes masks popularized by the 2005 film *V for Vendetta*, the protestors congregated outside of Scientology buildings. They performed chants and brandished signs bearing anti-Scientology slogans. When interviewed by the *Boston Globe*, protesters in Boston reported that they knew no one else at the event or who led the group, with one participating stating, "That's Anonymous. There is no hierarchy. Everyone is in charge of themselves."[27]

Beyond Scientology

This experience with Scientology established Anonymous as a hacktivist group and inaugurated their approach of combining real-world protests with distributed Internet-based attacks. Since these initial raids, the Anonymous group has protested against many corporate targets including Sony, Amazon, and PayPal, as well as geopolitical foils such as to the governments of Israel, Libya, and Spain.[28] The decisions to protest targets were made democratically via web voting software.[29] Since the group's inception, the Anonymous protestors frequently joined with other protest groups

---

[27] John Forrester, "Dozens of Masked Protesters Blast Scientology Church," *Boston Globe*, February 11, 2008, http://archive.boston.com/news/local/articles/2008/02/11/dozens_of_masked_protesters_blast_scientology_church/, (accessed July 3, 2019).

[28] The inclusion of the Anonymous group in this dissertation is limited to providing it as an example of how the software and schema of Free Software can be employed by groups that seemingly do not accord to the Free Software ideology.

[29] Rohit Shaw, "Weapon of Anonymous," Infosec Institute, August 29, 2013, http://resources.infosecinstitute.com/weapon-of-anonymous/, (accessed April 20, 2015).

such as the international Occupy movement, who protested social and economic inequality.[30]

Since the hacktivists are, by definition, anonymous, there is not sufficient data to support a conclusion about the group's ideology. The faction is voluntary and chooses its targets democratically, so in substance, it is the methods and actions that define Anonymous. Specific targets aside, the organization's approach closely resembles the distributed open schema found within free and Open Source software. The group also employs Free Software tools to plan and launch their attacks.[31] However, where Anonymous adopted the methodologies of Free Software, they do not observe Free Software ideologies. Anonymous employs peer-based methods to perform distributed denial of service attacks with the intention of knocking their victims offline. This is essentially an Internet-based assault on free speech. Since the Free Software Movement is predicated on the idea that code is speech and their defining goal is for freedom of expression with said code, Anonymous's suppression of free speech is incongruent with the Free Software Movement's philosophies.

The Advent of Hacktivism

The Cult of the Dead Cow introduced the idea of hacktivism and formed the Hacktivismo group to distribute tools for resisting oppressive regimes. Hacktivismo embraced Free Software ideologies and extended the beliefs to achieve geopolitical objectives. Despite this seminal role in online activism, Hacktivismo's actions were overshadowed by the denial of service raids conducted by Anonymous. Anonymous took

---

[30] Ginny Graves, "Attention Rapists: You've Met Your Match," *Glamour*, June 6, 2013, https://www.glamour.com/story/attention-rapists-youve-met-your-match, (accessed July 3, 2019).

[31] Rohit Shaw, "Weapon of Anonymous."

Free Software tools written by others and used them to silence the web presence of individuals, organizations, and nations that the group took issue with. Where they adopted the programs and distributed approach of the Free Software Movement, Anonymous's tactics, which suppressed free speech, were antithetical to Free Software's ideology. However different, both groups were key in bringing the concept of Internet activism, or hacktivism, into the public consciousness, but they were both just minor belligerents in the global information war. While Anonymous' raids dominated the headlines, a collection of cryptography-obsessed anarchists and libertarians were fashioning software tools intended to remake the structures of society itself.

Part 2: From Hacktivism to Anarchy

While the individuals behind Anonymous leveraged technology in a very public manner, a different group of programmers called the Cypherpunks labored silently to produce software designed to change society. The faction employed Richard Stallman's ideology and schema for Free Software, but extended it, adding in their own focus on cryptography. Where Stallman's movement was designed to create a software ecosystem outside of corporate control, the Free Software written by the Cypherpunks was designed to create a private sphere free from governmental constraints. And just as Stallman was reacting to corporate driven changes in computing, the Cypherpunks arose in response to governmental overreach, in particular, the U.S. government's attempts to access all the telecommunications of private citizens.

Early Cryptography

Cryptography, or secret writing (derived from Greek: kryptos grapho), has existed throughout history. In his treatise, *The Histories*, Greek historian Herodotus detailed the use of *steganography*, or hidden writing, in a fifth-century conflict between Greece and Persia where messages were carved onto tablets and then hidden under wax to prevent detection. Cryptography can also be in the form of substituting words or letters to obfuscate messages. Creating a lexicon of alternate words is a *code*.[32] Military endeavors frequently use code words to hide the true nature of planned activities, as seen in the Manhattan Project, the World War II era effort to refine fissile material and create nuclear weapons.[33] Replacing letters rather than words is called a *cipher*. An early account of this is seen in Roman historian Suetonius' second century AD text, *Lives of the Caesars LVI*. The history details Julius Caesar using a substitution cipher in which the alphabet was shifted three places:

Figure 1: A Caesar Shift Cipher

Alphabet	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Cipher	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

Example text:

Plaintext: Cross the Rubicon
Ciphertext: Furvv wkh Uxelfrq

Source: Adapted from Singh, 10.

[32] Simon Singh, *The Code Book* (New York, New York: Anchor Books, 1999), 4-5,30.

[33] "Atomic Glossary," Atomic Heritage Foundation, June 18, 2014, https://www.atomicheritage.org/history/atomic-glossary, (accessed July 7, 2019).

This simple cipher allowed the Roman leader to send hidden messages to anyone who knew the encryption method, or *algorithm*, as well as the specifics of the encryptions being used, the *key*. In the example of the *Caesar shift cipher*, the method is to shift the letters, and the key would be the specifics of exactly how to shift it (e.g. three places).[34] Caesar's cipher was relatively weak in that the algorithm could be guessed and key could be tested by shifting through each of the Latin alphabet's 23 characters.

Both codes and ciphers suffered from another weakness as well: in order to work, the person creating, or *encrypting* the message needed to be able to share the encryption information with their intended recipient. Without knowing the algorithm, key, or code definitions, the message would not be able to be easily *decrypted*, translated back into its plain text form. This information would have to be exchanged prior to the transmission of the encrypted messages, but this too would create a problem. For example, if the key stayed perpetually the same, then the messages would be more easily decrypted by anyone intercepting the transmission; they need only learn to decipher the first communication and they would be able to read all subsequent dispatches. The solution was to have a key that could be changed periodically. For example, the Caesar cipher shift could be changed with each message according to a pre-set pattern. This would offer some security in such that the key would not remain the same but eventually, the pattern might be identified. The only alternative was to periodically re-exchange keys (or codewords). However, this too presents a challenge in that the information might be intercepted during the exchange.

[34] Singh, 10-11.

Cryptography and Early Computing

ENIGMA

In 1918, German inventor Arthur Scherbius created an electronic cipher machine named the Enigma. The device had three main components – a keyboard where each plaintext letter would be typed, a scrambling unit that converted the entry into ciphertext, and a display board where a lamp lit up the corresponding ciphertext letter. Within the scrambling unit, there were three rotors, each of which scrambled the content in diverse ways. Each rotor had 26 different starting positions – one for each letter of the alphabet. The operator would set the rotors to pre-determined starting positions – the key – for starting the encryption process. The machine would then perform a substitution cipher using the rotors. The message would be encrypted through the settings on the first rotor and then the output from that would then be encrypted again using the settings on the second rotor. The device allowed for 17,576 possible key arrangements. On top of this, as each letter was typed, the machine incremented both rotors by one position, adding additional complexity to the encryption process.[35] To decrypt the message, the recipient would need both the machine and the starting position and order of the rotors.

Scherbius offered the product commercially but found little success due to the high cost of the device ($30,193 in 2020 dollars.) In 1925, Scherbius's fortunes began to change when the German military concluded that the Enigma machine would help them avoid the cryptographic issues they experienced during the First World War, where the British were able to easily intercept and decrypt German communications. The Enigma

[35] Ibid., 127, 132-136. For simplicity sake, the description here is of Scherbius's original machine. He later expanded the encryption strength by adding additional features such as a plugboard and a reflector.

was modified to suit the needs of the military, which purchased over 30,000 of the devices in the decades that followed.[36]

Solving the Puzzle

At the onset of World War II, the Enigma encryption offered the Nazi regime a clear advantage. Using the system, the military forces were able to secretly coordinate the fast paced, rapidly changing military maneuvers that later became known as *blitzkrieg*, or lightning war. Unsurprisingly, cracking the Enigma encryption quickly became an objective for the Allies. Luckily, several important advances had already been made in the lead up to the war. In November 1931, Hans-Thilo Schmidt, a failed business owner from Germany, met with a French secret agent in Belgium, and in exchange for 10,000 marks, he permitted the agent to photograph documentation on how the Enigma machine was used. This allowed the allies to create an accurate replica of the device. Polish cryptographer Marian Rejewski observed that there was frequently repetition at the start of each German transmission. With enough encrypted messages, Rejewski was able to begin to deduct the key being used. However, the key changed each day, forcing the process to be restarted daily. The Germans soon added components to the Enigma, such as a plug-board that allowed for additional complexity. The additions frustrated Rejewski's translation efforts. In July 1939, the Polish government shared Rejewski's advances with French and British cryptanalysts, going as far as providing them with two Enigma replicas, as well as blueprints for the Bombe, an electromechanical device designed to help ascertain the Nazi's day codes. By August 16, one of the promised

[36] Ibid., 138, 142.

replicas reached Britain. The device arrived just in time; two weeks later, World War II began with the Nazi invasion of Poland.[37]

In Britain, the group responsible for deciphering the German communications was the Government Code and Cypher School (GC&CS) at Bletchley Park, Buckinghamshire. The team incorporated the Polish contributions into their efforts. British cryptanalyst Alan Turing, who had postulated about a hypothetical universal computing machine as an academic at Cambridge, designed a new version of the Bombe, which expanded the device beyond what the Polish cryptographers had envisioned. Turing had observed that German messages sent a regular encrypted weather report each day. The daily message was formatted consistently, so the German word for weather, "wetter" was always in the same location. With this known plaintext translation, Turing's machine would be able to narrow down the possible day keys. The device would electromechanically deduce the day key through a process of elimination, trying possible rotor combinations. To hasten the calculations, each of Turing's devices featured twelve sets of Enigma rotors. The protype Bombe went into operation in March 1940. Within eighteen months, fifteen more devices were in operation, and on a good day, the devices could deduce the Enigma's day key within an hour.[38]

By the time that the GC&CS group operationalized the decrypting of Enigma devices, the Nazi high command began using a more advanced instrument for their communications. The Nazi's new Lorenz machine had ten rotors and was able to instantly transmit encrypted messages over telegraph lines. To counter this new cryptography, Alan Turning, mathematician Max Newman, and electrical engineer Tom

---

[37] Ibid., 145-146,151-155, 160.
[38] Ibid. 161,168,174-175,177.

Flowers designed a fully electronic computer designed to decrypt the messages. Dubbed the Colossus, the device used 2,500 vacuum tubes to process encrypted messages which were fed into the machine via paper tape at a rate of 5,000 characters per second. The early computer was staffed by sixty operators, women recruited from the Women's Royal Naval Services. The Colossus went into service on June 1, 1944. The device performed as designed, allowing the British cryptologists to decrypt messages vital to the Allies' Normandy landings planned for later that week. The team decoded orders sent by Nazi leader Adolf Hitler to Field Marshall Erwin Rommel. Hitler's instructions suggested that the Nazis had been deceived by an Allied operation which had fabricated a fake invasion force and suggested that the main invasion of Europe would be led by General George S. Patton at Pas de Calais in northeastern France. When the allied invasion began at Normandy, hundreds of miles away from Calais, Hitler was convinced that it was just a feint. The messages decrypted by the Colossus revealed that the Nazis would not shift forces to Normandy. The new computer provided vital information to the Allied forces just in time for them to undertake the largest amphibian landing in history.[39]

Beyond its timely contribution to the war, the Colossus marked several significant milestones. First, it was one of the first electronic computers, going live just months after the Harvard Mark 1. Second, it brought cryptography to electronic computing, demonstrating the value of the new electronic computational technologies.[40] Following the war in 1946, the group at Bletchley park, now renamed the Government

---

[39] Janet Abbate, *Recoding Gender: Women's Changing Participation in Computing*, History of Computing (Cambridge, Massachusetts: MIT Press, 2012), 14-16.

[40] The Colossus and the material surrounding the device were destroyed following the war. This had the temporary effect of concealing from the historical record the early advances the machine brought, as well as the contributions of the device's creators and WRENS operators.

Communications Headquarters (GCHQ), secretly continued their efforts in computerized cryptography. Despite the decryption issues resolved by technologies like the Bombe and the Colossus, a vital problem remained in cryptography. In order for an encrypted message to be successfully transmitted and unencrypted, both the sender and receiver needed to have the same encryption key. This meant a physical exchange of the key or of codebooks with keys printed within. These secret keys had to be written down and shared, presenting an inherent weakness.

The Diffie-Hellman Key Exchange

In November 1976, Stanford researchers Whitfield Diffie and Martin Hellman published a paper describing a solution for the problems inherent in exchanging an encryption key. Their article, "New Directions in Cryptography," observed that recent advancements in computing produced a growing need for cryptographic systems to secure data and authenticate transactions. The paper also suggested a way to minimize the need for distributing keys through the use of "the equivalent of a written signature."[41] Where traditional approaches to encryption used a single, private key to encrypt a message, Diffie and Hellman proposed that the key be split. Their paper proposed a key exchange, a digital infrastructure where there exists a key pair. This pair consisted of a private key, which would be kept secret, and a public key that would be shared widely. The public key could be used to encrypt messages that could only be opened with the private key.[42] The two keys were related mathematically in the form of a one-way

---

[41] Whitfield Diffie and Martin Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory,*22, no. 6 (November 1976), 644.

[42] Ibid., 647-649.

function, which "can be calculated easily in one direction but not easily reversed."[43] In the paper Diffie and Hellman presented an algorithm that used large prime numbers to generate the keys needed for their proposed scheme.[44]

Diffie and Hellman's design required that the encrypting key be public. Since only the decrypting key need to remain private, their proposal solved the need for the secure exchange of encryption keys. The only thing being shared was the public key, and this was unable to decrypt any secure messages.[45] The challenge with the Diffie-Hellman proposal was that it was largely conceptual. It was focused on bi-directional information exchanges but was not clearly applicable to activities such as email. And where they proposed the idea of using a one-way function for encryption, and speculated on ways that it might be done, they did not establish a proof that would do so, leaving this idea open for future research.[46]

The RSA Cryptosystem

---

[43] Steven Levy, *Crypto: How the Code Rebels Beat the Government, Saving Privacy in the Digital Age* (New York, New York: Viking, 2001), 28.

[44] Diffie-Hellman, 647-649.

[45] Their proposal included another idea: digital signatures through asymmetric encryption. Prior to Diffie and Hellman's proposal, encryption was symmetric; there was only one key and it was used to both lock and unlock the message. This encryption can be compared to a locked briefcase. A document could be placed within the briefcase. The case would then be locked with a key. The briefcase and its secured contents could be sent to the recipient, but in order to open the case, the recipient would need to possess a copy of the key that had locked it. Diffie and Hellman theorized that asymmetric encryption could be achieved using both a public and private key. Here, the hypothetical briefcase has a special kind of lock and two separate keys. Each key has different abilities. The first key is private. This key can be used to both lock and unlock the briefcase. The second key is public. It can lock the briefcase, but it is not able to unlock it. By sharing the public key, the owner of the briefcase empowers others to send them secure messages that only they can open. With asymmetric encryption, the briefcase could also be locked by the private key. When the private key locks the case, only a copy of the public key can unlock it. Thus, anyone able to open the case can be assured that the contents were sent from the possessor of the private key. In this way, encryption could act as a signature, since only the possessor of the private key could produce a message correctly decrypted by the public key.

[46] Ibid., 654.

Diffie and Hellman's paper quickly piqued the interest of other researchers. At the MIT, three researchers, Ron Rivest, Adi Shamir, and Leonard Adleman, took the system proposed by Diffie-Hellman as a challenge, and they began a hunt for an algorithm that could securely achieve the one-way function theorized in the paper.[47]

In April 1977, the MIT researchers hit on a solution. Ron Rivest proposed that they have a computer generate two large prime numbers, each 100 digits long. These would then be multiplied together and serve as the public key. The computer would then choose another large number, one that met "certain easy-to-calculate specified properties." This would serve as the encryption key. Rivest created a formula that would take the keys and create encrypted ciphertext. The solution, based on factoring the product of the two random primes, was grounded in what previously had been theoretical mathematics – it employed an algorithm developed by Greek mathematician Euclid to create the decryption key and the schema itself was derived from eighteenth-century Swiss mathematician Leonhard Euler. Adi Shamir and Leonard Adleman confirmed Rivest's hypothesis and agreed that his novel use of the mathematical theories allowed for the creation of a public key encryption system that met the requirements laid out by Diffie and Hellman's proposal. The trio documented their work on April 4, 1977 in MIT/Laboratory for Computer Sciences technical Memo Number 82, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems."[48] Despite Rivest's outsized role in the solution, he insisted that the other researchers also be included as inventors. They named their system after the order in which their names appeared on the article: Rivest, Shamir, Adleman – RSA.

---

[47] Levy, *Crypto*, 94-95.
[48] Ibid., 99-101.

The researchers shared the document with others in their field, including Diffie and Hellman. Rivest also sent a copy to Martin Gardner, a *Scientific American* journalist responsible for the publication's reoccurring "Mathematical Recreations" column. Gardner met with Rivest and composed an article for the magazines' August 1977 issue. In the piece, Gardner detailed the innovation and challenged readers to reverse a sentence encrypted by a 129-digit RSA key.

In December 1977, MIT filed a patent for the RSA cryptosystem. With the filing, Ron Rivest, Adi Shamir, and Leonard Adleman realized that, as the inventors of the system, they would be uniquely positioned to exploit their discovery. The researchers began to develop ways in which they might commercialize their innovation. They developed a plan to manufacture and sell semiconductor chips encoded with the RSA algorithm. In 1982, they incorporated RSA Security Inc. to sell the chips. In September 1983, MIT was granted a U.S. Patent for the RSA system. With the help of outside investors, RSA Security Inc., secured the rights to use the patents nine days later. For $150,000 plus 5% of all future revenues, MIT granted them exclusive rights to the patent. The organization soon determined that their plan to create RSA chips was too complicated, and they opted instead to develop Mailsafe, a public key cryptosystem that would encrypt email and secure data stored on business computers.[49] Strong, Public Key-based encryption was now available for any U.S. corporation that could afford it.

The Government Strikes Back

Governments had long enjoyed a monopoly on strong encryption, and that of the United States was not interested in permitting the dissemination of these technologies. As

[49] Ibid., 104-105, 130-137.

exemplified by the Colossus, the allied governments had made significant advances in computerized cryptography by the end of World War II. Following the conflict, they continued secret research in the field while endeavoring to prevent the technology from spreading. Starting in the late 1960s, the British GCHQ group responsible for the Colossus began research on public keys and asymmetric encryption. In 1969 GCHQ researcher James Ellis proposed using private and public keys connected by a one-way mathematical function. Three years later, GCHQ mathematician Clifford Cocks, discovered the asymmetric cipher later patented by Rivest, Shamir, and Adleman. Cocks' friend, GCHQ cryptographer Malcolm Williamson developed the exact key exchange schema that was later conceived by Diffie-Hellman. However, due to the secretive nature of the GCHQ, all these initial discoveries were kept secret until 1997.[50]

In addition to embargoing their research on encryption, governments worked to prevent the spread of the technologies. In the United States, this meant that cryptography was governed by the International Traffic in Arms Regulations (ITAR), U.S. regulations enacted in 1976 with the goal of controlling the export of defense technologies. The problem was that with increasing computerization of industries such as banking, cryptographic functionality was needed by business.

To address this need, the U.S. government sought to introduce an encryption standard that it was comfortable with the public using. In 1973, the National Bureau of Standards (NBS) attempted to create a public encryption standard that could be used to secure financial transactions and other business activities. They issued a request for proposals but initially received no responses. A year later, IBM proposed the use of an

---

[50] Singh, 282-290.

internally developed 128-bit key based symmetric encryption system that they named Lucifer.[51] After the corporation contacted the NBS about their algorithm, the National Security Agency (NSA) suddenly became interested in the technology. The NSA classified aspects of how the algorithm functioned, and they persuaded IBM to reduce the key size to a paltry 56-bits. The NBS adopted a specification based on IBM's work, an encryption schema that they entitled the Data Encryption Standard (DES). The government's hobbling of the key size and the influence of the NSA were condemned by many cryptanalysts including Whitefield Diffie and Martin Hellman. These critics believed that the NSA's influence over DES "ensured that the proposed standard is breakable by the NSA."[52]

The NSA also endeavored to restrict cryptographic research. By 1975, the agency was pressing the National Science Foundation to withhold funding from researchers working with cryptography. The government also tried to prevent the dissemination of cryptographic findings by suggesting that the research and anyone who published it would be violating the ITAR regulations. By the start of 1978, the agency's approach was to immediately classify any cryptographic work they encountered through publication or patent applications; as a result, cryptographers would be unable to share their work due to its newfound secret status. The NSA's crusade against private cryptography was eventually curtailed by the very regulations the organization used to justify its actions – ITAR. In the spring of 1978, the agency's controversial actions against public cryptography came to the attention of White House science advisor, Frank Press. Press requested the Justice Department review the ITAR and the role the directives might play

[51] Garfinkle, 64-67.
[52] Levy, *Crypto*, 60-61.

within cryptographic research. After researching the matter, the Office of the General Counsel released its opinion on May 11, 1978. It determined that the provisions of ITAR were unconstitutional in that they "establish a prior restraint on disclosure of cryptographic ideas and information developed by scientists and mathematicians in the private sector."[53] Using ITAR to prohibit the dissemination of cryptographic work was a violation of the First Amendment.

The Office of the General Counsel decision helped curtail the NSA's campaign against private cryptography in time for companies like RSA Security Inc. to build an industry in the 1980s. However, the U.S. government was not willing to give up control of cryptographic communication that easily. By the 1990s the government was arguing for various schemes designed to create a *key escrow*, where all users of cryptographic software would be legally required to provide a copy of their private encryption key. The key would be stored by a third-party organization which could share it with the government if requested.[54] On January 24, 1991 Senate Judiciary Committee Chairman Joseph Biden introduced S.266, the Comprehensive Counter-Terrorism Act of 1991. In section 2201 of this omnibus anti-crime bill, there was a clause requiring government access to secure telecommunications systems.

> SEC. 2201. COOPERATION OF TELECOMMUNICATIONS PROVIDERS WITH LAW ENFORCEMENT.
> It is the sense of Congress that providers of electronic communications services and manufacturers of electronic communications service equipment shall ensure that communications systems permit the government to obtain the plain text contents of voice, data, and other communications when appropriately authorized by law.[55]

[53] Ibid., 106-109,115-119.
[54] Singh, 310.
[55] Comprehensive Counter-Terrorism Act of 1991, S. S.266, 102d Cong., 1st sess. (January 24, 1991): https://www.congress.gov/bill/102nd-congress/senate-bill/266/text,(accessed January 21, 2020).

By June of 1991, corporations such as RSA Security and activist groups like the Electronic Frontier Foundations were pushing back on the language within the bill. The backers of the bill tried to reassure critics that S.266 wasn't intended as a ban on cryptography, but simply a way to make it easier for law enforcement to tap cellular telephone calls. Section 2201 was eventually removed from the bill, but despite assurances to the contrary, it seemed clear that the government intended to control private cryptography. For civil libertarians like those in the Free Software community, the law reified concerns with governmental overreach and bred mistrust in the nation's institutions. For one such programmer, Phil Zimmerman, S.266 served as a catalyst to write and release Free Software that would take strong encryption out of the government's control entirely.

Pretty Good Privacy

In 1981, free-lance programmer Phil Zimmerman co-founded a computing company in Boulder, Colorado. The organization, Metamorphic Systems, aimed to create single board computers based on the Intel 8088 chip that powered the IBM PC. The boards could be added to an Apple II computer and would give the machine the ability to run PC programs as well as additional processing power. In 1983, Zimmerman received a call from Charlie Merritt, a programmer who was interested in the Metamorphic add-on board for a potential RSA encryption implementation project. Although the project never came to be, the call resulted in the programmers forming a friendship over their common interest in cryptographic technologies. The discourse encouraged Zimmerman to pursue several projects – an encrypted terminal, a low-cost encrypted telephone, and an encryption suite for the personal computer.[56]

By 1986, Merritt was doing programming projects for RSA Security, Inc. During a trip to meet with Zimmerman in Boulder Colorado, Merritt arranged for a dinner with RSA Security's President Jim Bidzos. In the course of the meal, Bidzos learned of Zimmerman's cryptography work and offered him a job working on an RSA contract for the U.S. Navy.[57] Zimmerman demurred; he was an anti-war activist who had previously been arrested at a protest alongside Carl Sagan and Daniel Ellsberg – he could hardly work in the service of the military.[58] Zimmerman instead focused on his intention to write computer programs focused on the RSA encryption format. Bidzos gave Zimmerman a free copy of RSA's MailSafe product, and according to Zimmerman, Bidzos also offered a free RSA license to use for the software he was working on.[59] Phil Zimmerman left the meeting enthusiastic to work on his planned software.

To compose the programs that he intended to create, Zimmerman had to overcome one critical issue – most personal computers lacked the processing power to handle the mathematical functions needed for asymmetric encryption like RSA. Symmetric encryption was much less processor intensive, but it had limited utility, as it required a direct exchange of encryption keys. After several years of working on the project, Zimmerman found a novel solution; his program would only use the RSA asymmetric encryption briefly – to securely exchange encryption keys. Then, using the key obtained through RSA, the software would handle the rest of the cryptographic work

[56] Garfinkle,86-96.
[57] Ibid., 91.
[58] Singh, 295.
[59] Levy, *Crypto*, 193.

through the less intensive symmetric encryption approach. The approach solved the performance issue but created a new problem. Now Zimmerman needed a second algorithm for symmetric encryption. The programmer tried the government standard DES encryption, but he found it to be insecure. He opted instead to write his own, which he entitled "Bass-o-Matic," a reference to a skit the programmer had seen on Saturday Night Live.[60]

Using his own modified version of the RSA cryptosystem, Zimmerman began to finalize his planned programs and by mid-1990 the products were nearing completion.[61] Zimmerman wanted approval to continue with the RSA algorithm, so he sent a letter to RSA President Jim Bidzos.

> Well, I'm finally getting around to writing you about requesting a royalty-free license for your RSA algorithm. We talked about this a few times since your 1986 visit to Boulder, when I developed my own RSA math library in C. Both you and Ron said then and later that you would grant me a free license to make and sell products with your algorithm. I appreciate that a lot. When we last spoke, you said you would need a letter telling you what products it's for. It was unclear whether this meant highly detailed firm product plans or just general fuzzy plans.[62]

In the letter, Zimmerman went on to describe two planned projects, the first was a "low-cost secure telephone, based on encryption and voice compression," and the second was a program to perform RSA encryption on a personal computer, "This would be somewhat analogous to your MailSafe or ComSafe products," Zimmerman wrote.[63] Bidzos' response to Zimmerman's letter was terse. He denied ever promising a free

---

[60] Singh, 298-300
[61] Levy, *Crypto*, 195.
[62] Garfinkle, 96-97.
[63] Ibid.

license, and instead he suggested that Zimmerman should look to collaborate with an established company that had the funds to purchase a standard RSA license.[64]

In April 1991, Phil Zimmerman learned of S.266, Senator Biden's proposed omnibus anti-crime bill that would effectively criminalize any domestic cryptography that did not afford the government free access to encrypted messages. The programmer interpreted the potential passing of the law as the ultimate deadline: if Zimmerman could finish his PC encryption program quickly, it could be broadly shared before the law went into effect.[65] By June 1991, Zimmerman completed the software and released it under the Free Software Foundation's GNU Public License. He named his application Pretty Good Privacy (PGP), a reference to a fictional sponsor of Garrison Keillor's popular radio program, the *Prairie Home Companion*.[66] The software could be used to encrypt emails or files on a personal computer. Zimmerman shared the Free Software with associates who subsequently posted it to the world-wide dial-up network, Usenet.[67] According to journalist Steven Levy, this dissemination was intended; Zimmerman allegedly shared it with friend Kelly Goen, who coordinated with former Dr. Dobb's editor Jim Warren to upload the package to various computer networks.[68] Despite his indirect role in the distribution of PGP, it was clear that Zimmerman completed and released the program for political reasons. Where the Free Software that had come before was intended to create a sphere outside of corporate control, Zimmerman's release was a

[64] Levy, *Crypto*, 195.
[65] Ibid., 196.
[66] Singh, 302,298.
[67] Garfinkle, 100.
[68] Levy, *Crypto*, 197.

direct challenge to the government and an attempt to create a cryptosystem free from governmental oversight.

Predictably, the release and distribution of Pretty Good Privacy brought Zimmerman an abundance of unwanted attention. RSA Security, Inc. immediately began to pursue Zimmerman for patent infringement.[69] By February 1993, Zimmerman was also under investigation by the FBI and was the subject of a Grand Jury enquiry. Where RSA accused Zimmerman of stealing corporate technology, the U.S. government was accusing the programmer of being an arms dealer. Encryption was still treated as a munition and, as such, was subject to export controls. After PGP was uploaded to the Usenet, it quickly found its way abroad.[70] These legal problems beleaguered the programmer for several years, but ultimately had no long-term impact. Zimmerman was quick to remove the problematic RSA algorithm: by PGP version 2.6, both the symmetric and asymmetric encryption processes had been completely redone.[71] By 1996, the U.S. Attorney General's office dropped its case against Zimmerman. As the programmer had only given the software to a friend and did not himself distribute the program internationally, the government was unable to establish that Zimmerman had violated any export control laws. By this time, the MIT Press had also published a 600-page book on PGP, complete with the source code for the Free Software.[72] With the text, the program was distributed world-wide and any attempt to cease publication would be a clear abrogation of the First Amendment. Code is speech.

[69] Ibid., 199.
[70] Singh, 303.
[71] Garfinkle, 102-109.
[72] Singh, 314-315.

The Cypherpunks

In writing and sharing Pretty Good Privacy, Phil Zimmerman made strong encryption accessible to any computer user. In the process, he demonstrated how Free Software could be used to change society itself: Before the release of PGP, a governmental restriction like Senator Joseph Biden's S.266 may have worked. After the software was released, however, it would be impossible to contain. Even if it were made illegal, the technology would be out there, usable, and the government would be able to do extraordinarily little about it. In addition to his promethean role in bringing encryption technology to the public, Zimmerman had proven that Free Software could be used to create a sphere free from government oversite or control. His actions also inspired others to do the same. Based on Zimmerman's work, a group calling themselves the Cypherpunks sought to use Free Software and digital encryption to change the nature of society itself.

John Gilmore

The Cypherpunks began with John Gilmore. In early 1982, he met Sun Microsystems co-founder Andy Bechtolsheim, who convinced the programmer to become the fifth employee at then-fledgling computing company.[73] While at Sun, he began work with Stallman's GNU project. In the years that followed, Gilmore played a critical role in the Free Software Movement: The programmer was originally hesitant to release software under Stallman's license, but soon became an adherent to the cause.[74] In addition to being a prolific contributor of GNU software, submitting packages such as

[73] *The Internet: A Historical Encyclopedia* (Santa Barbara, California: ABC-CLIO, 2005), 115.
[74] John Gilmore, "Re: gnu copyrights." Email to net.emacs mailing list. May 21, 1986.

TAR (Tape Archive and Recovery), Gilmore also prompted Stallman to generalize his GNU Emacs license so that it might be applied broadly to other software packages. The result of this suggestion was the GNU Public License (GPL).[75] When Sun Microsystems went public in 1986, Gilmore became a millionaire. In 1989, the programmer used a portion of his fortune to co-found Cygnus Support (later renamed to Cygnus Solutions), offering commercial support services for the GNU project's Free Software.[76] The following year, John Gilmore teamed up with LOTUS founder Mitch Kapor and former Grateful Dead Lyricist John Perry Barlow to form the Electronic Frontier Foundation (EFF). In addition to the EFF's advocacy activities, the group helped to defend countless programmers against DMCA-based charges and was involved in many other cases defending various causes from free speech to online privacy.[77]

Gilmore's early involvement with the GNU project and later the Free Software Movement had shown him how programs could be used to protect the rights of the individual from corporations. Now, actions like Senator Joseph Biden's proposal to monitor private encrypted communications showed that government intended to be even more intrusive than the corporations had. Unlike corporations, the government knew no limits; it had shown this with Operation Sun Devil, the highly publicized May 1990 raids where agents seized and kept 40 computers and 23,000 discs from citizens who were never charged with a crime.[78] The world was progressively becoming dependent on

[75] Sam Williams, *Free as in Freedom: Richard Stallman's Crusade for Free Software* (Sebastopol, California: O'Reilly, 2002), 125.

[76] *The Internet: A Historical Encyclopedia*, 115-117.

[77] Turner,172.

[78] Dan Charles, "'Innocent' Hackers Want Their Computers Back," *New Scientist*, May 9, 1992.

computers, and yet the government's actions increasingly suggested that it had little regard for the rights of the individual in this new digital world.

In March 1991, Gilmore presented at the First Conference on Computers, Freedom & Privacy in Burlingame, California. The conference, chaired by former Dr. Dobb's editor Jim Warren, discussed the increasingly complex challenges of preserving individual freedoms in the information age. Gilmore's presentation, "Privacy, Technology, and the Open Society," gives insight into his worldview. Gilmore's talk focused on "the belief in an open society and the belief in privacy," two ideas he held to be ethical precepts.[79] In the discussion, he cited the need to deeply distrust the government, arguing that since the people running the government change, one must "assume that bad people -- criminals even -- will run the government, at least part of the time." The challenge with this being that if the government is permitted to monitor and collect information on the citizenry, even for innocuous purposes, it still has the potential to be used to malicious ends. Gilmore relays two historical examples of this: the Nazi use of phone records to track and identify opposition in World War II and the United States use of census data to round up Japanese Americans for internment in concentration camps. Gilmore proposed his vision for a different world:

> What if we could build a society where the information was never collected? Where you could pay to rent a video without leaving a credit card number or a bank number? Where you could prove you're certified to drive without ever giving your name? Where you could send and receive messages without revealing your physical location, like an electronic post office box? We also want real privacy of personal records…real freedom of trade…real financial privacy…. We also need real control of identification. We need the option to be anonymous while exercising all of these other rights….

[79] John Gilmore, "Privacy, Technology, and the Open Society" (lecture, First Conference on Computers, Freedom & Privacy, Burlingame, California, March 26-28, 1991), http://cpsr.org/prevsite/conferences/cfp91/gilmore.html/,(accessed August 12, 2019).

In his speech, John Gilmore suggested that his utopia was tenable, if the existing technologies were utilized correctly and expanded to provide the privacy he envisioned. "That's the kind of society I want to build," John Gilmore explained, would be created, "with physics and mathematics, not with laws…Encryption strong enough that even the NSA can't break it. We already know how. But we're not applying it."[80]

Tim May and Eric Hughes

Like John Gilmore, Timothy May made his fortune in Silicon Valley. The hardware engineer joined the Intel corporation in 1974 while the small microchip company was just beginning to expand. By the early 1980s the company enjoyed a near monopoly in providing microprocessors to IBM and its competitors in the personal computer market. In 1986, Timothy May exercised his stock options and retired at age thirty-four. Disillusioned with business, May withdrew to his home in Aptos, California and pursued a new life focused on intellectual exploration. May soon became active on Usenet, and in this new digital world, he discovered the promise of unbreakable cryptography. The retiree began to perceive the technology as empowering, a tool with which society itself could be restructured to protect the individual, shifting power away from corporations and the government. May began to research academic cryptography, and by 1988, he decided to focus his post-corporate life on writing a science fiction novel about a society that had fully realized the potential of private encryption. He began writing the text, tentatively entitled *Degrees of Freedom*, but after three years working on

[80] Ibid.

the effort, May decided that instead of writing about his techno-libertarian ideal, he wanted to make it reality.[81]

Around the same time that May was having his epiphany about societal change, he met programmer Eric Hughes at a party thrown by John Gilmore.[82] Following the party, Hughes worked briefly as a programmer for a cryptography start-up in Amsterdam. The job was brief, and when Hughes returned to the Bay area to apply to graduate school, he reconnected with May. Hughes's employer in the Netherlands, DigiCash, was founded by David Chaum, an American cryptographer.[83] Eric Hughes's time as Chaum's employee showed him that they were incompatible on a personal level, but he was greatly influenced by Chaum's ideas. Chaum's research had also informed Timothy May's techno-libertarian writings, and upon Hughes' return, the two men spent a great deal of time together exchanging ideas.[84]

Working with John Gilmore, Timothy May and Eric Hughes conceptualized a movement to change society through cryptography. They initially styled their group the Cryptology Amateurs for Social Irresponsibility (CASI). The invite-only group of 20 individuals met for the first time on September 19, 1992 at Eric Hughes's home in Berkeley. At the gathering they distributed copies of Phil Zimmerman's PGP v.2.0, released a week prior, and the assembly exchanged PGP keys. The group discussed the

---

[81] Andy Greenberg, *This Machine Kills Secrets: How Wikileakers, Cypherpunks and Hacktivists Aim to Free the World's Information* (New York, New York: Dutton, 2012), 49-52, 58-64, 70-78..
[82] Ibid. 78.
[83] David Chaum, "Blind Signatures for Untraceable Payments," *Advances in Cryptology Proceedings* 82, no. 3 (1983), 199-203.
[84] Greenberg, 78-79.

societal implications of cryptography, and Tim May presented a document he composed, the *Crypto Anarchists Manifesto*.[85]

The Crypto Anarchists Manifesto

In the *Crypto Anarchists Manifesto*, May stated that technology advancements, such as public-key encryption, had brought society to the advent of a "social and economic revolution" in which individuals will be able to anonymously communicate, conduct business, and negotiate contracts. May speculated that these abilities will alter the power structures of society itself and reduce the roles of corporations and the government in economic transactions. He argued that because of these power shifts, the state itself will attempt to halt cryptographic technologies, "citing national security concerns, use of the technology by drug dealers and tax evaders, and fears of societal disintegration." May predicts that these concerns may turn out to be valid, as the crypto anarchy encouraged by the document would "allow national secrets to be traded freely and will allow illicit and stolen materials to be traded." But the use of the technology by "criminal and foreign elements" would not stop the spread of crypto anarchy, as the changes would themselves would alter society's conception of intellectual property rights, just like "barbed wire made possible the fencing-off of vast ranches and farms, thus altering forever the concepts of land and property rights in the frontier West," these technology changes will be liberating, the "wire clippers which dismantle the barbed wire

[85] Timothy May. "The Crypto Anarchist Manifesto". Email to cypherpunks@toad.com Mailing List. November 22, 1992.

around intellectual property." May's document concluded with allusion to Marx and Engels, "Arise, you have nothing to lose but your barbed wire fences!"[86]

Despite the reference, Timothy May's document had more in common with Gilmore's "Privacy, Technology, and the Open Society" presentation than it did the *Communist Manifesto*. Both May and Gilmore offered visions of a future where technology allowed them to create a sphere apart from corporate or government control, but neither specified how society would get there. This was to come from Eric Hughes.

A Cypherpunks Manifesto

By the time of the September 1992 meeting, Eric Hughes was in a relationship with Judith Milhon, a hacker known as "St. Jude," who was renowned for her role in creating the first public Bulletin Board Systems (BBS) in the 1970s. At the time an editor of the counter-culture technology magazine *Mondo 2000*, Milhon linked the calls for societal change to crypto anarchy with the recent media interest in "cyberpunks." The cyberpunks were young hackers who connected "punk-rock rebels with the digital revolution," stylized in the works of science fiction authors such as Bruce Bethke, William Gibson, and Bruce Sterling. She offered her assessment to the others at the meeting, "you guys are *Cypher*punks!" The group liked the name, and following the initial meeting, Gilmore set up the Cypherpunks mailing list on his domain, toad.com.[87] The subsequent physical meetings were held at the headquarters of Gilmore's Free

[86] Ibid.
[87] Levy, *Hackers*, 283, 213.

Software support company, Cygnus Solutions, in Mountain View, California, but the digital center was cypherpunks@toad.com.[88]

On October 5, 1992, Hughes sent the list a draft "statement of purpose." This document sought to define the group and their objectives. By March of 1993, the document had evolved into *A Cypherpunks Manifesto*. In the document Hughes declared both privacy and the freedom of speech to be requirements to an open society in the electronic age. Fundamental to both was the ability of the individual to govern how much personal information is garnered in a situation. Hughes employed the example of purchasing a magazine, "When I purchase a magazine at a store and hand cash to the clerk, there is no need to know who I am…. When my identity is revealed by the underlying mechanism of the transaction, I have no privacy. I cannot here selectively reveal myself; I must *always* reveal myself."[89] Based on this, Hughes concluded that anonymous transactions are required in order to truly have privacy in an open society, as it "empowers individuals to reveal their identity when desired and only when desired."[90]

"We cannot expect governments, corporations, or other large, faceless organizations to grant us privacy out of their beneficence," Hughes warned. "We must defend our own privacy if we expect to have any." This, according to Hughes, was the role of Cypherpunks, who are dedicated to protecting privacy using cryptography to build anonymous systems, digital signatures, and electronic money. To fulfill this commitment, Hughes explained that the Cypherpunks would compose and share Free Software.

---

[88] Eric Hughes.,"Meeting Sat. Oct. 10, noon, Mt. View," Email to cypherpunks@toad.com Mailing List. October 5, 1992.

[89] Eric Hughes, "A Cypherpunk's Manifesto," Activism.net, March 9, 1993, https://www.activism.net/cypherpunk/manifesto.html, (accessed August 12, 2019).

[90] Eric Hughes. "A statement of purpose". Email to cypherpunks@toad.com Mailing List. October 5, 1992.

> Cypherpunks write code. We know that someone has to write software to defend privacy, and since we can't get privacy unless we all do, we're going to write it. We publish our code so that our fellow Cypherpunks may practice and play with it. Our code is free for all to use, worldwide.[91]

Just as the GNU project and the Free Software Movement, the Cypherpunks saw Free Software was a way to ensure their freedoms. Where Stallman's focus centered on the freedom of expression, the Cypherpunks deliberately extended their goals to individual privacy. Where mistrust of corporations informed Stallman's crusade, the Cypherpunks extended their distrust to the government as well. Where Richard Stallman wrote and distributed Free Software to create a sphere outside of corporate influence, the Cypherpunks proposed to create software that created a sphere outside of ALL control. Not only did the Cypherpunks adopt and extend the ideologies behind Free Software, but they adopted the schema as well. Hughes's manifesto explained that they would create and distribute software free, both libre and gratis, complete with source code, for the world to use.

Cryptography was the key element they were adding to the Free Software Movement, both ideologically and schematically. *A Cypherpunks Manifesto* made clear that they intended to pursue their goal globally, regardless of any government's concerns. Laws are constrained, Hughes argued, "they reach only so far as a nation's border and the arm of its violence." Cryptography, in contrast, was limitless. It would, Hughes predicted, inevitably "spread over the whole globe," and with it, the Cypherpunk systems that it made possible.[92]

---

[91] Ibid.
[92] Ibid.

The Clipper Chip

As both Tim May and Eric Hughes had predicted in their manifestos, the U.S. government was vehemently opposed to the spread of heavy encryption. In 1993, the U.S. government introduced the Clipper Chip, a microchip-based encryption system developed by the National Security Agency (NSA). The device was designed for adoption by telecommunications providers as a method to encrypt voice and data transmissions. The chip used Skipjack, an 80-bit encryption cipher designed by the NSA, but there was a catch. Each chip had a unique key, and each encrypted transmission included extra data – the Law Enforcement Access Field (LEAF). Using a private master key held in escrow and the LEAF information, the U.S. government could decrypt any information sent using the system. The government argued that it was secure; it would only remove keys from escrow for legitimate purposes. According to President Clinton, the nation needed the "Clipper Chip and other approaches that can both provide law-abiding citizens with access to the encryption they need and prevent criminals from using it to hide their illegal activities." "The war is upon us," Cypherpunk founder Tim May wrote of the plan. For the Cypherpunks, the scheme reified their concerns. The Clinton administration had "shown themselves to be enthusiastic supporters of Big Brother."[93] The plan also proved that the S.266 bill was not as innocuous as had been previously suggested by its backers.

The numerous issues with the Clipper Chip were apparent. Who would hold the keys in escrow? How would they secure the holdings to keep a malicious cracker from getting the keys and being able to decrypt any and all communications? What

---

[93] Levy, *Crypto*, 232-235,249-254.

constituted a legitimate purpose? Since the government was clear that it would be able to decrypt all communications, why would a criminal choose to use a clipper chip device for criminal activities? And if criminals would avoid the system, what was the point of doing it at all? Protests against the program came far and wide. Naturally, cryptographers like Whitfield Diffie were opposed, but joining the protest were groups like the American Civil Liberties Union (ACLU) and right-wing malcontent Rush Limbaugh. By early 1994, a *New York Times*/CCN poll revealed that 80 percent of the American public opposed the plan. That same year, AT&T researcher Matt Blaze published research showing a fatal flaw in the Clipper Chip; software could be used to alter the Law Enforcement Access Field (LEAF) information, eliminating the government's ability to decrypt the traffic.[94] So, on top of the public pushback on the device, it could not even be relied on to perform its intended function!

Despite the Clipper Chip failure, the U.S. government still made considerable progress on its anti-cryptography objectives. On October 25, 1994, Congress enacted the Communications Assistance for Law Enforcement Act (CALEA). The stated purpose of the law was to "make clear a telecommunications carrier's duty to cooperate in the interception of communications for law enforcement purposes, and for other purposes." The legislation stipulated that all telephony and computer network providers, needed to provide functionality that allowed the U.S. government to tap any communications occurring on their networks on demand. The telecommunications carriers were obliged by law to make available to the U.S. government the "metadata" about communications. For example, a telephony vendor needed to provide data about what calls were placed,

[94] Garfinkle, 97-98, 252-256, 161.

when, and by whom. In addition, if a digital vendor provided its customers with any form of encryption, the vendor needed to be able to decrypt the customer's traffic and surrender it to the government upon demand. Cellular network providers were required to provide metadata tracking callers through their systems even if the callers in question were not their clients. Where the Clipper Chip failed to give the government access to the devices used for secure communications, the CALEA side-stepped the issue by giving the government access to the communications themselves.[95]

The BlackNet Experiment

While the U.S. government was marketing the Clipper Chip and laying the groundwork for mass surveillance with CALEA, the Cypherpunks were imagining the anarchy made possible with the use of private cryptography. A prescient indication of what was to come was sent to the Cypherpunks mailing list on August 17, 1993 in the form of an anonymous email introducing the "BlackNet," an anonymous, Internet-based business focused on the "buying, selling, trading, and otherwise dealing with *information* in all its many forms." The group espoused a worldview similar to those described in the various Cypherpunk manifestos.

> BlackNet is nominally nondideological[SIC], but considers nation-states, export laws, patent laws, national security considerations and the like to be relics of the pre-cyberspace era…. BlackNet believes it is solely the responsibility of a secret holder to keep that secret--not the responsibilty [SIC] of the State, or of us, or of anyone else who may come into possession of that secret. If a secret's worth having, it's worth protecting.[96]

[95] Communications Assistance for Law Enforcement Act, H. Res. 4922, 103d Cong., 1st sess. (October 7, 1994): 1-4.

[96] Anonymous <nowhere@bsu-cs.bsu.edu>. "no subject (file transmission)". Email to cypherpunks@toad.com Mailing List. August 8, 1993.

The email solicited the purchase of information, “We buy and sell information using public key cryptosystems,” the message explained, “Unless you tell us who you are (please don't!) or inadvertently reveal information which provides clues, we have no way of identifying you, nor you us.” The correspondence went on to explain that BlackNet was building an inventory of information, including trade secrets, production methods, weapon plans, business intelligence and manufacturing data. The BlackNet organization offered payment through anonymous bank deposits or “CryptoCredits,” the “internal currency of BlackNet.”

On October 1, 1993, several weeks after the email was received, Cypherpunk founder Tim May took credit for the message, sending an email explaining that BlackNet was a thought experiment, a “plausible implication of current trends,” a warning that in the decades to come that the collection of information about individuals and organizations will be common businesses. “Entities like BlackNet will flourish in the crypto-anarchic world of cyberspace, May explained. “Wake up and prepare for this future. Embrace it or be left behind.” The author warned, “Crypto-anarchy is inevitable, and changes everything.” [97]

By the time of Tim May’s October email, the original BlackNet solicitation had spread throughout the Usenet, eventually coming to the attention of government facilities, such as the Oak Ridge National Laboratories, which issued warnings requesting that staff immediate report any contacts from the Black Net organization. The experiment also proved the tenability of such a group. May reported that in the weeks following the original email, one recipient followed the anonymous reporting process outlined in the

[97] Timothy May, "BlackNet Investigations--the Truth (fwd)". Email to cypherpunks@toad.com Mailing List. October 1, 1993.

email and offered evidence that the CIA was actively spying on ambassadors from a Central African country, and further that the evidence exposed the corruption within the African country's government. According to May, after unencrypting and reading the message, he archived it and never responded.[98] May had correctly assessed the state of society, and in the years that followed, the endeavors of his fellow Cypherpunks were to have a vital role in making his anarchist vision a reality.

Encoding Anarchy

Tim May's musing on Crypto anarchy would have been of little consequence if it were not for his fellow Cypherpunks who actively worked to create the world described by May. In the years to come, list members produced the Internet-based information markets May imagined through sites like Cryptome.org, and later, Wikileaks. Others wrote encryption-focused Free Software that permitted the creation of secret networks and anonymous currencies.

Cryptome

In 1996, Cypherpunks John Young and Deborah Natsios created the website Cryptome.org. Young and Natsios were principals of a New York-based architecture firm and were enthusiastic participants on the Cypherpunk mailing list. In the first eight years of the mailing list, Young sent over 1600 messages to the group, surpassing even the Cypherpunk founders.[99] According to Natsios, the Cypherpunks were "dealing with issues of cryptography and freedom of information, and was way more advanced than

---

[98] Greenberg, 90-91.

[99] "Cypherpunk List Authors by Number of Posts (Highest First)," cryptoanarchy.wiki, https://mailing-list-archive.cryptoanarchy.wiki/authors/by-posts/, (accessed August 12, 2019).

anything that architectural practice was interested in at the time."[100] Consistent with their interest in the freedom of information, the couple created Cryptome to share information with the public. Among the first documents on the site were government documents: Airforce Issue Papers, reports on the Clipper Chip program, and Whitehouse Office of Management and Budget Memos. The site also featured links to magazine articles about cryptographic issues.[101] The website soon expanded to publishing materials that no one else was willing to publish, including copyrighted material, sensitive national security documents, and lists of CIA assets. [102]

The Cypherpunk architects described their website as a public education project. The neither vet nor curate the information they post. When the website published the Association for Intelligence Officers membership list in 2000, the information included the name and address of Natsios' father, Nicholas Natsios, a career CIA officer. No one was exempt from Cryotome's exposure. At the same time, John Young made it clear that material published by Cryptome should not be regarded as authentic. "Don't believe anything we publish," Young warned. "We're totally untrustworthy. We may be a sting operation, we may be working for the Feds. If you trust us, you're stupid."[103] The warning encouraged the cynicism and paranoia that the architect himself espoused. In the years of sharing government secrets, including exposing 2,519 CIA sources, 400

---

[100] Joseph Grima, "Open Source Design 01: The Architects of Information," *Domus*, June 18, 2011, https://www.domusweb.it/en/interviews/2011/06/18/open-source-design-01-the-architects-of-information.html.

[101] "Cryptomb 1," Cryptome, https://cryptome.org/jya/cryptomb1.htm, (accessed August 12, 2019).

[102] Adrian Chen, "A Discussion with Cryptome," Gawker, June 19, 2013, https://gawker.com/a-discussion-with-cryptome-514154708, (accessed August 12, 2019).

[103] John Cook, "Secrets and Lies," *Radar Magazine*, August 13, 2007, http://radaronline.com/from-the-magazine/2007/08/cryptome_john_young_radar_anthony_haden_guest_1.php, (accessed August 12, 2019).

Japanese intelligence officers, and 276 MI6 agents, Young realized the potential of his website being used by intelligence agencies to disseminate misinformation.[104] Counterintelligence groups could also use the source disclosures for what Young referred to as "Wargaming," where the "means of digital disclosure can be quietly tracked from start to finish, through observation, recording and analysis…" as a means of discovering whistleblowers and their methods of sharing information.[105] Cryptome fulfilled part of Tim May's BlackNet vision, but the website's Cypherpunk founders realized that their creation was open to manipulation by the same geopolitical powers that their movement was acting against.

DarkNets

By the early 2000s this aspect of Tim May's BlackNet vision had begun to coalesce. Widespread encryption, a key component to the world he imaged was achieved through the creation of encrypted virtual private networks on the Internet. On these "DarkNets," individuals could use the Internet in a secure and anonymous manner, through the use of peer-to-peer encryption. The first of these networks appeared in March 2000, when Irish programmer Ian Clarke released a Free Software project, FreeNet. The software created an encrypted, private network connecting all the individuals using the program. It created anonymity by distributing the network traffic among the active users. In the case of files posted through the service, it furthered the anonymity through an "immune system that responds to any effort to determine the location of a piece of information by spreading the information elsewhere in the network." Clarke's ambition

---

[104] Ibid.

[105] John Young, "Wargaming Disclosures," Cryptome, June 11, 2013, http://cryptome.org/2013/06/wargaming-disclosures.htm, (accessed August 12, 2019).

with the software was decidedly anti-copyright. His program was focused on realizing the perspective, "expressed by many in the free-software movement…that copyright protection is simply obsolete in the Internet era."[106]

Within a year, other Free Software alternatives began to emerge, most notably the GNU-based darknet, GNUNet. Where FreeNet was interested in secure, anonymized file sharing, the GNUNet project sought to create a secure private network that resided on, but was completely outside of the control of, the Internet.

> Due to fundamental Internet design choices, Internet traffic can be misdirected, intercepted, censored and manipulated by hostile routers on the network…We believe liberal societies need a network architecture that uses the anti-authoritarian decentralized peer-to-peer paradigm and privacy-preserving cryptographic protocols. The goal of the GNUnet project is to provide a Free Software realization of this ideal. [107]

Just as the GNU project had created a sphere of software free from commercial control, the GNUnet effort created a section of the Internet free from governmental oversite.

In 2004, the Hacktivismo group introduced their own anonymous darknet software, Six/Four. Named in reference to the June 4, 1989 Tiananmen Square massacre, the software worked by creating a network of anonymous proxies. The software saw little adoption outside of its intended constituency, but it influenced the creation of other anonymous network systems, including The Onion Router (TOR.) The TOR software was so named because its messages were structured in layers like an onion. After transmission, subsequent network connections (nodes) would access only the layer

---

[106] John Markoff, "Cyberspace Programmers Confront Copyright Laws," *New York Times*, May 10, 2000, A1, https://www.nytimes.com/2000/05/10/business/cyberspace-programmers-confront-copyright-laws.html, (accessed August 12, 2019).

[107] "About GNUnet," GNUnet.org, https://gnunet.org/en/about.html, (accessed August 12, 2019).

detailing the message's destination, the layer with the origination of the message, having been removed after the first TOR node received it. Developed originally at the US Naval Research Laboratory, the project subsequently received funding from DARPA. Despite the governmental origins of the software, it was adopted by private citizens and activist groups. In 2004, the Electronic Frontier Foundation awarded a grant to the project.[108] In addition to the financial support, the grant helped established the software's bona fides within the Cypherpunk community.

In the decade that followed Tim May's BlackNet thought experiment, his crypto anarchist vision had begun to coalesce. The Cypherpunk website Cryptome shared whatever interesting material its authors came across regardless of sensitivity or secrecy level of the information. Meanwhile, the Internet's denizens increasingly had their choice of dark networks, each of them running on Free Software, and each protecting the anonymity and privacy of user activities. But, as May wrote in his October 1[s], 1993 BlackNet explanation email, he and the other crypto anarchists on the Cypherpunk list perceived their quest in geopolitical terms. To them, strong cryptography was a key "technological tool to demolish governments (including the U.S. government)." To May, crypto anarchy was not just a theoretical possibility, but a stated goal. "If you're scared about using technology to bypass laws and ultimately overthrow national governments," May warned the others on the Cypherpunks mailing list, "well, then this list is probably not for you."[109]

---

[108] Menn, 128-129.

[109] Timothy May, "BlackNet Investigations--the Truth (fwd)." Email to cypherpunks@toad.com Mailing List. October 1, 1993.

WikiLeaks

In late 1995, a user named "Proff" joined the Cypherpunk list. Apart from a screed against computer crime laws, the posts from Proff were decidedly less anarchistic than that of contributors like Tim May. Proff tended to mostly post condescending responses to mailing list discussions. For example, after receiving an invitation for a housewarming party that ended at 10 PM, Proff chaffed, "Thats [SIC] not a party. Thats[SIC] an after-school tupperware[SIC] get-together."[110] In another posting, he chided the other list members as "morons," and questioned their contributions. "Do we really need your amatuer [SIC] political views?"[111]

The pugnacious poster named Proff was Julian Assange, an Australian system administrator in his mid-20s. The belligerent programmer was decidedly anti-establishment because of his previous interactions with the law. In 1987, Assange, then going by the username Mendax, had cracked into several institutions, including Australia's Overseas Telecommunication Commission and Nortel, a Canadian telecommunications company. In October 1991, the Australian Federal Police raided Assange's home and subsequently charged him with 31 counts of computer-related crimes. Assange received a sentence of a $2,000 fine and a $5,000 good behavior bond.[112] Although this punishment was extremely lenient, it helped solidify the young Australian's distrust of government power.

---

[110] Julian Assange, "Re: Housewarming/birthday party." Email to lstewart@prisminc.com,cypherpunks@toad.com Mailing List. December 30, 1995.

[111] Julian Assange.,"Re: Risk v. Charity (was: RE: Workers Paradise. /Political rant." Email to craigw@dg.ce.com.au,cypherpunks@toad.com Mailing List. September 17, 1996.

[112] Greenberg, 104-112.

On October 3, 2006, Julian Assange sent an anonymous email to fellow Cypherpunk John Young, asking for his assistance in setting up an organization in the spirit of Young's own Cryptome site.

> You knew me under another name from cypherpunk days. I am involved in a project that you may have feeling for…. The project is a mass document leaking project that requires someone with backbone to hold the .org domain registration….[113]

John Young, who was no stranger to government pressure, agreed to act as the domain registrant for the project's website, WikiLeaks.org. Assange added Young to WikiLeaks's restricted internal development mailing list. Within months, Young became concerned with the activities being discussed in the emails. In December 2006, Young expressed concerns that the group was not verifying documents it received, "It would be prudent to attempt to verify that the Somali document is not a forgery…. The spooks forge such documents as a matter of regular tradecraft, and leaks of them are frequently through an alleged third party." He warned the new leaking organization that if they were not careful, they would be used by governments or individuals as a tool for distributing misinformation. On January 4, 2007, following WikiLeaks's first leak publication, Young cautioned the burgeoning group of what was to come as governments took interest. The elder architect warned that agents of various governments would try to infiltrate WikiLeaks as supporters, journalists, or board members. He warned them to expect negative responses – but also to expect flattery and attempts at bribery.[114]

Within two days of Young's warning, the list was focused on developing a public relations and funding strategy. According to the press copy being discussed on the list,

---

[113] "WikiLeaks Leak," Cryptome, January 7, 2007, http://cryptome.org/wikileaks/wikileaks-leak.htm, (accessed August 12, 2019).

[114] Ibid.

the group claimed that WikiLeaks "may become the most powerful intelligence agency on earth, an intelligence agency of the people….an Open Source, democratic intelligence agency." WikiLeaks, saw itself as better than traditional government organizations, as the group would be "far more principled," in that it would have "no commercial or national interests at heart; its only interests will be truth and freedom of information." However, despite this avowed lack of commercial interest, the conversation quickly turned to monetizing the site, with the stated goal of raising $5 million by July 2007. Young spoke up again, criticizing the idea, "Announcing a $5 million fund-raising goal by July will kill this effort. It makes WL appear to be a Wall Street scam." He pointed out that the operational costs of running the servers to support the organization were minimal, and as such the $5 million goal was suspect. The response to Young's concerns was terse and dismissive: "Advice noted. We'll polish up our sheers for cutting fleeces golden."

The course of the conversation suggested to John Young that the fledgling organization was not interested in apolitical leaks. Instead, the group intended to use any information they acquired to further personal objectives. It was unconcerned with the prospect of being used by a government, and instead was focused on how WikiLeaks could bring in revenue. Convinced of their venality, Young posted the WikiLeaks mailing list archive to his website, Cryptome.org.[115] Young then sent the group a message calling them out.

> Cryptome is publishing the contents of this list, and how I was induced to serve as US person for registration.

[115] "WikiLeaks Leak," Cryptome, January 7, 2007, http://cryptome.org/wikileaks/wikileaks-leak.htm, (accessed August 15, 2019).

Wikileaks is a fraud:

> Fuck your cute hustle and disinformation campaign against legitimate dissent. Same old shit, working for the enemy. [116]

The immediate response from the list was an attempt to convince Young of WikiLeaks' sincerity. "We're going to fuck them all…," the anonymous response claimed, stating that they expected to have ten thousand documents per day, from sources including India and the Russian Mafia. "We're going to crack the world open and let it flower into something new."[117] Julian Assange then emailed the list directly to dismiss Young's criticism. "No idea what JYA [Young] was saying!" Assange wrote, "Perhaps he feels WL [Wikileaks] is a threat to the central status mechanism in his life? I think he just likes the controversy." The WikiLeaks founder then went on to reinterpret the mailing list leak in a positive light, suggesting that the action was a "great favor," that makes the group look good.[118]

WikiLeaks' initial leak soon led to more high-profile exposures: In August 2007, it provided information detailing the extensive corruption of Kenyan leader Daniel Arap Moi and his family.[119] Two months later, WikiLeaks exposed the U.S. government's practices at the Guantanamo Bay Detention Camp in Cuba.[120] Leaks exposing the Chinese government's heavy handed reaction to Tibetan dissent followed the following

[116] John Young, "Re: [WL] Funding/who is on this list." Email to wikileaks@wikileaks.org Mailing List. January 7, 2007.

[117] Anonymous, "martha stuart pgp," Email to wikileaks@wikileaks.org Mailing List. January 7, 2007.

[118] Julian Assange, "[WL] cryptome disclosure." Email to funtimesahead@lists.riseup.net,wikileaks@wikileaks.org Mailing List. January 8, 2007.

[119] Xan Rice, "The Looting of Kenya," *Guardian*, August 31, 2007, https://www.theguardian.com/world/2007/aug/31/kenya.topstories3, (accessed August 15, 2019).

[120] Jane Sutton, "Guantanamo Operating Manual Posted On Internet," *Reuters*, November 14, 2007, https://www.reuters.com/article/us-guantanamo-manual/guantanamo-operating-manual-posted-on-internet-idUSN1424207020071114, (accessed August 15, 2019).

year, along with the release of Scientology documents, and U.S. politician Sarah Palin's hacked email contents. Numerous other exposures followed and by the end of the decade, the organization was releasing U.S. intelligence materials, including footage from a July 2007 airstrike that killed over a dozen Iraqi citizens, including children. Among the dead was a photojournalist from Reuters and his driver.[121]

Despite his friction with fellow Cypherpunks like Young, Assange took pains to connect his activities with the Cypherpunks. In his book, *Cypherpunks: Freedom and the Future of the Internet*, Assange describes himself as "a leading voice in the Cypherpunk movement since the 1990s," and "one of the most prominent exponents of Cypherpunk philosophy in the world." Within the first several pages of the book, Assange takes umbrage with the U.S. government's investigation of "unprecedented scale and nature" into his activities. He chaffs at the politicians who have called for his arrest or assassination while recasting government efforts to stop him as "censorship." Most of the *Cypherpunks* book is in the form of a conversation between Assange and other WikiLeaks members. The discourse grounds the activities of Assange's WikiLeaks group in the movements from which it came. The ideas of the eponymous Cypherpunks are discussed at length, as are the motives of the Free Software Movement, WikiLeaks member Jacob Applebaum puts the group's perspective plainly: "We need Free Software for a free world." According to the WikiLeaks members, closed software was a form of societal control. "We need software that is as free as laws in a democracy, where everyone is able to study it, to change it, to be able to really understand it and to ensure

[121] Elisabeth Bumiller, "Video Shows U.S. Killing of Reuters Employees," *New York Times*, April 5, 2010, https://www.nytimes.com/2010/04/06/world/middleeast/06baghdad.html, (accessed August 15, 2019).

that it does what they wish." In addition to the ideological connection to Stallman's Free Software Movement, the text connects the practices to their efforts as well. The authors cite the co-operative approach of the movement as "infecting and coming together with the thoughts of American Cypherpunks and Julian Assange/WikiLeaks and so on."[122] WikiLeaks saw themselves as the embodiment of the Cypherpunk movement, and Free Software's ideologies remained front and center.

Just as the Free Software Movement had built a software ecosystem free from commercial control, WikiLeaks sought to create "the most powerful intelligence agency on earth," free from any form of governmental influence or constraint.[123] The group's ambition was no less than to be a geopolitical player, leveraging their technical expertise to "crack the world open."[124] Their objectives recalled Tim May's anarchist vision of cryptography being leveraged as a "tool to demolish governments (including the U.S. government)."[125] But one critical Cypherpunk objective still remained: the creation of an anonymous, cryptography-based currency.

Blockchains and Bitcoins

Anonymous cryptographic currency was a long unattainable objective for the Cypherpunk movement. The desire for the technology features heavily in the thoughts shared by the movement's founders. In John Gilmore's presentation, "Privacy, Technology, and the Open Society," the programmer expressed his desire for a society

---

[122] Julian Assange et al., *Cypherpunks: Freedom and the Future of the Internet* (New York: OR Books, 2012), 7, 14-16, 152-156.

[123] "WikiLeaks Leak," Cryptome, January 7, 2007, http://cryptome.org/wikileaks/wikileaks-leak.htm, (accessed August 15, 2019).

[124] Anonymous, "martha stuart pgp," Email to wikileaks@wikileaks.org Mailing List. January 7, 2007.

[125] Timothy May, "BlackNet Investigations--the Truth (fwd)." Email to cypherpunks@toad.com Mailing List. October 1, 1993.

with anonymous payments, “What if we could build a society where the information was never collected? Where you could pay to rent a video without leaving a credit card number or a bank number?”[126] Cypherpunk co-founder Eric Hughes echoes this desire in *A Cypherpunks Manifesto*, “When I purchase a magazine at a store and hand cash to the clerk, there is no need to know who I am…. When my identity is revealed by the underlying mechanism of the transaction, I have no privacy.”[127] The same interest in anonymous cryptographic currency also resonated throughout the writings of Tim May. In the *Crypto Anarchists Manifesto*, May envisions the “social and economic revolution” that would occur when business was conducted through cryptographic means.[128] Similarly, in May’s August 1993 “BlackNet” thought experiment, the anonymous, Internet-based business purchased information with “CryptoCredits,” the “internal currency of BlackNet.”[129]

As May’s writings demonstrate, the effect of a move to anonymous cryptographic currency goes far beyond the privacy sought by Gilmore and Hughes. Such a technology would affect the power structures of society itself, significantly reducing the control of governments.[130] Decentralized anonymous transactions offered no role to any government, and as such, governmental powers of regulation and taxation would be minimized. Governments themselves would be weakened as well. Tax revenues would diminish, but more importantly, the government would no longer have centralized control of their nations’ economy. Centralized banks, such as the U.S. Federal Reserve use

[126] John Gilmore, “Privacy, Technology, and the Open Society.”.
[127] Eric Hughes, “A Cypherpunk's Manifesto.”
[128] Timothy May. "The Crypto Anarchist Manifesto."?
[129] Anonymous <nowhere@bsu-cs.bsu.edu>. "no subject (file transmission)."?
[130] Timothy May. "The Crypto Anarchist Manifesto."

monetary and fiscal policies as macroeconomic tools to control the economy. In removing a government from the currency, a cryptographic solution would also remove its power to control the economy itself. These changes would have an outsized effect on the United States, whose dollar was frequently held internationally as a reserve currency.[131] A world that moved to a crypto-currency would no longer have a reserve bank, nor reserve funds. This would significantly diminish the role of the United States within the international economy. For a crypto-anarchist like Tim May, such a prospect fit well in line with his goal of using cryptography as a tool to demolish governments.[132]

While working on his PhD at the University of California, Berkeley, David Chaum developed an interest in the concept of electronic currencies. By 1982, he produced an academic paper on the matter, "Blind Signatures for Untraceable Payments," in which he proposed an anonymous electronic currency system based on cryptography.[133] By 1990, Chaum was actively pursuing the creation of such a system. Using his own capital and the funds from a contract with the Dutch government, Chaum formed the DigiCash corporation and began to create a product called "eCash."[134] The technology appeared promising, but after a series of bad deals, the DigiCash shareholders

---

[131] This practice was established as part of the 1944 Bretton Woods agreement. The US's economic hegemony was diminished in 1971 when President Richard M. Nixon moved the nation to fiat currency and in doing so effectively ended the agreement but using the U.S. dollar as reserve currency remained as a de facto practice. Fiat currency is a form of money that has no intrinsic value, but instead only has value due to government support. The United States had previously been on the gold standard, wherein the value of the dollar was connected to the value of gold. The intent of this section is not to debate macroeconomics, but rather to explain the potential effect of a switch to crypto currency. For more insight into the legacy of Bretton Woods, see Michael P. Dooley, David Folkerts-Landau, and Peter Garber, "The Revived Bretton Woods System," *International Journal of Finance and Economics* 9(4), 307-313.

[132] Timothy May, "BlackNet Investigations--the Truth (fwd)." Email to cypherpunks@toad.com Mailing List. October 1, 1993.

[133] David Chaum, "Blind Signatures for Untraceable Payments," *Advances in Cryptology Proceedings* 82, no. 3 (1983), 199-203.

[134] Levy, *Crypto*, 218-219.

replaced Chaum with a new CEO. The change was not enough to help the company recover, and by 1998 the firm had gone out of business.[135]

Despite the short-lived nature of the effort, Chaum's currency was very influential on the Cypherpunks. As detailed previously, Eric Hughes had worked at the company, and upon his return from working with Chaum in Amsterdam, Hughes bonded with Tim May over Chaum's ideas. In addition, the limitations of eCash taught the Cypherpunks the key shortcoming of a commercial solution. A business product would be centered on the company which created it; as such, a commercial digital currency would take the centralized control enjoyed by governments in the conventual economic system and transfer it to the corporation. To offer the privacy sought by the Cypherpunks, a solution would have to be both anonymous and decentralized. Additionally, some of the Cypherpunks took issue with Chaum's decision to patent the process. The Cypherpunks approach of creating Free Software would encourage success by allowing the platform to be widely adopted.[136]

On November 27, 1998 Cypherpunk Wei Dai sent an email directing his fellow Cypherpunks to a paper he had written entitled "B-Money."[137] He opened the document by discussing his interpretation of Tim May's crypto-anarchy. To Dai, crypto-anarchy did not require a world where governments are destroyed, but one in which they are "permanently forbidden and permanently unnecessary." In a digital realm focused on anonymity, "the threat of violence is impotent because violence is impossible," since it is

[135] Greenberg, 119.

[136] Levy, *Crypto*. 219.

[137] Wei Dai. "PipeNet 1.1 and b-money". Email to cypherpunks@cyberpass.net Mailing List. November 27, 1998.

not possible to link digital entities to true names or to determine their physical locations.[138] The problem is that in the physical world, it is the government that provides both the medium of exchange, money, as well as the institutions that enforce contracts. Dai proposed solutions to the problem, offering two "protocols."

In the first protocol, Dai required each participant to maintain a database of electronic monies owed by various pseudonyms. To create money, participants must perform processor-intensive computational functions that can be quickly verified by others. For example, if one unit of wealth is equal to 30 hours of computing time, a computation requiring 90 hours would net the participant 3 units of wealth. To transfer money, participants would broadcast the transaction to all involved. Contracts would be set in a comparable way.[139] At the successful conclusion of a contract, a broadcast message would likewise inform all participants of the closure.[140] In the protocol, Wei Dai defined a way to create wealth through digital scarcity; time-consuming computing efforts define the amount of wealth created. He also established a method to handle decentralization through duplicate databases. In his paper Dai admitted this first protocol would be impractical because of the heavy use of network activity needed to keep the separate databases in sync.[141]

In the second protocol, Wei Dai offered a solution to the inefficiencies of the first approach. Instead of having duplicate databases and constant broadcast messages, the second protocol called for a subset of participants called "servers." These servers would

---

[138] Wei Dai, "B-money," Weidai.com, November 27, 1998, http://www.weidai.com/bmoney.txt, (accessed August 15, 2019).
[139] Ibid.
[140] Contested contracts would require arbitration or a predetermined set of reparation payments.
[141] Wei Dai, "B-money.

stay in contact with each other and journal the transactions made by the entire community, periodically publishing their databases for verification. The servers would be contrasted to each other to confirm accuracy, and each participant could reconcile the server information with their own records. The result would be a digital transaction ledger that can be independently verified for accuracy.[142]

Wei Dai's paper was purely theoretical, but it laid out a workable process for untraceable participants to have a medium of exchange and contract enforcement that met the Crypto-anarchy objective of being completely free of any governmental institutions.

Bitcoin

On October 31, 2008, an entity by the name of Satoshi Nakamoto sent the Cypherpunks an email announcing a paper on "Bitcoin P2P e-cash."[143] The document outlined a technical solution to achieve what Wei Dai had envisioned with B-Money, an "electronic payment system based on cryptographic proof instead of trust," which allowed any two participants to have direct transactions without the need for a third party like the government.[144]

The system described in the white paper featured electronic coins in the form of a succession of cryptographic digital signatures. (See Figure 2) When a coin was transferred, a new set of digital signatures were added to the end of the sequence. The

---

[142] Ibid.

[143] The original Cypherpunk mailing list was retired by John Gilmore in 2001. The Cypherpunks (Gilmore, et al) moved their discourse to the cryptography@metzdowd.com mailing list (among others) at this time. It was to this list that Satoshi Nakamoto sent their correspondence. It is unclear why Gilmore shut down the original list; he offered little justification other than it had become degenerate. This may have been due to interpersonal conflicts on the list. Cypherpunks and related mailing lists such as extropians exhibited a fair amount of interpersonal conflict throughout the late 1990s. In several cases, the Free Software/Open Source schism began to bleed into these groups. This may have been a contributing factor.

[144] Satoshi Nakamoto, "Bitcoin: A Peer-to-peer Electronic Cash System," bitcoin.org, October 31, 2008, https://bitcoin.org/bitcoin.pdf, (accessed August 15, 2019).

recipient could use the signatures to verify ownership. To avoid coins being spent more than once, the system used a chain of time-stamped blocks to record the transactions. These were distributed to a series of decentralized (peer-to-peer) servers that can verify the chain using a unique cryptographic signature called a "hash."[145] Each new block in the chain contains the hash of the previous block. As a result, each block is linked and able to be checked for veracity. Any attempts to manipulate a block would become obvious, as the hash for the altered block would change.[146] To cover up the manipulation would be difficult as each subsequent block in the chain would also need to be modified.[147] The entire block chain would be verified by servers on a peer-to-peer network. These servers would dedicate computing resources to confirming the transactions in a chain. This processing would also generate the wealth within in the system.[148] "The steady addition of a constant of amount of new coins is analogous to gold miners expending resources to add gold to circulation," Nakamoto's paper explained. "In our case, it is CPU time and electricity that is expended."[149]

---

[145] In computing, a hash is like a fingerprint, a unique string of characters that is created by encoding information using a hash algorithm. As with a fingerprint, the hash is unique to the information that was encoded with the algorithm. If the data changes in anyway, the hash will change. If the expected hash does not match the current hash, in indicates that the information has been either corrupted or tampered with. As such, hashing is a way to verify data integrity.

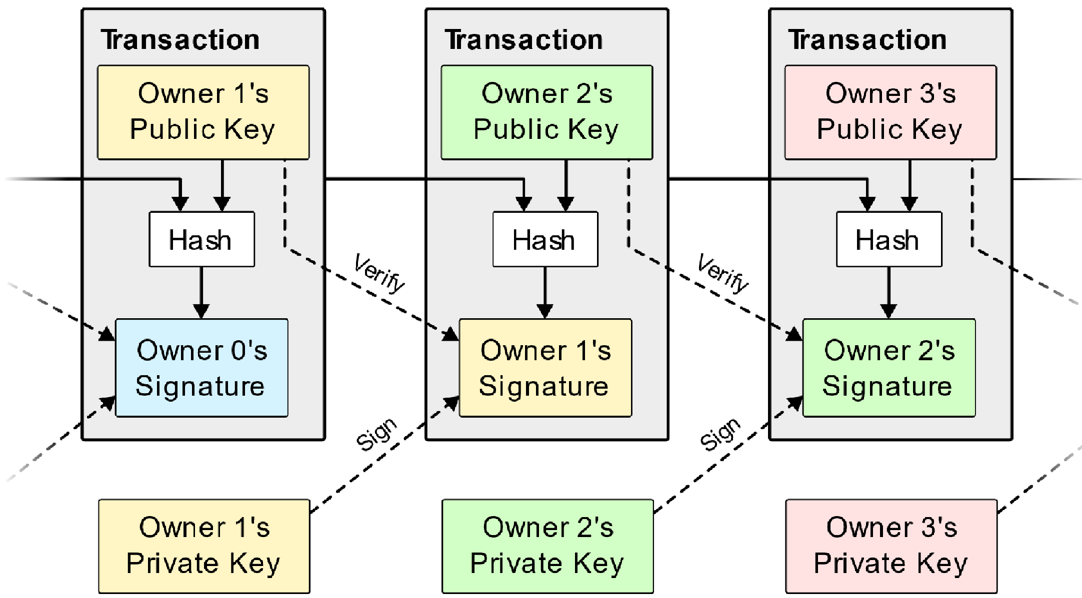
[146] For the sake of brevity, this paper will avoid some of the details outlined in the paper, such as the use of Merkle Trees to concatenate coins to save disk space, etc.

[147] This immutability also creates a challenge for blockchains: coding problems also create irreversible issues. This was seen in June 2016, when a "recursive call bug" was found in the Ethereum cryptocurrency was exploited, resulting in the theft of $3 million. The theft could not be undone without retroactively altering the majority of the mining nodes, a prospect that could endanger the trust in the decentralized network. The ultimate solution was to "fork" Ethereum into two chains. The compromised chain was renamed to "Ethereum Classic," and the new "Ethereum" chain with the theft removed.

[148] To avoid inflation caused by increases in hardware, the Bitcoin system had built-in restrictions on processing as well as reductions in mining payments every four years.

[149] Nakamoto, "Bitcoin."

Figure 2: Block Chain



Source: Satoshi Nakamoto, "Bitcoin: A Peer-to-peer Electronic Cash System," bitcoin.org, October 31, 2008, https://bitcoin.org/bitcoin.pdf, (accessed August 15, 2019).

In the Bitcoin paper, Nakamoto provided an achievable plan to create a cryptocurrency system following the basic properties that Wei Dai had outlined a decade prior. The system centered around a shared ledger, the blockchain, which could be used not only for currency, but to record and verify any manner of transactions, anything from interpersonal contracts to distributed asset inventory. It could even be used to provide an audit trail for voting. As in Dai's proposal, the transactions were publicly announced, and wealth was generated through processing power. Bitcoins could be earned by individuals who used their computers to perform energy intensive "mining," using Bitcoin's software for confirming the transactions after they occurred.[150] The system was decentralized, so if one part of the system was taken off line, the currency would still be available through redundant systems.

On February 11, 2009, Nakamoto announced the release of Bitcoin v.1. The person or persons who had previously published the "Bitcoin P2P e-cash" followed up on their proposal by developing the platform and releasing it under the MIT Free Software license.[151] The Cypherpunks' holy grail, anonymous cryptographic currency, had finally been realized.

It is unclear who Satoshi Nakamoto was. The person or persons behind the software were active on Cypherpunk mailing lists but remained focused solely on their innovation. Clues to the real identity of the programmer or programmers has remained elusive. However, where the identity remained a mystery, the motivations did not. In the Bitcoin Free Software announcement, Nakamoto explained the drive to create the software by implicating the conventional banking system.

> Banks must be trusted to hold our money and transfer it electronically, but they lend it out in waves of credit bubbles with barely a fraction in reserve. We have to trust them with our privacy, trust them not to let identity thieves drain our accounts. [152]

With Bitcoin, that trust was no longer required. The software was based on cryptographic proof, verified continually by distributed peer computers. No corporate or governmental third party was needed. Nakamoto's distrust of the government and the banks can also be seen in the software itself. In the genesis block, the first block, on the

[150] Confirming the digital signatures on a coin's blockchain to ensure that it had not been tampered with and, in the process, also confirming the chain of ownership.

[151] Satoshi Nakamoto, "Bitcoin Open Source Implementation of P2p Currency," Nakamoto Institute, February 11, 2009, https://satoshi.nakamotoinstitute.org/posts/p2pfoundation/1/#selection-9.0-9.50, (accessed August 12, 2019).

[152] Ibid.

first block chain, the author encoded the headline from the *London Times*, “The Times 03/Jan/2009 Chancellor on brink of second bailout for banks.”

Nakamoto’s decision to include this message in the genesis block, directly ties the cryptocurrency innovation with the central banking system that it intended to replace. In addition to the distrust of commercial banks detailed above, Nakamoto had similar concerns about governmental banks, particularly in that they “must be trusted not to debase the currency, but the history of fiat currencies is full of breaches of that trust.” [153] The software offered a weapon against governmental overreach, enabling users to “win a major battle in the arms race and gain a new territory of freedom for several years.” In this, the software’s distributed nature was vital, as “governments are good at cutting off the heads of a centrally controlled networks.”[154]

Just as the Free Software Movement had created a software community independent of corporate software, the Cypherpunk author(s) of Bitcoin introduced a currency that did not require government or corporate involvement. The macroeconomic levers employed by central banking institutions held no influence over the cryptographic currency-based economy and the anonymous nature of the transaction meant that it was free from government oversight. As Tim May had predicted, the government would have “hard time collecting taxes” or “regulating the behavior of individuals…when it can't

[153] Ibid.

[154] Satoshi Nakamoto. "Bitcoin P2P e-cash paper". Email to cryptography@metzdowd.com Mailing List. November 6, 2008.

even tell what _continent_ [SIC] folks are on!"[155] Free software combined with strong encryption, enabled freedom from both corporations and the government.[156]

Summary: Cypherpunks Write Code

By the start of the twenty-first century, the ideologies and processes that had propelled the Free Software Movement were translated in other spheres, as various groups adopted the ideas and the software to advance their geopolitical agendas. In July 2001, as Free Software activists were in the streets protesting the government's DMCA-fueled crackdown on Free Software, members of the hacking group, the Cult of the Dead Cow, introduced the concept of hacktivism, the approach of using technology and electronic media to advance activist causes, and promote human rights.[157] They formed Hacktivismo, and the group began releasing programs under their revised Free Software license, the Hacktivismo Enhanced-Source Software License Agreement (HESSLA). Although heavily influenced by Richard Stallman's GNU public license, the HESSLA restricted the use of the software in any manner that could "subvert or infringe the freedoms of end-users."[158]

By 2003, the activist group Anonymous had formed in fringe Internet forums. The group began to use Free Software tools, such as Apache J-Meter, to take down the web servers of corporations, nations, and groups with which they disagreed. They adopted

[155] Tim May, "The Cyphernomicon.

[156] Freedom in theory. Nations like China are pursuing central-bank cryptocurrencies that afford them the ability to spy on financial transactions and challenge the dominance of the U.S. dollar in International monetary markets.

[157] Menn, 91.

[158] "The Hacktivismo Enhanced-Source Software License Agreement."

Free Software and its distributed approach to projects, but they used the programs and the Free Software schema to perform an Internet-based assault on their chosen enemies. By taking down the websites of those with whom they disagreed, they stifled their opponent's freedom of expression. Thus, their actions were in direct opposition to the Free Software Movement's worldview, which was focused on civil liberties, in particular, the idea that code is speech. The Anonymous movement deliberately suppressed these freedoms and acted in a manner antithetical to Free Software ideologies.

Meanwhile, a group of Free Software programmers and cryptography enthusiasts joined to form the Cypherpunk group in the early 1990s. Focused around meetings at the Free Software support company Cygnus, and later, a mailing list hosted by the company's co-founder, John Gilmore, the group members spanned the Free Software, cryptography, and hacking community. In addition to co-founders Eric Hughes and Tim May, the Cypherpunks included key figures from public cryptography such as Whitfield Diffie, Martin Hellman, and Phil Zimmerman. The co-founders of John Gilmore's Electronic Frontier Foundation, Mitch Kapor and John Perry Barlow, were also among the list members. Mark "Phiber Optik" Abene, a hacker from the infamous hacker groups Legion of Doom and Masters of Deception, was also a Cypherpunk, as was legendary phone phreaker John "Captain Crunch" Draper.[159] Even Clifford Stoll, famous for hunting down the KGB hacker Markus Hess, was a list recipient.[160] Many prominent

[159] "Cypherpunk List Authors by Number of Posts (Highest First)," cryptoanarchy.wiki, https://mailing-list-archive.cryptoanarchy.wiki/authors/by-posts/, (accessed August 12, 2019).

[160] Stoll was not an active poster. His solitary email to the list offered "a nice fake mail program that interfaces with emacs." As discussed in his 1995 book *Silicon Snake Oil*, Stoll had a more prosaic and foreboding sense of what the Internet would become.

Free Software community members were also Cypherpunks, including Fen Labalme and even the founder of the Free Software Movement, Richard Stallman.[161]

However, it was not the group's membership that mattered in so much as what the group did. Co-founders John Gilmore and Tim May espoused visions of a future where cryptography-focused Free Software would bring about radical social transformations. May advocated for what he called Crypto-Anarchy, where the new software would undermine conventional world governments. While acknowledging that his vision would result in "Terrorism, assassination, crime," and groups like the Russian Mafia, and Aryan Nation "gone digital," May was fine with this, stating, "Many of us are explicity [SIC] anti-democratic, and hope to use encryption to undermine the so-called democratic governments of the world."[162] As for the rest of the Cypherpunks, as co-founder Eric Hughes wrote in *A Cypherpunks Manifesto*, "Cypherpunks write code."[163]

Just as Richard Stallman's Free Software created a world-wide community focused on individual liberties and built a digital sphere free from corporate control, the Free Software creations of the Cypherpunks aimed to create a libertarian paradise centered on digital cryptographic technologies. They imagined a world free from government hegemony, focused entirely on the wants of the individual. The Cypherpunks wrote code to realize this self-interested utopia. Cypherpunk John Young created Cryptome, a website that undermined corporations and governments by displaying their undisclosed shortcomings. Cypherpunk Julian Assange sought to take this idea further,

[161] Raymond was an active, early supporter of the Cypherpunks and in 1993, offered to work on the FAQ for the group. Stallman's only correspondence with the group was in October 1998 as the Open Source schism bled over into the group and resulted in a debate about software licenses. Stallman's correspondence was focused on responding to Raymond's emails and clarifying the GPL licenses and their functions.

[162] Tim May, "The Cyphernomicon."

[163] Eric Hughes, "A Cypherpunk's Manifesto."

launching WikiLeaks to generate income, acquire state secrets, and act as a private intelligence agency. Cloaked in the façade of journalism, the group sought to alter the geopolitical landscape through the release of classified information and state secrets. Meanwhile, Free Software programmers developed applications that could be used to form DarkNets, anonymous encrypted networks within the Internet. Finally, the enigmatic Cypherpunk entity known as Satoshi Nakamoto created and shared the Free Software cryptocurrency, Bitcoin, realizing the long awaited Cypherpunk dream of having an anonymous, untraceable, digital currency. With the Free Software Movement, the medium was the message; the software was a vector to transmit and replicate Richard Stallman's ideological payload, the promise of software free from corporate control. With the Cypherpunks, this had evolved into something more potent. The Free Software they composed and shared was intended to reshape society to comport with their ideology and fulfill an anti-authoritarian anarchist vision of cryptographic capitalism and individual privacy. The Cypherpunks wrote code, and the time for anarchy had come.

CONCLUSION: TOWARD BINARY FREEDOM

Richard Stallman ran his first computer program in 1969 while working at IBM's New York Scientific Center in downtown Manhattan. At the time, the young programmer was still struggling to fit in with his classmates in high school, and the specter of the Vietnam war loomed large. Stallman was opposed to the war, and to the authoritarian threat of the draft. At the same time, the programmer was not comfortable with his contemporaries who opposed the conflict: he disliked rock 'n' roll music, hated drugs, and found counter-cultural attitudes to be unsophisticated and anti-intellectual. Stallman gravitated toward technology, math, and science. Mathematical problems and computer programs were like puzzles; they were solvable, and they offered a logical world, one seemingly free of the complications of relationships or geopolitical conflicts.

Following his graduation, Stallman enrolled at Harvard University. The awkwardness he had felt with his high school cohort reappeared at the prestigious institution. A new school meant new people and with that, more uncomfortable human interaction. However, the situation changed at the end of his first year at Harvard, when Stallman learned of a special laboratory at MIT. The college freshman visited the MIT AI Lab with the hope of finding spare manuals with which to continue his programming education. Instead, he discovered a group of hackers, programmers focused entirely on their work, and whose status was determined by their technical achievements. The lab was an open place, with unrestricted access to computer terminals. The programmers within the lab harbored a mistrust for authority and were focused instead on freely sharing information with each other. When a technical problem arose, it was perceived as a challenge and the programmers gained social status within the lab by implementing solutions. The hackers were judged by their programming skills. Coding was perceived as an artistic endeavor, and the programmer who produced the most elegant solution to a problem was held in the highest regard. Although he had felt awkward with social situations previously, Stallman felt at home within the meritocratic lab environment. He quickly became part of the hacker community, regularly participating in all-night programming marathons, and joining in other lab rituals, such as the group's weekly Chinese-food dinners.[1]

The practices, values, and beliefs of the lab's programmers informed the development of Stallman's worldview. The meritocratic social structure and the technological focus resonated with Stallman and informed how he would later interpret political and social relationships. But the lab was a walled garden; there were changes occurring in the outside world that had not yet made it to MIT. When the computing industry began, the sharing of software was common. To address the scarcity of programmers in the workforce, many computer manufacturers facilitated the distribution of shared programs between their clients. This began to change with IBM's unbundling of hardware, software and services in 1969. With this action, software became a commodity, a marketable product. Copyright law changes made in 1980 reversed precedent and made computer software able to be copyrighted and enabled manufacturers to limit consumer rights through licensing terms.[2] This new re-interpretation of software finally came to the lab in the early 1980s when the two competing corporations, LMI, and

[1] Williams,30-41.

[2] "H.R.6933 - An act to amend the patent and trademark laws," Congress.gov, December 12, 1980, https://www.congress.gov/bill/96th-congress/house-bill/6933 (accessed April 16, 2018).

Symbolics, were both licensed to sell the LISP programming technologies that had been created in the lab. By 1983, the result was capitalist competition that divided the labs denizens, destroyed the open hacking culture, and drove the programmers out from the Edenic hacker's paradise.

For Stallman, the first place in his life where he had felt comfortable had been destroyed. Even worse, the programmer, who had refused to take sides in the conflict, found himself the target of corporate malfeasance and spying. Unsurprisingly Stallman developed an aversion to corporations and the idea of proprietary software, which restricted consumers from accessing, altering, or, in some cases, even discussing the programs they owned. The programmer viewed the conflict in Manichean terms and as such, he had a choice: he could either participate in an industry he felt to be immoral, or he needed to find an alternative.

Utopian Dreams

Richard Stallman's response was to develop a schema that would create a sphere apart from corporate control. It would enshrine his ideological views: Software should be open, freely shared information. Programs were a form of artistic expression. Moreover, any authority that wished to restrict this expression would be abrogating the civil liberties of both the software's creators and its users. Authority should be mistrusted and avoided. To meet this objective, Stallman launched the GNU project and developed the GNU public license. These tools allowed him to create a computing environment that would protect the freedoms that he believed were being abolished with proprietary software – the freedom for a consumer to run the programs for any purpose and to review and change the source code if they needed. The GNU public license also permitted users to share copies of original software, as well as any subsequent changes, with others in the programming community. With the project, Stallman sought to recreate a utopian vision of the MIT AI lab hacking community on a global scale. His ideological conception of software and its associated civil liberties was now enforceable by software licenses. Central to Stallman's plan was the belief that the technology itself could change society; the GNU software was both the vector and the payload, transmitting his digital civil libertarian ideology, while at the same time creating a digital world free from corporate control. But Stallman was not the only one with utopian dreams. Early computing was driven by individuals who imagined an idealized future that could be realized through technology. These visions of the future, although not always realized, helped drive the progression of software and computing to their modern forms.

In 1964, Dartmouth Professors John G. Kemeny and Thomas E. Kurtz developed the BASIC programming language with the intent of making computers more accessible to a broad audience, and in the process, better integrating man and machine. Kemeny believed that the relationship between computers and mankind was akin to symbiotic species. For both species to advance, they needed to work together. Once this symbiotic evolution occurred, it would solve many of the problems facing society. It was this belief in an idyllic cooperative future that informed Kemeny's drive to have Dartmouth make his creation available to all.[3]

In July 1945, the Director of the Office of Scientific Research and Development, Dr. Vannevar Bush published an article in *Atlantic* magazine. Entitled "As We May

---

[3] John G. Kemeny, *Man and the Computer,* 19.

Think," the article relayed a vision of interconnected knowledge. Bush's concept centered around a theoretical mechanical device which he called a "Memex." This apparatus would enable an individual to centralize all their media and communications and would be organized in a manner that permitted the linking of information in an associative manner. As the individual acquired new material, he could create additional links, creating topical connections between the various forms of media.[4] This specific system never came to fruition, but the idea and its promise of an interconnected nirvana inspired many later innovators.

One such innovator was Douglas Engelbart, who first read Vannevar Bush's "As We May Think," while in the U.S. Navy. Five years later, Engelbart experienced a revelation of what computers were to become. He imagined himself before a computer screen, navigating the system based on symbolic icons – Bush's Memex on an electronic computer.[5] Engelbart realized his vision through the creation of the Graphical User Interface (GUI). The GUI made computing more broadly accessible by removing the need for users to learn arcane text commands. In the process, it abstracted computer users from the software itself. With the GUI, users no longer needed to be programmers to use a computer. The graphical interface allowed them to navigate to programs and execute them.

In 1960, a college student named Ted Nelson proposed his own derivative of Bush's Memex. Entitled Project Xanadu, Nelson's concept was to have a global database for accessing any of the world's literature. The system included what Nelson termed

---

[4] Vannevar Bush, "As We May Think.".
[5] Rheingold, *The Virtual Community*, 7-9.

hypertext, his conception of Bush's proposed inter-linking between documents.[6] Although Nelson worked on the project for decades, his forward-thinking vision was never fully realized.[7] However, as with the Memex, Nelson's ideas also informed later projects.

In 1989, Tim Berners-Lee, a consultant at the Conseil Européen pour la Recherche Nucléaire (CERN), created his own version of the Memex in the form of text-based platform with links between documents. Berners-Lee's version expanded the topical connections between material into an interconnected Internet-based system which scientists around the world could use for distributed research. Berners-Lee named his inter-connected text *hypertext* in reference to Nelson's Xanadu. He and his associates at CERN called their software the World Wide Web, a reflection of the web of connections created by the content. With the World Wide Web, aspects of these visions began to be realized: The interconnected media of Bush's Memex was merged with the global nature of Nelson's Xanadu, and when combined with Engelbart's graphical user interface, the system democratized computing, making global knowledge and resources available to anyone with access to a computer and the Internet.

Even the Internet itself has its origins in technological idealism. In 1962, the United States Department of Defense's Advanced Research Projects Agency appointed J.C.R. Licklider to head the Information Processing Techniques Office (IPTO). Although formally trained as a psychologist, Licklider was best known for his work in computing. In 1960, Licklider published "Man-Computer Symbiosis," a document in which he

---

[6] Ibid., 100-101.

[7] Ted Nelson released the source code to his project in 1998. In 2015, he announced the availability of OpenXanadu, a form of the idea running on the world wide web. It can be found here: http://xanadu.com/xanademos/MoeJusteOrigins.html, (accessed January 21, 2020).

envisioned a future wherein people and machines would have a symbiotic relationship, an "interdependency of humans and computers working in unison as a single system."[8] A year after his IPTO appointment, Licklider shared another vision. In a memo entitled, "Memorandum for Members and Affiliates of the Intergalactic Computer Network," Licklider described the eponymous global "integrated network" of computers on which information could be exchanged and programmatical work completed.[9] In their book *Where the Wizards Stay Up Late: The Origins of the Internet*, authors Katie Hafner and Matthew Lyon attribute Licklider's vision of this global network to his belief that technological progress would save humanity.[10] Licklider's dream of a global network was realized by the IPTO's subsequent director, Robert Taylor, who led the effort to create the ARPAnet, the predecessor to the Internet.

Many of these technological ideals were extropic, beliefs predicated on the assumption that technology would continually evolve and, in the process, improve human existence. However, visions of paradise are subjective. For some, a symbiosis between man and computers would bring answers to societal issues. For others, however, the creation of their idyllic future first required the destruction of the present state. Such was the case for Cypherpunks like John Gilmore and Tim May. As with Stallman, they sought to use Free Software to change society. But where Stallman had been focused on the deleterious effect that commercial software was having of the programming community, the Cypherpunks sought to create a sphere apart from governmental control. Their

---

[8] Katie Hafner and Matthew Lyon, 31.

[9] J.C.R. Licklider, "Memorandum for Members and Affiliates of the Intergalactic Computer Network," Kurzweil Accelerating Intelligence, last modified December 11, 2001, https://www.kurzweilai.net/memorandum-for-members-and-affiliates-of-the-intergalactic-computer-network, (accessed January 21, 2020).

[10] Hafner and Lyon, 34.

extropian vision required cryptographic technologies that would allow them to create a "Crypto Libertaria," an Internet-based Promised Land redolent of Ayn Rand's "Galt's Gulch," a happy valley where technically-able "men of the mind" retreated from their obligations to society.[11] Rand's fictional libertarian paradise was free of government oversite, taxes, and societal mores.[12] However, the Cypherpunk Canaan required the conquest of traditional structures. This included demolishing national governments, creating privatized intelligence agencies, and establishing monetary systems.[13]

Unforeseen Consequences

The significance of historical events can be most clearly seen in their outcomes. As such, a historical investigation must connect the changes that occurred in the past with their present ramifications. In the decades since Stallman's GNU Project, the ideology and schema of the Free Software Movement was transposed to other fields and evolved into other movements. For example, the commons-based peer production of Free Software informed the creation of Wikipedia, a web-based encyclopedia. Since its inception, Wikipedia, has come into common usage and had such peer-based refinement that researchers like Historian Roy Rosenzweig have audited the platform and reported surprising levels of accuracy.[14] Other derivatives, such as the Creative Commons, have encouraged the creation of a "Re-mix" culture, where various forms of media – music,

---

[11] Tim May, "Libertaria in Cyberspace." Email to Extropians@gnu.ai.mit.edu Mailing List. September 1, 1992.

[12] Ayn Rand, *Atlas Shrugged*, 35th ed. (New York, New York: Signet, 1992, 1985), 769. https://ia800305.us.archive.org/19/items/AtlasShrugged/atlas%20shrugged.pdf, ,(accessed January 21, 2020), 769.

[13] Tim May, "The Cyphernomicon."

[14] Roy Rosenzweig, *Clio Wired: The Future of the Past in the Digital Age* (New York, New York: Columbia University Press, 2011), 60-63.

pictures, and films -- are created, edited, and shared.[15] However, not all of the consequences were necessarily intended.

The business entrepreneurs who encouraged commercial adoption of Free Software by rebranding it Open Source had a surprising level of success. VA Linux, an Open Source hardware vendor had its record-setting IPO on December 9, 1999. Eric Raymond, the Open Source Movement's self-appointed leader and a VA Linux board member, wrote the article "Surprised by Wealth" after the first day of trading saw VA Linux's stock close at $239 a share. At the time he wrote the article, Raymond's stake in the company was worth thirty-six million dollars.[16] But by the time Raymond's options were vested four years later, VA Linux was selling at two dollars a share, less than 1% of its original value.[17] The commercial adoption that boosters like Raymond had worked so hard to promote had an unintended effect; as Open Source became mainstream, traditional hardware vendors like IBM and Hewlett Packard began to support the software on their equipment.[18] The formerly niche market that had been built by Open Source hardware vendors like VA Linux was soon dominated by industry giants.

As with the hardware market, Open Source software was also assimilated into the offerings of existing corporate software. In the years that followed the introduction of Open Source, corporations like Apple Computers and Microsoft began to commonly use and share Free and Open Source software. Microsoft's Windows 10 product was offered

---

[15] Lawrence Lessig, *Remix: Making Art and Commerce Thrive in the Hybrid Economy* (London:Penguin Books, 2009), 68-70.

[16] Leander Kahney, "Open Source Rich Opens Mouth," Wired.com, December 10, 1999, https://www.wired.com/1999/12/open-source-rich-opens-mouth/, (accessed January 21, 2020).

[17] Avimanyu Bandyopadhyay, "A Timeline of VA Linux: Through the Years," Medium, October 5, 2018, https://medium.com/@avimanyu786/a-timeline-of-va-linux-through-the-years-6813e2bd4b13, (accessed January 21, 2020).

[18] "IBM Launches Biggest Linux Lineup Ever," IBM.com, March 2, 1999, https://www-03.ibm.com/press/us/en/pressrelease/2262.wss, (accessed January 21, 2020).

free to users that wished to upgrade. This version of the Window desktop operating system also included the ability to run a fully functional Linux kernel under it.[19] Free and Open Source companies were similarly absorbed by existing corporations. For example, both the Cygwin support company and Red Hat, the maintainer of one of the most popular Linux distributions are now owned by IBM.[20]

The continued growth of the Internet and the ubiquity of network connected devices has altered the way in which software is licensed. In new approaches such as "Software as a Service" (SAAS), software is no longer purchased. Instead, users subscribe to Internet-based applications that replace traditional desktop-based programs. Similar licensing schemes exist for desktop (Virtual Desktop Infrastructure or VDI) and server hosting environments (Infrastructure as a Service or IAAS.) In these arrangements, consumers have no control over their software, and their purchase of the product only grants them the ability to use it for a limited period.

Software is not the only thing that has evolved into more restrictive forms. Since its passing, the 1998 Digital Millennium Copyright Act has also been leveraged by manufacturers such as General Motors and John Deere to ensure themselves a monopoly in future repairs. Citing DMCA protections for the computerized components of their modern offerings, the organizations held that any non-dealer repairs or modifications

[19] Tom Warren, "Microsoft Will Ship a Full Linux Kernel in Windows 10," The Verge, May 6, 2019, https://www.theverge.com/2019/5/6/18534687/microsoft-windows-10-linux-kernel-feature, (accessed January 21, 2020).

[20] Jim Whitehurst, "IBM Closes Landmark Acquisition of Red Hat for $34 Billion; Defines Open, Hybrid Cloud Future," Redhat.com, July 9, 2019, https://www.redhat.com/en/about/press-releases/ibm-closes-landmark-acquisition-red-hat-34-billion-defines-open-hybrid-cloud-future, (accessed January 21, 2020).

were illegal – even if performed by the owners of the equipment.[21] Far from "kindling flames of worry over intellectual-property expansion," as Anthropologist Chris Kelty put it, the Free Software Movement's concerns with the law were remarkably prescient.[22] The use of the DMCA legislation, as well as the introduction of SAAS software licenses, has fundamentally challenged the concept of ownership. This can be seen in John Deere's post-sale restrictions on the equipment they sell. If farmers who purchased John Deere tractors are legally forbidden from maintaining them, then do they really own them? If so, what, specifically does ownership mean? As a result of these issues, some farmers have begun buying antique equipment that come free of such repair constraints.[23] The leveraging of the DMCA to restrict the use of material purchases also resulted in the emergence of a "Right to Repair" movement, which advocated against the laws' restrictions and encouraged the creation of consumer-focused legislation proposals in dozens of states.[24]

Extropian Nightmares

Some of the most startling consequences of the evolution of Free Software ideology are seen in its legacy of online protest groups and in the products of the Cypherpunks. In both cases, the ramifications of the groups increasingly connect to broader geopolitical events.

---

[21] Kyle Weins and Elizabeth Chamberlain, "John Deere Just Swindled Farmers Out of Their Right to Repair," Wired.com, September 19, 2018, https://www.wired.com/story/john-deere-farmers-right-to-repair/, (accessed January 21, 2020).

[22] Christopher Kelty, *Two Bits: The Cultural Significance of Free Software*, Experimental Futures (Durham, North Carolina: Duke University Press, 2008), 99.

[23] Matthew Gault, "Farmers Are Buying 40-year-old Tractors Because They're Actually Repairable," Vice.com, January 7, 2020, https://www.vice.com/en_us/article/bvgx9w/farmers-are-buying-40-year-old-tractors-because-theyre-actually-repairable, (accessed January 21, 2020).

[24] Timothy Lee, "How Copyright Law Threatens Your Right to Repair Your Car," Vox.com, May 28, 2015, https://www.vox.com/2015/5/26/8656281/dmca-car-repair-rights, (accessed January 21, 2020).

Hacktivism

Since the introduction of Hacktivism with the founding of Hacktivismo in 1999, online protests have become increasingly less prevalent. This has been due, in great part, to the collapse of groups like Anonymous. In 2013, the FBI arrested a core group of Anonymous crackers who styled themselves "Lulzsec." The group had been responsible for finding the security compromises that were exploited in the attacks performed by Anonymous. Their arrest resulted in decreasing efficacy in Anonymous's subsequent campaigns.[25] In addition, political disagreements divided the Anonymous group in the lead up to the 2016 U.S. Presidential election. At one point during the campaign season, the group voted to perform an anti-Trump operation. Despite the group's fundamental commitment to a democratic decision-making process, Anonymous members found themselves under siege by other members who opposed the campaign. The group's on-line resources, including their Twitter account were taken over by pro-Trump agents. In addition, some Anonymous members also reported that they felt the group was being used by outside forces to influence the election.[26] As the group's name would suggest, the online participants of Anonymous are unknown, so it is difficult to establish the depth of these issues. However, there were physical demonstrations that accompanied Anonymous's online campaigns, and the influence of external groups is more apparent when examining this broader protest community. This can be seen with Cassandra

---

[25] Charles Arthur, "Lulzsec: What They Did, Who They Were and How They Were Caught," *The Guardian*, May 16, 2013, https://www.theguardian.com/technology/2013/may/16/lulzsec-hacking-fbi-jail, (accessed January 21, 2020).

[26] Nicky Woolf, "An Anti-Trump Campaign Sparks a Civil War Among Anonymous Hackers," *Business Insider*, March 24, 2016, https://www.businessinsider.com/anonymous-hackers-donald-trump-2016-3, (accessed January 21, 2020).

Fairbanks, an activist who helped arrange both Occupy and Anonymous protests.[27] Fairbanks initially advocated for a far-left agenda and was an avowed supporter of Bernie Sanders in the 2016 campaign. Once Hillary Clinton emerged as the Democratic nominee, Fairbanks underwent an instant transition, and began to espouse a far-right agenda and became an avowed supporter of Alt-Right ideologies and Donald Trump.[28] Fairbanks became employed as a journalist for the Russian propaganda news outlet, Sputnik, and it was in this role that she communicated directly with the Guccifer 2.0, the on-line personae of the Russian GRU agents responsible for hacking Hillary Clinton's campaign.[29] Following the election, Fairbanks was invited to the Trump White House, where she gained notoriety after being photographed displaying a "white power" gesture during a photo session.[30]

Cypherpunks

As with the Free Software Movement's apprehensions with the DMCA, the Cypherpunks found their concerns over personal privacy reified. In the 1990s, the group suspected the U.S. Government of aspiring to become a surveillance state. Throughout the early 2010s, leaks about government surveillance programs such as PRISM and XKeyscore confirmed these fears. Through these technologies and with the cooperation

---

[27] Ginny Graves, "Attention Rapists: You've Met Your Match," June 6, 2013, https://www.glamour.com/story/attention-rapists-youve-met-your-match, (accessed January 21, 2020).

[28] Rebecca Nelson, "Cassandra Fairbanks loved Bernie Sanders. Now she's a Donald Trump superfan," *Cosmopolitan*, May 18, 2017, https://www.cosmopolitan.com/politics/a9653830/cassandra-fairbanks-donald-trump-deplorable/, (accessed January 21, 2020)

[29] Jack Moore, "Sputnik News under FBI Investigation as Russian Propaganda Arm," *Newsweek*, September 11, 2017, https://www.newsweek.com/russian-network-sputnik-under-fbi-investigation-possible-kremlin-propaganda-662844, (accessed January 21, 2020).

[30] Emily Shugerman, "Two Members of Alt-Right Accused of Making White Supremacist Hand Signs in White House After Receiving Press Passes," April 29, 2017, https://www.independent.co.uk/news/world/americas/us-politics/white-power-hand-symbol-cassandra-fairbanks-mike-cernovich-alt-right-white-house-a7709446.html, (accessed January 21, 2020).

of corporations such as Google, Microsoft, and Apple, the U.S. government had free reign to access the digital communications of private citizens.[31] And, as suspected with the key embargo plans in the mid-1990s, the government's surveillance of the citizenry did not include the government first receiving warrants, as required by the fourth amendment of the constitution. Meanwhile, the crypto creations of the cypherpunk still continue to effect societal change:

Bitcoins & Darknets

Although the identity of its cypherpunk creator remains unknown, Bitcoin, the first crypto-currency, has been wildly successful, having posted gains of more than 9,000,000% since 2010.[32] The technology has also seen the emergence of similar currencies, all free from the oversite and control of any governments or the central banking system.

Cryptocurrencies and the anonymous darknets pioneered by the Cypherpunks have fostered international markets for illegal activity. Drugs, human trafficking, and pornography has been facilitated by these technologies. The government's ability to police these networks remain limited. For example, in 2013, U.S. law enforcement was able to shut down the notorious bitcoin-based drug market called the Silk Road. By the time the site was deactivated, the Silk Road had already produced Bitcoin revenue valued

[31] Glenn Greenwald, "Xkeyscore: NSA Tool Collects 'nearly Everything a User Does On the Internet,"' *The Guardian*, July 31, 2013, https://www.theguardian.com/world/2013/jul/31/nsa-top-secret-program-online-data, (accessed January 21, 2020).

[32] Vildana Hajric, "Bitcoin's 9,000,000% Rise This Decade Leaves the Skeptics Aghast," Bloomberg, December 31, 2019, https://www.bloomberg.com/news/articles/2019-12-31/bitcoin-s-9-000-000-rise-this-decade-leaves-the-skeptics-aghast, (accessed January 21, 2020).

at $1.3 billion.[33] But, for every network that was taken down, another would appear in its place. In the years that followed the demise of the Silk Road, "Hydra," a Russian darknet marketplace, has appeared and rapidly expanded. Recently, the network introduced its own cryptocurrency, as part of an ominous effort to "start a new era in the West" at a scale that is "hard to imagine."[34]

WikiLeaks

Perhaps the most public manifestation of the cypherpunk legacy has been WikiLeaks. In June 2012, WikiLeaks founder Julian Assange lost his legal fight to avoid extradition to Sweden where he stood accused of rape by various women. To avoid the consequences of this legal decision, Assange approached the Ecuadorian embassy in London and requested political asylum. Rafael Correa, the president of Ecuador agreed to the request and allowed Assange to take up residence in the London embassy. Assange's status as a fugitive and his indebtedness to Ecuador for his protection did not restrain WikiLeaks's activities. In fact, the group's geopolitical machinations increased with their efforts to influence the 2016 U.S. Presidential election as detailed in Special Counsel Robert S. Mueller's investigation into Russian campaign interference.[35]

In 2016, from his sanctuary in the affluent Belgravia district of Central London, Assange began coordination with various parties against candidate Hillary Clinton's campaign. In March, WikiLeaks provided emails acquired through governmental

[33] Jose Pagliey, "FBI Shuts Down Online Drug Market Silk Road," CNN, October 2, 2013, https://money.cnn.com/2013/10/02/technology/silk-road-shut-down/index.html, (accessed January 21, 2020).

[34] Andrew Fenton, "New Wave of Darknet Crypto Icos Led by Russian 'silk Road,'" Micky Media, December 17, 2019, https://micky.com.au/new-wave-of-darknet-crypto-icos-led-by-russian-silk-road/, (accessed January 21, 2020).

[35] Robert Mueller, *Report On the Investigation Into Russian Interference in the 2016 Presidential Election* (Washington, DC: U.S. Department of Justice, 2019), 44

Freedom of Information Act (FOIA) requests. However, by July 2016, WikiLeaks was working with the Russian intelligence directorate, Glavnoye Razvedyvatel'noye Upravleniye or GRU. A Russian agent posing as Guccifer 2.0 provided WikiLeaks with a PGP-encrypted archive of documents stolen from the Democratic National Committee's network. On July 22, 2016, three days before the start of the Democratic National Convention, Wikileaks released more than twenty thousand of the emails and documents provided by the Russian operatives. Later, in September 2016, WikiLeaks coordinated with Donald Trump Jr., to help his father's campaign prepare for the imminent launch of a web-site, TrumpPutin.org, which documented the many ties between the candidate and the Russian leader. WikiLeaks provided the candidate's son with the credentials to the yet-to-be-launched website. The collusion between the GRU and WikiLeaks continued into the fall, with the Russian agents providing additional documents for WikiLeaks to distribute.[36]

Wikileaks' support of the Trump campaign escalated as the election neared. On October 7, 2016, the *Washington Post* revealed *Access Hollywood* recordings in which candidate Donald Trump bragged to a reporter about sexually assaulting women. Within a day of the story's release, Trump began to lose support as Republican politicians rescinded endorsements and suggested that the candidate drop out in favor of Vice-Presidential nominee Mike Pence. In response to the growing concerns over the incriminating tape, Wikileaks began to publicize emails stolen from Clinton Campaign member John Podesta. In the month leading up to the election, WikiLeaks circulated

[36] Ibid., 44-47,59-60.

thirty-three releases of GRU-supplied material.[37] The near-daily releases kept news coverage focused on the contents of the purloined Clinton documents. At the same time, WikiLeaks refused to release information that it had acquired on Trump.[38] WikiLeaks, the organization which had promised its "only interests will be truth and freedom of information," was now curating and selectively releasing information to further a political agenda favoring one presidential candidate.[39]

Binary Freedom

The Free Software Movement, which began in response to Richard Stallman's mistrust of corporate influence in software has grown and evolved. The licensing and production schema developed by Stallman informed the creation of countless other projects, from academic software licenses to on-line encyclopedias and media publication platforms. The Open Source re-articulation of Free Software sold the schema as a programming model and brought corporate adoption of the applications produced by the movement.

The ideological evolution has taken a less mainstream path. The Free Software Movement emerged from Stallman's effort to protect his civil libertarian beliefs about software by creating a sphere free from corporate oversight or control. Later groups adopted the schema and transposed Stallman's ideas into potent new ideologies. Organizations such as Hacktivismo adopted the Free Software licensing approach and used it to distribute tools for human rights activism. Other groups, such as the

---

[37] Ibid., 48.

[38] Mark Hensch, "Assange: WikiLeaks's Trump Info No Worse Than Him," The Hill, August 26, 2016, https://thehill.com/blogs/ballot-box/presidential-races/293453-assange-wikileaks-trump-info-no-worse-than-him. (Accessed 2/2/2020.)

[39] "WikiLeaks Leak," Cryptome, January 7, 2007, http://cryptome.org/wikileaks/wikileaks-leak.htm, (accessed January 21, 2020).

Cypherpunks, broadened Stallman's mistrust to include governments. They leveraged Free Software with the intent of changing society as Stallman had, but they sought the creation of an on-line libertarian utopia free from the control of any state or governmental power. Both the Free Software Movement and the Cypherpunks envisioned a future where Free Software fundamentally changed society in a manner consistent with their conceptions of liberty. Their methods and tools were largely the same, and each group sought a digital utopia. However, they differed in one key respect: their vision of what it meant to achieve binary freedom.

BIBLIOGRAPHY

Primary Sources:

2600.com. “UNIVERSAL CITY v REIMERDES.” May 30, 2001. https://www.2600.com/news/112801-files/universal.html (Accessed April 9, 2018).

Agranoff, Michael H. “Free Software Philosophy Refuted.” *ComputerWorld*, March 4, 1985.

Ahl, David. “Masthead.” *Creative Computing*, 1974.

Ahl, David. *101 BASIC Computer Games*. Maynard, Massachusetts: Digital Equipment Corporation, 1975.

Aleris. “Stereotypes, Marytrdom and growing up.” Slashdot. March 31, 1999. https://linux.slashdot.org/story/99/03/31/1832226/understand-my-job-please-esr-explains. (Accessed January 30, 2020).

Allen, David. “PRAISE FOR PITTMAN’S 6800 TINY BASIC and A Minor Complaint.” *Dr. Dobb*'s *Journal of Tiny BASIC Calisthenics and Orthodontia*, June/July 1976.

Allen, Paul. “March Software Contest.” *Computer Notes*, March 1976.

Allison, Dennis. “Build your own BASIC.” *Dr. Dobb*'s journal of Tiny BASIC Calisthenics and Orthodontia, January 1976.

Allison, Jeremy. "NT password -> smbpassword dump utility." Email to comp.os.ms-windows.nt.admin.security newsgroup. March 24, 1997.

Amateur Computer Users Group Homebrew Computer Club Invitation. Menlo Park California: Homebrew Computer Club, 1975.

Andreessen, Marc. "Introduction to Mosaic Communications Corp." Email to comp.infosystems.www,comp.infosystems,alt.hypertext newsgroup. May 9, 1994.

Anonymous <nowhere@bsu-cs.bsu.edu>. "no subject (file transmission)". Email to cypherpunks@toad.com Mailing List. August 8, 1993.

Anonymous. "martha stuart pgp". Email to wikileaks@wikileaks.org Mailing List. January 7, 2007.

Apache License, Version 1.0. The Apache Group, 1995.

Apple 2 Reference Manual. Cupertino, California: Apple Computer, 1978.

Assange, Julian, Jacob Appelbaum, Andy Müller-Maguhn, and Jérémie1 Zimmermann. *Cypherpunks: Freedom and the Future of the Internet*. New York,New York : OR Books, 2012.

Assange, Julian. "[WL] cryptome disclosure". Email to funtimesahead@lists.riseup.net,wikileaks@wikileaks.org Mailing List. January 8, 2007.

Assange, Julian. "Re: Housewarming/birthday party". Email to lstewart@prisminc.com,cypherpunks@toad.com Mailing List. December 30, 1995.

Assange, Julian. "Re: Risk v. Charity (was: RE: Workers Paradise. /Political rant". Email to craigw@dg.ce.com.au,cypherpunks@toad.com Mailing List. September 17, 1996.
Barlow, John Perry. "Crime and Puzzlement." Electronic Frontier Foundation. June 8, 1990. https://www.eff.org/pages/crime-and-puzzlement. (Accessed January 30, 2020).
Bawden, Alan, Richard Greenblatt, Jack Holloway, Thomas Knight, David Moon, and Daniel Weinreb. Lisp Machine Progress Report. Cambridge, Massachusetts: Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1977.
Berkeley Software Distribution License. Berkeley, California: The Regents of the University of California, 1988.
Berners-Lee, Tim. "New "XMosaic" World-Wide Web browser from NCSA." Email to alt.hypertext,comp.infosystems newsgroup. January 29, 1993.
Berners-Lee, Tim . "WorldWideWeb: Summary", August 6 1991. alt.hypertext. Accessed October 30, 2018.
Bhushan, A. RFC 114: A File Transfer Protocol. Network Working Group, 1971.
Blum, Robert . "The latest TODO list", November 25, 1991. Email to linux-activists@joker.cs.hut.fi.
Blum, Robert . "The new TODO file", December 6, 1991. Email to linux-activists@joker.cs.hut.fi.
Brand, Stewart. "S P A C E W A R: Fanatic Life and Symbolic Death Among the Computer Bums," December 7, 1972. " http://www.wheels.org/spacewar/stone/rolling_stone.html (Accessed November 17, 2017.)
Bray, James. "Gnu and you." Email to net.unix-wizards,net.usoft mailing list. October 18, 1983.
Brian Kernighan, interviewed by Michael Mahoney, Murray Hill, New Jersey, April 17, 1989. https://www.princeton.edu/~hos/mike/transcripts/kernighan.htm (Accessed March 20, 2018).
Brown, Laird. "Thoughts On Hacktivism from the CDC." *The Register*. April 19, 2002. https://www.theregister.co.uk/2002/04/19/waging_peace_on_the_internet/. (Accessed January 30, 2020).
Bush, Vannevar. "As We May Think." *Atlantic*, July 1 1945." http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/. (Accessed March 23, 2015.)
Byte. "Introducing Apple II". June 1977.
Byte. "The Computer for the Professional." July 1978.
Byte. "World Altair Computer Convention" Advertisement. March 1976.
Carruth, Rusty. "Phoenix Linux User's Group," August 6, 1994. comp.os.linux.announce.
Carter, Joseph. "Re: It's Time to Talk About Free Software Again." Email to debian-devel@lists.debian.org mailing list. February 18, 1999.
Cerf, Vinton, and Robert Kahn. "A Protocol for Packet Network Intercommunication." IEEE Transactions on Communications, May 5, 1974.

CERN. "Public Domain Document of 30 April 1993: Page 1." April 30, 1993." https://tenyears-www.web.cern.ch/tenyears-www/Declaration/Page1.html. (Accessed November 3, 2017.)

Chan, Sharon. “Long Antitrust Saga Ends for Microsoft,” *Seattle Times*, May 11, 2011.

Chappell, Bill. “Steve Jobs Dishes On the Tech Business in 'lost interview' from 1995.” National Public Radio. November 16, 2011. https://www.npr.org/sections/alltechconsidered/2011/11/16/142373973/steve-jobs-dishes-on-the-tech-business-in-lost-interview-from-1995. (Accessed January 30, 2020).

Chaum, David. “Blind Signatures for Untraceable Payments.” *Advances in Cryptology Proceedings* 82, no. 3 (1983).

CNN. “Greenspan Warns On Rates.” February 17, 2000. https://money.cnn.com/2000/02/17/economy/greenspan/. (Accessed January 30, 2020).

Congress.gov. “H.R.6933 - An act to amend the patent and trademark laws.” December 12, 1980. https://www.congress.gov/bill/96th-congress/house-bill/6933 (Accessed April 16, 2018).

Costello, Sam. “Sklyarov granted bail, while protests continue.” *ComputerWorld*. August 7, 2001. https://www.computerworld.com.au/article/41340/sklyarov_granted_bail_while_protests_continue/ (accessed April 9, 2018).

Crocker, Steve. Network Working Group Request for Comment: 1. Santa Barbara: University of California, 1969.

Cryptoanarchy.wiki. “Cypherpunk Mailing List Authors by Number of Posts (Highest First).” Accessed August 12, 2019. https://mailing-list-archive.cryptoanarchy.wiki/authors/by-posts/.

Cryptome. "Cryptomb 1." https://cryptome.org/jya/cryptomb1.htm. (Accessed August 12, 2019.)

Cryptome. “WikiLeaks Leak.” January 7, 2007. http://cryptome.org/wikileaks/wikileaks-leak.htm. (accessed January 30, 2020).

Cryptome.org. “WikiLeaks Mailing List Leak.” Last modified February 21, 2007. https://cryptome.org/wikileaks/wikileaks-leak2.htm (accessed April 9, 2018).

Dai, Wei. "PipeNet 1.1 and b-money". Email to cypherpunks@cyberpass.net Mailing List. November 27, 1998.

Dai, Wei. “B-Money.” Wei Dai. http://www.weidai.com/bmoney.txt (accessed April 9, 2018).

DiBona, Chris, Sam Ockman, and Mark Stone, eds. *Open Sources: Voices from the Open Source Revolution.* Beijing: O’Reilly, 1999.

Diffie, Whitfield, and Martin Hellman. “New Directions in Cryptography.” *IEEE Transactions on Information Theory*, 22, no. 6 (November 1976).

Digital Law Online. “National Commission On New Technological Uses of Copyrighted Works (CONTU) Report.” September 28, 2003. Accessed March 14, 2015. http://digital-law-online.info/CONTU/.

Dobson, Michael . "Re: lpdemon, init, shutdown, uucp", January 8, 1992. Email to linux-activists@joker.cs.hut.fi.

Doug McIlroy, interviewed by Michael Mahoney, Murray Hill, NJ, August 18, 1989. https://www.princeton.edu/~hos/mike/transcripts/mcilroy.htm (accessed March 20, 2018).

DVD Copy Control Association. “Content Scramble System (Css).” Accessed April 30, 2019. http://www.dvdcca.org/css.aspx.

Dyson, Esther, George Gilder, George Keyworth, and Alvin Toffler. "Cyberspace and the American Dream: A Magna Carta for the Knowledge Age." The Progress and Freedom Foundation." http://www.pff.org/issues-pubs/futureinsights/fi1.2magnacarta.html. (Accessed May 8, 2019.)

“EFF Effector.” July 27, 2001. https://www.eff.org/effector/14/16. (accessed January 30, 2020).

Emerson, Sandra. "Usenet: A Bulleting Board for Unix Users." *Byte*, October 1983." https://archive.org/stream/byte-magazine-1983-10/1983_10_BYTE_08-10_UNIX#page/n219/mode/2up. (Accessed March 23, 2015.)

Engelbart, Douglas. "Augmenting Human Intellect: A Conceptual Framework." dougengelbart.org. October 1962." http://www.dougengelbart.org/pubs/augment-3906.html. (Accessed March 23, 2015.)

Epic Ventures. "Larry Augustin." http://www.epicvc.com/larry-augustin/. (accessed November 7, 2018.)

Federal Communications Commission. “Telecommunications Act of 1996.” Last modified June 20, 2013. https://www.fcc.gov/general/telecommunications-act-1996. (accessed January 30, 2020).

Felten, Edward. "Response to SDMI Press Release, November 8, 2000." Princeton University." https://sip.cs.princeton.edu/sdmi/nov8.html. (Accessed April 30, 2019.)

Felten, Edward. “The following statement was read by Edward W. Felten at the Fourth International Information Hiding Workshop, in Pittsburgh, on April 26, 2001.” Princeton University. April 26, 2001. https://sip.cs.princeton.edu/sdmi/sdmimessage.txt. (accessed January 30, 2020).

Felten, et al, Edward. "Statement Regarding the SDMI Challenge." Princeton University." http://sip.cs.princeton.edu/sdmi/announcement.html. (Accessed April 30, 2019.)

Felten, et al, Edward. “Reading between the Lines: Lessons from the SDMI Challenge.” Paper presented at the 10th annual meeting USENIX Security Symposium, Washington, D.C., April 15, 2001.

Fillingham, Peter . "NJ computer user's group Linux-related SIG", May 22, 1994. comp.os.linux.announce.

Final report of the National Commission on New Technological Uses of Copyrighted Works, July 31, 1978. Washington: Library of Congress, 1979.

Fluke, Jeff. "emacs from UniPress." Email to net.emacs mailing list. April 20, 1983.

Frenzel, Lou. “How to Choose a Microprocessor.” *Byte* Magazine, July 1978.

Gallaher, Michael. "UniPress and Gnumacs." Email to net.emacs mailing list. August 1, 1985.

Gallant, John. “IBM Policy Draws fire.” *ComputerWorld*, March 18, 1985.

Gates, Bill. “A Second and Final Letter.” *Computer Notes*, April 1976.

Gates, Bill. “An Open Letter to Hobbyists.” Homebrew Computer Club Newsletter, January 31. 1976.

Gates, Bill. “Software Contest Winners Announced.” *Computer Notes*, July 1975.

Gates, William. "The Internet Title Wave Memo." Department of Justice." http://www.justice.gov/atr/cases/exhibits/20.pdf. (Accessed November 3, 2017.)

Gilin, Paul. “Do Users Fear IBM Backlash?” ComputerWorld, May 21, 1984.

Gilmore, John. "Re: gnu copyrights." Email to net.emacs mailing list. May 21, 1986.

Gilmore, John. “Privacy, Technology, and the Open Society.” Lecture, First Conference on Computers, Freedom & Privacy, Burlingame, California, March 26-28, 1991. http://cpsr.org/prevsite/conferences/cfp91/gilmore.html/. (accessed January 30, 2020).

Glass, James. Letter to the Editor. *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985.

GNUnet.org. "About GNUnet." https://gnunet.org/en/about.html. (Accessed August 12, 2019.)

Goerwitz, Richard . "MONEY + Linux", December 17, 1992. Email to linux-activists@joker.cs.hut.fi.

Gosling, James. "Emacs goes commercial.”. Email to net.emacs mailing list. April 9, 1983.

Gosling, James. "Re: Permission." Email to net.emacs mailing list. June 21, 1985.

Gosling, James. "Re: public domain destruction." Email to net.emacs mailing list. April 21, 1983.

Gosling, James. "Software Sabotage." Email to net.emacs mailing list. April 21, 1983.

Gosling, James. *Unix Emacs*. Pittsburg, Pennsylvania: James Gosling, 1981.

GovTrack. “H.R. 2281 (105th): Digital Millennium Copyright Act.” October 12, 1998. https://www.govtrack.us/congress/bills/105/hr2281/text (accessed April 9, 2018).

GovTrack. “H.R. 4262 (100th): Berne Convention Implementation Act of 1988.” October 31, 1988. https://www.govtrack.us/congress/bills/100/hr4262/text/enr (accessed April 9, 2018).

Graetz, J. M.”SPACEWAR! REAL-TIME CAPABILITY OF THE PDP-1.” *DECUS Proceedings 1962*. Maynard, Massachusetts: Digital Equipment Corporation, 1963.

Grossman, Lev. "New Free License to Cover Content Online." Time, September 8, 1998." "http://web.archive.org/web/20000619122406/http://www.time.com/time/digital/daily/0,2822,621,00.html." (Accessed March 14, 2015.)

HackSDMI.org. “Click-Through Agreement for the SDMI Public Challenge.” Last modified October 8, 2000. https://sip.cs.princeton.edu/sdmi/clickthru.pdf. (accessed January 30, 2020).

Hacktivismo. “The Hacktivismo Declaration.” July 4, 2001. http://www.hacktivismo.com/public/declarations/en.php. (accessed January 30, 2020).

Hacktivismo. “The Hacktivismo Enhanced-Source Software License Agreement.” November 25, 2002. http://www.hacktivismo.com/about/hessla.php. (accessed January 30, 2020).

Harlan, Jim. Letter to the Editor, Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia, March 1985.
hart@cp1.UUCP. "Whats GNU with you?" Email to net.unix-wizards,net.usoft mailing list. October 11, 1983
Heinanen, Juha. "Re: Permission." Email to net.emacs mailing list. June 25, 1985.
Holmgren, Steve. “RFC 681: NETWORK UNIX.” Internet Engineering Task Force. May 14, 1975. https://tools.ietf.org/html/rfc681 (accessed March 20, 2018).
Holmgren, Steve. The Network Unix System. Champaign, Il: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1975.
Hubbard, Jordan. "Re: Future of the `Open Source' trademark ". Email to undisclosed-recipients. November 24, 1998.
Hughes, Eric. "A Cypherpunk's Manifesto," Activism.Net, March 9, 1993, " http://www.activism.net/Cypherpunk/manifesto.html. (Accessed April 20, 2015.)
Hughes, Eric. "A statement of purpose". Email to cypherpunks@toad.com Mailing List. October 5, 1992.
Hughes, Eric. "Meeting Sat. Oct. 10, noon, Mt. View". Email to cypherpunks@toad.com Mailing List. October 5, 1992.
IBM.com. “IBM Launches Biggest Linux Lineup Ever.” March 2, 1999. https://www-03.ibm.com/press/us/en/pressrelease/2262.wss, (accessed January 21, 2020).
IDC, Inc. "Android and iOS Squeeze the Competition, Swelling to 96.3% of the Smartphone Operating System Market." February 24, 2015." http://www.idc.com/getdoc.jsp?containerId=prUS25450615. (Accessed May 13, 2015.)
Internet Engineering Task Force. “Request for Comments (RFC).” Accessed November 3, 2017. http://www.ietf.org/rfc.html.
Internet Systems Consortium. “Domain Server Software Distribution.” February 7, 2017. https://ftp.isc.org/www/survey/reports/2017/01/fpdns.txt. (accessed January 30, 2020).
Johnson, Michael . "Re: MONEY + Linux", December 17, 1992. Email to linux-activists@joker.cs.hut.fi. .
JUSTIA US Law. “Dann v. Johnston, 425 U.S. 219 (1976).” March 31, 1976. https://supreme.justia.com/cases/federal/us/425/219/case.html (accessed April 16, 2018).
JUSTIA US Law. “Gottschalk v. Benson, 409 U.S. 63 (1972).” November 20, 1972. https://supreme.justia.com/cases/federal/us/409/63/case.html (accessed April 16, 2018).
JUSTIA US Law. “Lewis Galoob Toys, Inc., Plaintiff-Appellee, V. Nintendo of America, Inc., Defendant-Appellant.nintendo of America, Inc., Plaintiff-Appellant, V. Lewis Galoob Toys, Inc., Defendant-Appellee, 964 F.2d 965 (9th Cir. 1992).” http://law.justia.com/cases/federal/appellate-courts/F2/964/965/341457/. (Accessed November 3, 2017.)
JUSTIA US Law. “United States v. American Tel. and Tel. Co., 552 F. Supp. 131.” February 28, 1983. https://law.justia.com/cases/federal/district-courts/FSupp/552/131/1525975/ (accessed April 17, 2018).

JUSTIA US Law. “White-Smith Music Pub. Co. v. Apollo Co., 209 U.S. 1 (1908).” February 24, 1908. https://supreme.justia.com/cases/federal/us/209/1/ (accessed April 16, 2018).
Kawasaki, Guy. The Macintosh Way. Glenview, Illinois.: Scott Foresman Trade, 1990.
Kemeny, John G. *Man and the Computer*. New York, New York: Scribner, 1972.
Kemeny, John, and Thomas Kurtz. *The Dartmouth Time-Sharing Computing System - Final Report.* Hanover, New Hampshire: Dartmouth College, 1967.
Ken Thompson, interviewed by Michael Mahoney, Murray Hill, New Jersey, September 6, 1989. https://www.princeton.edu/~hos/mike/transcripts/thompson.htm (accessed March 20, 2018).
Klosowski, Przemek . "First meeting of metropolitan DC Linux User group", June 7, 1994. comp.os.linux.announce.
Kornblum, Janet. “Netscape sets source code free.” C|Net. March 31, 1998. https://www.cnet.com/news/netscape-sets-source-code-free/ (accessed April 9, 2018).
Labalme, Fen. "Whither GNU EMACS ?" Email to net.emacs mailing list. June 26, 1985.
Lagua, Vir . "Salt Lake Linux Users Group", July 17, 1994. comp.os.linux.announce.
law.cornell.edu. "Copyright Act 1976." http://www.law.cornell.edu/copyright/copyright.table.html. (Accessed November 3, 2017.)
Lea, Graham. "MS' Ballmer: Linux Is Communism." *The Register*. July 31, 2000." http://www.theregister.co.uk/2000/07/31/ms_ballmer_linux_is_communism/. (Accessed March 23, 2015.)
Leinwand Leger, Donna. “How FBI brought down cyber-underworld site Silk Road.” *USA Today*. October 21, 2013. https://www.usatoday.com/story/news/nation/2013/10/21/fbi-cracks-silk-road/2984921/ (accessed April 18, 2018).
Licklider, J.C.R. “Memorandum for Members and Affiliates of the Intergalactic Computer Network.” Kurzweil Accelerating Intelligence. Last modified December 11, 2001. https://www.kurzweilai.net/memorandum-for-members-and-affiliates-of-the-intergalactic-computer-network, (accessed January 21, 2020).
Lindahl, Greg. "Re: Perl may be great, but..." Email to alt.religion.computers mailing list. November 27, 1989.
Lovelace, Ada King. Ada, the Enchantress of Numbers: A Selection from the Letters of Lord Byron's Daughter and Her Description of the First Computer. Edited by Betty Alexandra Toole. Mill Valley, California: Strawberry Press, 1992.
LWN.net.” Digital Millennium Copyright Act Protest at Stanford University.” Last modified May 12, 2000. http://lwn.net/2000/features/DMCA/announcement.html. (accessed January 30, 2020).
MacDonald, Peter . "Re: MONEY + Linux", December 18, 1992. Email to linux-activists@joker.cs.hut.fi. .
Magid, Jonathan . "Re: MONEY + Linux", December 17, 1992. Email to linux-activists@joker.cs.hut.fi.
Markwalder, Phillip . "FIRST MEETING LINUX User Group Switzerland (LUGS)", June 10, 1994. comp.os.linux.announce.

Martindale, David . "GNU considered harmful to software quality" Email to net.unix-wizards,net.usoft mailing list. October 8, 1983.
Martinson, Yanek . "lpdemon, init, shutdown, uucp", January 7, 1992. Email to linux-activists@joker.cs.hut.fi.
Massachusetts Institute of Technology License. Cambridge, Massachusetts: Massachusetts Institute of Technology, 1988. https://opensource.org/licenses/MIT. (accessed January 30, 2020).
May, Tim. "The Cyphernomicon." September 11, 1994. https://ia600208.us.archive.org/10/items/cyphernomicon/cyphernomicon.txt. (accessed January 30, 2020).
May, Timothy. "BlackNet Investigations--the Truth (fwd)". Email to cypherpunks@toad.com Mailing List. October 1, 1993.
May, Timothy. "Libertaria in Cyberspace". Email to Extropians@gnu.ai.mit.edu Mailing List. September 1, 1992.
May, Timothy. "The Crypto Anarchist Manifesto". Email to cypherpunks@toad.com Mailing List. November 22, 1992.
Maynard, James. "Re: Perl may be great, but..." Email to alt.religion.computers mailing list. November 24, 1989.
McEnaney, Maura. "All Software Should Be Free, Software Developer Maintains." *ComputerWorld*, December 24, 1984.
McIlroy, Doug. A Research UNIX Reader. Murray Hill, New Jersey: Bell Labs, 1986.
McKeon, Siobahn. "Wordpress." WordPress History, Chapter 3." https://wordpress.org/about/history/chapter3.pdf. (Accessed April 13, 2019.)
Mellody, Patrick. "Statement of Business Software Alliance President and CEO Robert Holleyman Concerning the First Indictment and Arraignment under the Dmca." Business Software Alliance. August 30, 2001. http://web.archive.org/web/20010917030022/http://www.bsa.org/usa/press/newsreleases/2001-08-30.692.phtml. (accessed January 30, 2020).
Microsoft.com. "Applies To: Internet Information Services 8.0microsoft Internet Information Services 8.0 Express." October 20, 2017. https://support.microsoft.com/en-us/help/224609/how-to-obtain-versions-of-internet-information-server-iis. (accessed January 30, 2020).
MITS Christmas Catalog 1975. Albuquerque, New Mexico: MITS Corporation, 1975.
Mockapetris, P. RFC 882: DOMAIN NAMES - CONCEPTS and FACILITIES. Network Working Group, 1983.
Montgomery, Warren. "btl emacs" Email to net.emacs mailing list. January 19, 1983.
Mueller, Robert. *Report On the Investigation Into Russian Interference in the 2016 Presidential Election*. Washington, DC: U.S. Department of Justice, 2019.
Murdock, Ian. "The Debian Manifesto." Debian.org. Last modified January 6, 1994. https://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html. (accessed January 30, 2020).
Murphy, Thomas. "Case Against Free Software." *ComputerWorld*, January 21,1985.
Nakamoto, Satoshi. "Bitcoin P2P e-cash paper". Email to cryptography@metzdowd.com Mailing List. November 6, 2008.

Nakamoto, Satoshi. "Bitcoin P2P e-cash paper". Email to cryptography@metzdowd.com Mailing List. October 31, 2008.

Nakamoto, Satoshi. "Bitcoin Open Source Implementation of P2p Currency." Nakamoto Institute. February 11, 2009. https://satoshi.nakamotoinstitute.org/posts/p2pfoundation/1/#selection-9.0-9.50. (accessed January 30, 2020).

Nakamoto, Satoshi. "Bitcoin: A Peer-to-peer Electronic Cash System." bitcoin.org. October 31, 2008. https://bitcoin.org/bitcoin.pdf. (accessed January 30, 2020).

National Center for Supercomputing Applications at the University of Illinois at Urbana - Champaign, Illinois. "NSCA HTTPD." January 21, 1998. https://web.archive.org/web/20090415224404/http://hoohoo.ncsa.uiuc.edu/ (accessed April 9, 2018).

Nelson, Russ . "Networking.", January 9, 1992. Email to linux-activists@joker.cs.hut.fi.

Netcraft. "Netcraft Web Server Survey." April 02, 2014." http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html. (Accessed November 3, 2017.)

Netcraft. "Site Report for Www.Google.com." Last modified March 31, 2019. https://toolbar.netcraft.com/site_report?url=http://www.Google.com. (accessed January 30, 2020).

Newman, Elsa. "PAPERS AND PRESENTATIONS of The Digital Equipment Computer Users Society." *DECUS Proceedings 1962*. Maynard, Massachusetts: Digital Equipment Corporation, 1963.

Open Office. "Major Openoffice.org Deployments." Last modified July 2, 2014." https://wiki.openoffice.org/wiki/Major_OpenOffice.org_Deployments. (Accessed March 14, 2015.)

Openwatcom. "Sybase Open Watcom Public License Version 1.0." http://www.openwatcom.org/index.php/Open_Watcom_Public_License. (Accessed May 13, 2015.)

Oppenheim, Esq., Matthew J. "RIAA/SDMI Letter, April 9, 2001." Princeton University. April 9, 2001. https://sip.cs.princeton.edu/sdmi/riaaletter.html. (accessed January 30, 2020).

O'Rourke, Beto. "A Feature On Money - Today's Monster." Cult of the Dead Cow." http://textfiles.com/groups/CDC/cDc-0031.txt. (Accessed July 3, 2019.)

Pack, Charles L. "Response to Bill Gates." Homebrew Computer Club Newsletter, March 31, 1976.

PDP-11 Decnet-11m, Phase II Brochure. Maynard, Massachusetts: Digital Equipment Corporation, 1978.

Perens, Bruce, Wichert Akkerman, and Ian Jackson. "The Apple Public Source License - Our Concerns." *Linux Today*, March 17, 1999. https://www.linuxtoday.com/developer/1999031700205NWLF. (accessed January 30, 2020).

Perens, Bruce. "Email Threat." lists.debian.org. April 5, 1999." https://lists.debian.org/debian-user/1999/04/msg00623.html. (Accessed March 14, 2015.)

Perens, Bruce. "It's Time to Talk About Free Software Again". Email to debian-devel@lists.debian.org mailing list. February 17, 1999.
Perens, Bruce. "The Open Source Definition." The Open Source Initiative." http://opensource.org/osd. (Accessed November 3, 2017.)
Perens, Bruce. “Bruce Perens Replies to ESR's Letter.” Free(code). April 6, 1999. http://freshmeat.sourceforge.net/articles/bruce-perens-replies-to-esrs-letter. (accessed January 30, 2020).
Perens, Bruce. “Debian Social Contract.” Debian.org. July 5, 1997. https://www.debian.org/social_contract.1.0. (accessed January 30, 2020).
Perens, Bruce. “Do We Need a New Evangelist?” Perens.com. Accessed December 20, 2018. www.perens.com/Articles/Evangelist.html.
Perens, Bruce. “Email Threat.” lists.debian.org. April 5, 1999. Accessed March 14, 2015. https://lists.debian.org/debian-user/1999/04/msg00623.html.
Perens, Bruce. “It's Time to Talk About Free Software Again.” lists.debian.org. February 17, 1999. Accessed March 14, 2015. https://lists.debian.org/debian-devel/1999/02/msg01641.html.
Perens, Bruce.” The Apple Public Source License - Our Concerns.” *Linux Today*.May 17, 1999. http://www.linuxtoday.com/developer/1999031700205NWLF. (accessed March 14, 2015.)
Perens, Bruce. “The Open Source Definition.” The Open Source Initiative. Accessed November 3, 2017. http://opensource.org/osd.
Pfaff, Ben. "Re: It's Time to Talk About Free Software Again". Email to debian-devel@lists.debian.org mailing list. February 17, 1999.
Plant, Emmett. “The Welcome Wagon.” *Linux Today*. November 9, 1999. https://www.linuxtoday.com/developer/1999110900105NWSM. (accessed January 30, 2020).
Postel, J. RFC 801: NCP/TCP Transition Plan.Network Working Group, 1981.
Rand, Ayn. *Atlas Shrugged*. 35th ed. New York, New York: Signet, 1992, 1985, https://ia800305.us.archive.org/19/items/AtlasShrugged/atlas%20shrugged.pdf. (accessed January 21, 2020.)
Raucci, Richard. A Windows NT Guide to the Web. New York, New York: Springer New York, 1997.
Raymond, Eric "Goodbye, 'Free Software'; hello, 'Open Source'," catb.org, November 22, 1998." http://www.catb.org/esr/open-source.html. (Accessed April 20, 2015.)
Raymond, Eric "Why I Hate Proprietary Software," http://esr.ibiblio.org, October 1, 2008." http://esr.ibiblio.org/?p=556. (Accessed May 13, 2015.)
Raymond, Eric. *The Cathedral and the Bazaar,* Beijing, 2001.
Raymond, Eric. "A Brief History of Hackerdom." catb.org. May 5, 2000." http://www.catb.org/~esr/writings/hacker-history/hacker-history.html. (Accessed May 13, 2015.)
Raymond, Eric. "How to Learn Hacking." catb.org. November 30, 2014." http://www.catb.org/esr/faqs/hacking-howto.html. (Accessed May 13, 2015.)
Raymond, Eric.” Open Source Summit.” *Linux Journal*. June 1, 1998. https://www.linuxjournal.com/article/2918. (accessed January 30, 2020).

Raymond, Eric. "Resume of Eric Steven Raymond." catb.org. January 29, 2003." http://www.catb.org/esr/resume.html. (Accessed May 13, 2015.)
Raymond, Eric. "Revenge of the Hackers," catb.org, August 31, 1999." http://www.catb.org/~esr/writings/cathedral-bazaar/hacker-revenge/. (Accessed April 20, 2015.)
Raymond, Eric. "Surprised by Wealth." *Linux Today*. December 10, 1999." http://www.linuxtoday.com/infrastructure/1999121000105NWLF. (Accessed March 14, 2015.)
Raymond, Eric. "The New Hacker's Dictionary, Third Edition," MIT Press, " http://mitpress.mit.edu/books/new-hackers-dictionary. (Accessed April 20, 2015.)
Raymond, Eric. "The Open Source Initiative needs your help". Email to comp.os.linux.announce newsgroup. December 3, 1998.
Raymond, Eric. "Why We Fight "" an Anti-Idiotarian Manifesto." catb.org. December 26, 2003." http://www.catb.org/esr/aim/. (Accessed May 13, 2015.)
Raymond, Eric. "Homesteading the Noosphere." catb.org. April 10, 1998. http://www.catb.org/~esr/writings/homesteading/homesteading/index.html. (accessed January 30, 2020).
Raymond, Eric. "How to Become a Hacker." catb.org. October 4, 2007. http://www.catb.org/esr/faqs/hacker-howto.html#history. (accessed January 30, 2020).
Raymond, Eric. "OSI clarifies the status of the APSL." *Linux Today*, March 19, 1999. https://www.linuxtoday.com/developer/1999031900405NWLF. (accessed January 30, 2020).
Raymond, Eric. "Shut up and Show Them the Code." *Linux Today*, June 28, 1999. https://www.linuxtoday.com/developer/1999062802310NWSM. (accessed January 30, 2020).
Raymond, Eric. "Take My Job, Please!" catb.org. March 29, 1999. http://www.catb.org/esr/writings/take-my-job-please.html. (accessed January 30, 2020).
Raymond, Eric. "The Cathedral and the Bazaar." catb.org. August 16, 1999. http://www.catb.org/~esr/writings/cathedral-bazaar/afterword/. (accessed January 30, 2020).
Raymond, Eric. "The Cathedral and the Bazaar: Revision History." catb.org. Last modified February 8, 2002. http://www.catb.org/esr/writings/cathedral-bazaar/cathedral-bazaar/index.html#catbmain. (accessed January 30, 2020).
Raymond, Eric. "Understand My Job, Please!" catb.org. March 31, 1999. http://www.catb.org/esr/writings/understand-my-job-please.html. (accessed January 30, 2020).
Raymond, Eric. Open Source Software: A (New?) Development Methodology Annotations. Malvern, Pennsylvania, 1998. https://www.gnu.org/software/fsfe/projects/ms-vs-eu/halloween1.html#_Toc427495764. (accessed January 30, 2020).
Reimer, Jeremy. "Total Share the 8-Bit Era: 30 Years of Personal Computer Market Share Figures." ARS Technica. December 15, 2005.

https://arstechnica.com/features/2005/12/total-share/4/. (accessed January 30, 2020).
Richter, Adam . "ANNOUNCEMENT: Alpha release Linux/GNU/X unix clone on CDROM for PC's", November 25, 1992. alt.os.linux,comp.unix.pc-clone.32bit,comp.unix.wizards,. Accessed November 05, 2018.
Ritchie, Dennis, and Ken Thompson. "The Unix Time- Sharing System." Paper presented at the Fourth ACM Symposium on Operating Systems Principles, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, New York, October 15-17, 1973. http://www.cs.berkeley.edu/~brewer/cs262/unix.pdf. (Accessed May 13, 2015.)
Ritchie, Dennis. "Space Travel: Exploring the Solar System and the PDP-7." cs.bell-labs.com. 2001." http://www.cs.bell-labs.com/who/dmr/spacetravel.html. (Accessed March 14, 2015.)
Roberts, H. Edward, and William Yates. "ALTAIR 8800 The most powerful minicomputer project ever presented - can be built for under $400." Popular Electronics, January 1975.
Rochkind, Marc. "The Source Code Control System." IEEE Transactions on Software Engineering 1, no. 4 (December 1975): 364-70.
Surprised by Wealth Comments." *Linux Today*. December 10, 1999." http://www.linuxtoday.com/infrastructure/1999121000105NWLF#comment_container. (Accessed March 14, 2015.)
Sasaki, Marty. "gnu emacs on VMS." Email to net.emacs mailing list. June 3, 1985.
Schwartz, Robert. Letter to the Editor, *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985.
Shankland, Stephen. "'Open Source' Infighting Grows." C|net. February 19, 1999." http://news.cnet.com/Open-source-infighting-grows/2100-1001_3-221918.html. (Accessed March 14, 2015.)
SHARE PROGRAM LIBRARY AGENCY User's Guide and Catalog of Programs. Research Triangle Park, North Carolina: Triangle Universities Computation Center, 1977.
Shaw, Rohit."Weapon of Anonymous," Infosec Institute, August 29, 2013." http://resources.infosecinstitute.com/weapon-of-anonymous/. (Accessed April 20, 2015.)
Slettnes, Tor. "Re: It's Time to Talk About Free Software Again". Email to debian-devel@lists.debian.org mailing list. February 17, 1999.
Smith, Michael. "DC DMCA Protest". Email to ruben@sruben.dental.nyu.edu,nylug-talk@nylug.org Mailing List. May 1, 2000.
SOFTWARE AGREEMENT between WESTERN ELECTRIC COMPANY, INCORPORATED and [REDACTED]. New York, New York : WESTERN ELECTRIC COMPANY, 1973.
Spanbauer, Rick . ""Whats GNU with you?"" Email to net.unix-wizards,net.usoft mailing list. October 5, 1983.
Stallman, Richard. "Apple's Non-Free Source License." *Linux Today*. March 22, 1999. https://www.linuxtoday.com/infrastructure/1999032200105NWLF. (accessed January 30, 2020).

Stallman, Richard. "A Serious Bio." Stallman.org." https://stallman.org/biographies.html. (Accessed March 14, 2015.)

Stallman, Richard. “Free Software: Freedom and Cooperation.”, Speech, New York, New York, May 29, 2001, GNU.org. https://www.gnu.org/events/rms-nyu-2001-transcript.txt. (accessed January 30, 2020).

Stallman, Richard. *Free Software, Free Society: Selected Essays of Richard M. Stallman.* Boston, Massachusetts: Free Software Foundation, 2002.

Stallman, Richard. "GNU and Minix", October 17, 1990. comp.os.minix.

Stallman, Richard. "GNU Emacs." Email to net.emacs mailing list. June 27, 1985.

Stallman, Richard. Gnu Emacs Availability Information. Cambridge, Massachusetts: GNU Project, 1985.

Stallman, Richard. Gnu General Public License: Version 1. Cambridge, Massachusetts: Free Software Foundation, 1989.

Stallman, Richard. "GNU kernel status", November 11, 1990. gnu.announce.

Stallman, Richard. "GNU Manifesto." April 12, 2014." https://www.gnu.org/gnu/manifesto.html. (Accessed November 3, 2017.)

Stallman, Richard. “GNU Operating System and the Free Software Movement” in *Open Sources: Voices from the Open Source Revolution.* Ed. DiBona, Chris, Ockman, Sam and Stone, Mark. Beijing: O’Reilly. 1999.

Stallman, Richard. “GNU Public License.” Free Software Foundation. April 1, 1989. https://www.gnu.org/licenses/gpl-1.0.html (accessed November 5, 2017).

Stallman, Richard. "honor." Email to net.emacs mailing list. June 18, 1985.

Stallman, Richard. "Independence Day." Email to net.emacs mailing list. July 5, 1985.

Stallman, Richard. “Initial Announcement (Gnu Project).” GNU. April 2, 2014. Accessed November 3, 2017. https://www.gnu.org/gnu/initial-announcement.html.

Stallman, Richard. "Latest draft of library license." Email to gnu.g++.help,comp.lang.c++. mailing lists. January 5, 1991.

Stallman, Richard. “My Lisp Experiences and the Development of Gnu Emacs.” Speech, San Francisco, California, October 28, 2002 GNU.org. https://www.gnu.org/gnu/rms-lisp.html. (accessed January 30, 2020).

Stallman, Richard. "New Documentation License--Comments Requested". Email to Newsgroup gnu.announce,comp.os.linux.misc,gnu.misc.discuss. September 12, 1999.

Stallman, Richard. "new Unix implementation" Email to net.unix-wizards,net.usoft mailing list.. September 27, 1983.

Stallman, Richard. "Re: Evaluation of Gcompris". Email to vtamara@gnu.org. December 18, 2000.

Stallman, Richard. "Richard Stallman's Personal Site." stallman.org." https://stallman.org/. (Accessed May 13, 2015.)

Stallman, Richard. "Software Sabotage." Email to net.emacs mailing list. April 17, 1983.

Stallman, Richard. "Whither GNU EMACS ?" Email to net.emacs mailing list. June 26, 1985.

Stallman, Richard. “The Gnu Manifesto.” *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985.

Stallman, Richard. “Why “Free Software” is better than “Open Source”.” GNU.org. https://www.gnu.org/philosophy/free-software-for-freedom.html (accessed April 9, 2018).
Stephenson, Kim . "NCSA Mosaic for the Macintosh 1.0",. Email to comp.sys.mac.comm. November 11, 1993.
Systems Network Architecture General Information. Research Triangle Park: IBM Systems Development Division, 1975.
Szoldra, Paul. "WikiLeaks Just Released a Massive 'insurance' File That No One Can Open" " http://www.businessinsider.com/wikileaks-insurance-file-2013-8. (Accessed March 14, 2015.)
Templeton, Brad . "GNU considered harmful to software quality." October 7, 1983. net.unix-wizards,net.usoft newsgroup.
Terbush, Randy . "Announcing: Apache-1.0 Release", December 4, 1995. comp.infosystems.www.servers.unix. Accessed November 05, 2018.
Terry, Douglas, Mark Painter, David Riggle, and Songnian Zhou. The Berkeley Internet Name Domain Server. Berkeley, California: The Regents of the University of California, 1984.
The BSD License. Berkeley, California: Regents of the University of California. 1988.
The Free Software Foundation. “The Free Software Definition.” Last modified June 12, 2018. https://www.gnu.org/philosophy/free-sw.html. (accessed January 30, 2020).
The Pet Personal Computer Brochure. London, England, Commodore International, Ltd, 1978.
Theodore Tso. "Re: No FTP ... HELP!", January 20, 1992. Email to linux-activists@joker.cs.hut.fi. .
Thomas, Robert. "(Australia) Sydney Linux users Group", May 13, 1993. comp.os.linux.announce.
Thompson, Ken, and Dennis Ritchie. *UNIX programmer’s Manual, 2nd Edition*. Murray Hill, New Jersey: Bell Labs, 1972.
Tichy, Walter. “RCS—a System for Version Control.” Paper presented at the IEEE Computer Society 6th international conference on Software engineering, Los Alamitos, California, September 13-16, 1982.
Time. “Machine of the Year”. January 3, 1983.
Torvalds, Linus . "Free minix-like kernel sources for 386-AT", October 5, 1991. comp.os.minix.
Torvalds, Linus . "Gcc-1.40 and a posix question", July 3, 1991. comp.os.minix. Accessed
Torvalds, Linus . "What would you like to see most in minix?", August 25, 1991. comp.os.inix.
Torvalds, Linus, and David Diamond. Just for Fun: The Story of an Accidental Revolutionary. Reprint ed. New York, New York: HarperBusiness, 2002.
Touretzky, David. “Gallery of CSS Descramblers.” Carnegie Mellon University. Last modified February 13, 2008. https://www.cs.cmu.edu/~dst/DeCSS/Gallery/ (accessed April 9, 2018).

Tower, Leonard . "Linux and the GNU system by Richard Stallman", May 31, 1996. gnu.misc.discuss,comp.os.linux.misc,comp.os.linux.development.apps,comp.os.linux.advocacy.
Tower, Leonard. "Another GNU software distribution." Email to net.emacs mailing list. August 5, 1985.
Tower, Leonard. "New General Public License." Email to gnu.announce mailing list. February 25, 1989.
Tridgell, Andrew. "announcing file services from non-DEC machines". Email to vmsnet.networks.desktop.pathworks newsgroup. January 10, 1992.
TRS-80 Microcomputer System. Boston, Massachusetts: Radio Shack, 1977.
Tso, Theodore . "Re: No FTP ... HELP!", January 20, 1992. Email to linux-activists@joker.cs.hut.fi. .
Turner, Michael . "GNU considered harmful to softwa -." Email to net.unix-wizards,net.usoft mailing list. October 9, 1983.
Turner, Scott. "Re: gnu copyrights." Email to net.emacs mailing list. May 23, 1986.
U.S. Congress. House. Communications Assistance for Law Enforcement Act. H. Res. 4922. 103d Cong., 1st sess., (October 7, 1994).
U.S. Congress. Senate. Comprehensive Counter-Terrorism Act of 1991. S. S.266. 102d Cong., 1st sess., (January 24, 1991): https://www.congress.gov/bill/102nd-congress/senate-bill/266/text. (accessed January 30, 2020).
United States Copyright Office. "Digital Millennium Copyright Act." http://www.copyright.gov/legislation/hr2281.pdf. (Accessed November 3, 2017.)
UNIX System V and Add on Applications Pricing. Summit, New Jersey: AT&T, 1984.
VA Linux Systems Unveils Sourceforge, an Application Service Provider for Open Source Projects." January 4, 1999." http://www.prnewswire.com/news-releases/va-linux-systems-unveils-sourceforgetm-an-application-service-provider-for-open-source-projects-71926707.html. (Accessed March 14, 2015.)
Valloppillil, Vinod. “Open Source Software.” Redmond, Washington: Microsoft Corporation, 1998.
Vir Lagua. "Salt Lake Linux Users Group", July 17, 1994. comp.os.linux.announce.
Volkerding, Patrick . "ANNOUNCE: Slackware Linux 1.00", July 17, 1993. Email to linux-activists@joker.cs.hut.fi.
Wada, Robert. “AN OPINION ON SOFTWARE MARKETING.” *Byte*, July 1976.
Wales, Jimmy. "Re: [Bug-gnupedia] gnupedia.org resolves to nupedia". Email to Newsgroup Bug-gnupedia. January 21, 2001.
Wang, Li-Chen. “TINY BASIC FOR INTEL 8080.” *Dr. Dobb*'s journal of Tiny BASIC Calisthenics and Orthodontia, May 1976.
WarGames, directed by John Badham. United Artists,1983. DVD. (MGM/UA Entertainment, 2008).
Warren, Jim. “COPYRIGHT MANIA: It’s mine; it’s mine, and you can’t play with it!.” *Dr. Dobb*'s journal of Tiny BASIC Calisthenics and Orthodontia, May 1976.
Warren, Jim. “First Word On a Floppy-Disc Operating System.” *Dr. Dobb*'s journal of Tiny BASIC Calisthenics and Orthodontia, April 1976.
Warren, Jim. “History Repeats Itself... I hope.” *Dr. Dobb*'s journal of Tiny BASIC Calisthenics and Orthodontia, April 1976.

Warren, Jim. “Response to Bill Gates Open letter to Hobbyists.” People's Computing Company Newsletter, April 10, 1976.
Warren, Jim. The First West Coast Computer Faire Program. Palo Alto, California: Computer Faire, 1977.
Weeks, Rollin V.. Letter to the Editor, *Dr. Dobb's journal of Tiny BASIC Calisthenics and Orthodontia*, March 1985.
Wiegand, Jim . "lp support & daemonology", January 7, 1992. Email to linux-activists@joker.cs.hut.fi.
Wilson, Chris. "NCSA Mosaic for MS Windows 1.0 release notice (Win31- Winsock)." Email to comp.os.ms-windows.announce newsgroup. November 11, 1993.
Windows Refund Day Letter. Foster City: Microsoft, 1999. http://marc.merlins.org/linux/refundday/dear-valued-customer.html. (accessed January 30, 2020).
Wise, T.A. “IBM's $5,000,000,000 Gamble.” Fortune, September 1966.
Woods, Dan. "The Myth of Crowdsourcing." *Fortune Magazine*. September 9, 2009." http://www.forbes.com/2009/09/28/crowdsourcing-enterprise-innovation-technology-cio-network-jargonspy.html. (Accessed May 3, 2016.)
Wordpress. "Democratize Publishing." https://wordpress.org/about/. (Accessed April 13, 2019.)
Wordpress.com. "WikiLeaks." https://mywikikileaks.wordpress.com/. (Accessed April 18, 2016.)
Young, John. "Re: [WL] Funding / who is on this list.". Email to wikileaks@wikileaks.org Mailing List. January 7, 2007.
Young, John. “Wargaming Disclosures.” Cryptome. June 11, 2013. http://cryptome.org/2013/06/wargaming-disclosures.htm. (accessed January 30, 2020).
YouTube. “Anonymous Official.” https://www.youtube.com/channel/UCA071Pllf2wk-B8Rkwt47bQ (accessed April 9, 2018).
Zimmerman, Steven. "honor." Email to net.emacs mailing list. June 24, 1985.
Zimmerman, Steven. "Important notice for EMACS users." Email to net.general,net.unix-wizards,net.emacs mailing lists. June 13, 1983.
Zimmerman, Steven. "New CCA EMACS Distribution." Email to net.emacs mailing list. January 11, 1983.

Secondary Sources:

Abbate, Janet. *Inventing the Internet*. Cambridge, Massachusetts: MIT Press, 1999.
Abbate, Janet. *Recoding Gender: Women's Changing Participation in Computing. History of Computing*. Cambridge, Massachusetts: MIT Press, 2012.
Apache.org. “About Apache.” http://httpd.apache.org/ABOUT_APACHE.html. (accessed October 30, 2018.)
Apple Computer. “Open at the Source.” https://www.apple.com/opensource/. (accessed March 14, 2015.)
Arthur, Charles. “Lulzsec: What They Did, Who They Were and How They Were Caught.” *The Guardian*. May 16, 2013.

https://www.theguardian.com/technology/2013/may/16/lulzsec-hacking-fbi-jail. (accessed January 21, 2020).
Assange, Julian. “The World Tomorrow.” WikiLeaks. https://worldtomorrow.wikileaks.org/episode-8.html (accessed November 5, 2017).
Atomic Heritage Foundation. “Atomic Glossary.” June 18, 2014. https://www.atomicheritage.org/history/atomic-glossary. (accessed January 30, 2020).
Auletta, Ken. *Googled: The End of the World as We Know It.* New York, New York: Penguin Press, 2009.
Bandyopadhyay, Avimanyu. “A Timeline of VA Linux: Through the Years.” *Medium*. October 5, 2018. https://medium.com/@avimanyu786/a-timeline-of-va-linux-through-the-years-6813e2bd4b13. (accessed January 21, 2020).
BBC. “Key Reaction to WikiLeaks Cables Revelations,” *BBC News*, November 29, 2010, http://www.bbc.com/news/world-us-canada-11866220. (accessed April 20, 2015.)
Benkler, Yochai.” Coase's Penguin, Or, Linux And.” *The Yale Law Journal* 112, no. 3 (December, 2002): 369-446.
Berger, Matt. “Jury: Elcomsoft Not Guilty.” *NetworkWorld*, December 17, 2002. https://www.networkworld.com/article/2339223/jury--elcomsoft-not-guilty.html. (accessed January 30, 2020).
Beyer, Kurt. *Grace Hopper and the Invention of the Information Age*. Lemelson Center Studies in Invention and Innovation. Cambridge, Massachusetts: MIT Press, 2009.
Booker, Ellis. “Spyglass to Commercialize Future Versions.” *ComputerWorld*, August 29,1994. 16.
Boyle, James. “The Second Enclosure Movement and the Construction of the Public Domain.” *Law and Contemporary Problems*, Winter 2003. Pages?
Brand, Stewart. “Discussions for the Hacker's Conference, 1984.” *Whole Earth Review*, May 1985.
Brand, Stewart.” S P A C E W A R: Fanatic Life and Symbolic Death Among the Computer Bums,” December 7, 1972. *Rolling Stone*. http://www.wheels.org/spacewar/stone/rolling_stone.html(accessed November 17, 2017.)
C2. “Wikiwikiweb FAQ.” March 7, 2014. http://c2.com/cgi/wiki?WikiWikiWebFaq. (accessed March 14, 2015.)
Campbell-Kelly, Martin, William Aspray, Nathan Ensmenger, and Jeffrey R. Yost. *Computer: A History of the Information Machine*. 3rd ed. Boulder, Colorado: Westview Press, 2013.
Campbell-Kelly, Martin. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry.* History of Computing. Cambridge, Massachusetts: MIT Press, 2003.
Ceruzzi, Paul E. *A History of Modern Computing.* 2nd ed. History of Computing. Cambridge, Massachusetts: MIT Press, 2003.

Charles, Dan. "'Innocent' Hackers Want Their Computers Back." *New Scientist*, May 9, 1992. https://www.newscientist.com/article/mg13418201-400-innocent-hackers-want-their-computers-back-/,/ (Accessed 7/16/2020).

Chen, Adrian. "A Discussion with Cryptome." *Gawker*. June 19, 2013. https://gawker.com/a-discussion-with-cryptome-514154708. (accessed January 30, 2020).

Chung, Thomas. "History of Red Hat Linux." Fedora Project. Last modified November 6, 2016. https://fedoraproject.org/wiki/History_of_Red_Hat_Linux. (accessed January 30, 2020).

Coleman, Gabriella. *Hacker, Hoaxer, Whistleblower, Spy: The Many Faces of Anonymous*. London: Verso, 2014.

Coleman, Gabriella. "Code Is Speech." *Reason*. https://reason.com/2013/03/21/code-is-speech. (accessed May 1, 2019.)

Computer History Museum. "Artifact Details: Altair Basic Interpreter Source Tape, Replica." Last modified April 7, 2002. https://www.computerhistory.org/collections/catalog/102631998. (accessed January 27, 2020.)

Cook, John. "Secrets and Lies. *Radar* Magazine, August 13, 2007. http://radaronline.com/from-the-magazine/2007/08/cryptome_john_young_radar_anthony_haden_guest_1.php. (accessed January 30, 2020).

Creative Commons. "History of the Creative Commons." http://creativecommons.org/about/history. (accessed March 14, 2015.)

Cunningham, Ward. "Wiki History." C2.com. Last modified December 22, 2014. http://wiki.c2.com/?WikiHistory. (accessed January 30, 2020).

Davidson, Duncan. "Symposium Common Law, Uncommon Software: The Future of Software Protection." *The University of Pittsburgh Law Review*, Pittsburgh, Pennsylvania, Summer, 1986. https://litigation-essentials.lexisnexis.com/webcd/app?action=DocumentDisplay&crawlid=1&srctype=smi&srcid=3B15&doctype=cite&docid=47+U.+Pitt.+L.+Rev.+1037&key=06ab427bb1746d681c01861a5adac1cc. (accessed April 15, 2015.)

Dixon, Emily. "Police Arrest Julian Assange at Ecuadorian Embassy in London." *CNN*. April 11, 2019. https://www.cnn.com/2019/04/11/uk/julian-assange-arrested-gbr-intl/index.html. (accessed January 21, 2020).

Dybka, Patrycja."Mysql: Why "my"?" vertabelo.com. August 14, 2014. https://www.vertabelo.com/blog/notes-from-the-lab/mysql-history. (accessed January 30, 2020).

Earls, Alan R. Digital Equipment Corporation (Images of America): Mount Pleasant, South Carolina: Arcadia Publishing, 2004.

EFF. "A History of Protecting Freedom Where Law and Technology Collide," https://www.eff.org/about/history. (accessed April 20, 2015.)

Eldritch Press. "Eldritch Press." http://www.ibiblio.org/eldritch/. (accessed April 10, 2019.)

Electronic Frontier Foundation. "Felten, et al., v. RIAA, et al." https://www.eff.org/cases/felten-et-al-v-riaa-et-al. (accessed May 8, 2019.)

Electronic Frontier Foundation. “Us v.Elcomsoft Sklyarov.” https://www.eff.org/cases/us-v-elcomsoft-sklyarov. (accessed May 1, 2019.)
Ensmenger, Nathan. *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise.* History of Computing. Cambridge, Massachusetts: MIT Press, 2010.
Espinoza, Christopher. *Apple II Reference Manual*. Cupertino, California: Apple Computer, 1979.
Fenton, Andrew. “New Wave of Darknet Crypto Icos Led by Russian ‘Silk Road’.” Micky Media. December 17, 2019. https://micky.com.au/new-wave-of-darknet-crypto-icos-led-by-russian-silk-road/. (accessed January 21, 2020).
FindLaw.com. “Taxpayer Relief Act of 1997.” https://tax.findlaw.com/federal-taxes/taxpayer-relief-act-of-1997.html. (accessed April 2, 2019.)
Fink, Colin. “ReactOS 0.4.1.0 Released.” ReactOS.org. November 6, 2018. https://reactos.org/news. (accessed January 30, 2020).
Forrester, John. “Dozens of Masked Protesters Blast Scientology Church.” *The Boston Globe*. February 11, 2008. Pages? http://archive.boston.com/news/local/articles/2008/02/11/dozens_of_masked_protesters_blast_scientology_church/. (accessed January 30, 2020).
Gault, Matthew. “Farmers Are Buying 40-year-old Tractors Because They're Actually Repairable.” Vice.com. January 7, 2020. https://www.vice.com/en_us/article/bvgx9w/farmers-are-buying-40-year-old-tractors-because-theyre-actually-repairable. (accessed January 21, 2020).
Gibson, William, and Neil Gaiman. *Neuromancer*. Penguin Galaxy. New York, New York: Penguin Books, 2016.
Glass, Robert. “The Sociology of Opensource: Of Cults and Cultures.” *IEEE Software* 17 (May - June 2000).
Gleason, Jim. "Linux Users to Protest New Copyright Law in Washington, D.C.” Email to nylug-talk@nylug.org Mailing List. March 21, 2000.
Gleason, Jim. "press coverage of the DMCA protest." Email to nylug-talk@nylug.org Mailing List. May 4, 2000.
Gleason, Jim. “July 17, 2000.” New York Linux Users Group. http://www.nylug.org/7_17_00_page.html. (accessed April 30, 2019.)
GNU.org. “The Gnu Emacs FAQ: Where Does the Name “Emacs” Come From?.” https://www.gnu.org/software/emacs/manual/html_mono/efaq.html#Origin-of-the-term-Emacs. (accessed October 28, 2015.)
Graves, Ginny. “Attention Rapists: You've Met Your Match.” *Glamour*. June 6, 2013. https://www.glamour.com/story/attention-rapists-youve-met-your-match. (accessed January 30, 2020). Pages?
Greenberg, Andy. *This Machine Kills Secrets: How Wikileakers, Cypherpunks and Hacktivists Aim to Free the World's Information*. New York, New York: Dutton, 2012.
Greenwald, Glenn. “Xkeyscore: NSA Tool Collects 'nearly Everything a User Does On the Internet'.” *The Guardian*. July 31, 2013. https://www.theguardian.com/world/2013/jul/31/nsa-top-secret-program-online-data. (accessed January 21, 2020).

Grima, Joseph. "Open Source Design 01: The Architects of Information." *Domus,* June 18, 2011. https://www.domusweb.it/en/interviews/2011/06/18/open-source-design-01-the-architects-of-information.html. (accessed January 30, 2020).

Grune, Dick. "Concurrent Versions System CVS." VU University Amsterdam. https://dickgrune.com/Programs/CVS.orig/. (accessed November 7, 2018.)

Hafner, Katie. *Where Wizards Stay Up Late: The Origins Of The Internet.* New York, New York: Simon & Schuster, 1998.

Hajric, Vildana. "Bitcoin's 9,000,000% Rise This Decade Leaves the Skeptics Aghast." *Bloomberg*. December 31, 2019. https://www.bloomberg.com/news/articles/2019-12-31/bitcoin-s-9-000-000-rise-this-decade-leaves-the-skeptics-aghast. (accessed January 21, 2020).

Hardmeier, Sandi. "The History of Internet Explorer." Microsoft.com. August 25, 2005. http://www.microsoft.com/windows/ie/community/columns/historyofie.mspx.

Harvard Berkman Center Openlaw. "Cert Granted: The Supreme Court will hear Eldred v.Ashcroft in its fall term." https://cyber.harvard.edu/eldredvreno/legal.html. (accessed April 10, 2019.)

Harvard Berkman Center Openlaw. "Openlaw: Informal Decss History Timeline." https://cyber.harvard.edu/openlaw/DVD/research/chronology.html. (accessed April 30, 2019.)

Hensch, Mark. "Assange: WikiLeaks's Trump Info No Worse Than Him." The Hill. August 26, 2016. https://thehill.com/blogs/ballot-box/presidential-races/293453-assange-wikileaks-trump-info-no-worse-than-him.

Hunter, David. "The Roots of DOS: Tim Paterson." Paterson Technology. http://www.patersontech.com/dos/softalk.aspx. (accessed July 2, 2018.)

Hwang, Kenneth. "Blizzard Versus Bnetd: A Looming Ice Age for Free Software Development?" *Cornell Law Review* 92 (February 2013), 1044-1046.

IBM.Com. "IBM Enhances and Expands Websphere Product Line in Collaboration with Apache and Netobjects." June 22, 1998. https://www-03.ibm.com/press/us/en/pressrelease/2587.wss. (accessed January 30, 2020).

International Business Machines. "History of IBM." http://www-03.ibm.com/ibm/history/history/year_1969.html. (accessed March 14, 2015.)

International Business Machines. "IBM 701 Introduction 2." https://www.ibm.com/ibm/history/exhibits/701/701_intro2.html. (accessed January 27, 2020.)

Johns, Adrian. *Piracy: The Intellectual Property Wars from Gutenberg to Gates.* Chicago: University of Chicago Press, 2009.

Kahney, Leander. "Open Source Rich Opens Mouth." *Wired.com*. December 10, 1999. https://www.wired.com/1999/12/open-source-rich-opens-mouth/. (accessed January 30, 2020).

Kant, Immanuel. *Ethical Philosophy: The Complete Texts of Grounding for the Metaphysics of Morals, and Metaphysical Principles of Virtue.* 2nd ed. (Indianapolis: Hackett Pub. Co., 1994).

Kasson, John F. *Civilizing the Machine: Technology and Republican Values in America, 1776-1900*. New York, New York: Hill and Wang, 1999.

Kelty, Christopher M. *Two Bits: The Cultural Significance of Free Software.* Experimental Futures. Durham, North Carolina: Duke University Press, 2008.

Kramer, Dave. "A Brief History of Microsoft On the Web." Microsoft.December 24, 1999. http://www.microsoft.com/misc/features/features_flshbk.htm. (accessed March 14, 2015.)

Kurtz, Thomas. "BASIC Commands: The Original-Original Version." Dartmouth College. October 26, 2005. http://www.dartmouth.edu/basicfifty/commands.html (accessed March 22, 2018).

"Lexmark international, inc. V. Static control components, inc." Harvard Law Review 128, no. 321 (November, 2014). https://harvardlawreview.org/2014/11/lexmark-international-inc-v-static-control-components-inc/. (accessed January 30, 2020).

Laptop.org. "One Laptop Per Child." http://one.laptop.org/about/mission. (accessed May 3, 2016.)

Lee, Timothy. "How Copyright Law Threatens Your Right to Repair Your Car." Vox.com. May 28, 2015. https://www.vox.com/2015/5/26/8656281/dmca-car-repair-rights. (accessed January 21, 2020).

Lee, Yvonne. "Compuserve, MCI Mail Introduce Gateways to Internet Network." *Infoworld*, September 25, 1989.

Lerner, Josh, and Jean Tirole. "Some Simple Economics of Open Source." *The Journal of Industrial Economics*, 50, no. 2 (June 2002).

Lessig, Lawrence. *Remix: Making Art and Commerce Thrive in the Hybrid Economy*. London: Penguin Books, 2009.

Levy, Steven. *Crypto: How the Code Rebels Beat the Government, Saving Privacy in the Digital Age*. New York, New York: Viking, 2001.

Levy, Steven. *Hackers*. Sebastopol, California: O'Reilly Media, 2010.

Levy, Steven. "Busted by the Copyright Cops." *Newsweek*, August 26, 2001. https://www.newsweek.com/busted-copyright-cops-151703. (accessed January 30, 2020).

Linux Professional Institute. "About LPI Global." https://www.lpi.org/about/. (accessed March 14, 2015.)

Living Internet. "TCP/IP Internet Protocol." https://www.livinginternet.com/i/ii_tcpip.htm. (accessed October 29, 2018.)

Lorenzo, Mark Jones. *Endless Loop: The History of the Basic Programming Language (Beginner's All-Purpose Symbolic Instruction Code).* Philadelphia, Pennsylvania.: CreateSpace Independent Publishing Platform, 2017.

Malda, Rob. "A Brief History of Slashdot Part 1" Slashdot. October 3, 2007. https://news.slashdot.org/story/07/10/02/1553218/a-brief-history-of-slashdot-part-1-chips-dips. (accessed January 30, 2020).

Manne, Robert. "The Cypherpunks Revolutionary." *The Monthly*. March 2011. http://www.themonthly.com.au/issue/2011/march/1324265093/robert-manne/Cypherpunk-revolutionary. (accessed May 5, 2015.)

Marbach, William, Madlyn Resener, John Carey, Richard Sandza, Jennet Connant, and Susan Agrest. "Beware: Hackers at Play." *Newsweek*, September 5, 1983, 42-43.

Market Watch. “20 Years Already? Alan Greenspan and the ‘irrational Exuberance’ Flop.” December 6, 2016. https://www.marketwatch.com/story/20-years-already-alan-greenspan-and-the-irrational-exuberance-flop-2016-12-05. (accessed January 30, 2020).

Markoff, John. *What the Dormouse Said: How the Sixties Counterculture Shaped the Personal Computer Industry*. New York, New York: Penguin Books, 2005.

Marsan, Carolyn. “Where Is Robert Morris Now?” *NetworkWorld*, October 30, 2008. https://www.networkworld.com/article/2268914/where-is-robert-morris-now-.html. (accessed January 30, 2020).

Martin, David. “iPhone 3G S Jailbreaking Prohibited by Apple Terms of Service,” C|Net, June 8, 2009, http://www.cnet.com/news/iphone-3g-s-jailbreaking-prohibited-by-apple-terms-of-service/. (accessed April 20, 2015.)

McKenzie, Alexander. “INWG and the Conception of the Internet: An Eyewitness Account.” *IEEE Annals of the History of Computing*, January-March 2011.

McMillan, Robert. “The World's First Computer Password? It Was Useless Too.” *Wired Magazine*. January 27, 2012. https://www.wired.com/2012/01/computer-password/. (accessed January 30, 2020).

Menn, Joseph. *Cult of the Dead Cow: How the Original Hacking Supergroup Might Just Save the World*. New York, New York: Public Affairs, 2019.

Merriam-Webster. “Schema.” https://www.merriam-webster.com/dictionary/schema. (accessed December 3, 2019.)

Microsoft Introduces Internet Explorer. Redmond, Washington: Microsoft, 1994.

Microsoft.com. “Open Source, Open Cloud, and Azure Solutions from Microsoft Openness.” April 4, 2016. https://www.microsoft.com/en-us/openness/default.aspx#home. (accessed May 3, 2016.)

Miller, Alaska. “The Valley Goes Nuts for Cute Fuzzy Things.” Business Insider. November 12, 2009. https://www.businessinsider.com/open-source-mascots-2009-11.

Montague, Bruce. “Comparing the BSD and GPL licenses.” *Technology Innovation Management Review*. October, 2007. http://timreview.ca/article/67 (accessed November 5, 2017).

Moore, Jack.” Sputnik News under FBI Investigation as Russian Propaganda Arm.“ *Newsweek.* September 11, 2017. https://www.newsweek.com/russian-network-sputnik-under-fbi-investigation-possible-kremlin-propaganda-662844. (accessed January 21, 2020).

Morris, Robert, and Dennis Richie.” Password Security: A Case History.” *Communications of the ACM*, November, 1979.

Mosaic Communications Changes Name to “Netscape Communications Corporation.” Mountain View: Netscape Corporation, 1994.

Mosaic Communications Offers New Network Navigator Free On the Internet. Mountain View: Netscape Corporation, 1994.

Moss, Caroline. “From the 'Apple Lisa' to the U2 Ipod: Apple Products That Totally Flopped.” *Business Insider.* November 5, 2013. http://www.businessinsider.com/10-old-apple-products-that-totally-failed-2013-11#the-apple-iii-1980-1981-1. (accessed January 30, 2020).

Mowery, David C., and Nathan Rosenberg. *Paths of Innovation: Technological Change in 20th-Century America.* Cambridge, UK: Cambridge University Press, 1998. Accessed February 12, 2015.

Mulligan, Deirdre, and Aaron Perzanowski. "The Magnificence of the Disaster: Reconstructing the Sony Bmg Rootkit Incident." *Berkeley Technology Law Journal* 22, no. 3 (Summer 2007)**,**1998-1200.

Multicians. "Multics History." August 29, 2017. http://multicians.org/history.html (accessed March 20, 2018).

Murphy, Charles. "The White House Since Sputnik." *Fortune*, January 1958,98-101,228-232,

National Center for Supercomputing Applications." NCSA Mosaic." Accessed October 30, 2018. http://www.ncsa.illinois.edu/enabling/mosaic.

Nelson, Rebecca. "Cassandra Fairbanks loved Bernie Sanders. Now She's a Donald Trump Superfan." *Cosmopolitan*. May 18, 2017. https://www.cosmopolitan.com/politics/a9653830/cassandra-fairbanks-donald-trump-deplorable/. (accessed January 21, 2020).

Netcraft. "April 2014 Web Server Survey." April 2, 2014. http://news.netcraft.com/archives/2014/04/02/april-2014-web-server-survey.html. (accessed March 14, 2015.)

Netscape Announces Plans to Make Next-Generation Communicator Source Code Available Free On the Net. Mountain View: Netscape Corporation, 1998. http://netscape.com/newsref/pr/newsrelease558.html. (accessed January 30, 2020).

Norberg, Arthur L. *Computers and Commerce: A Study of Technology and Management at Eckert-Mauchly Computer Company, Engineering Research Associates, and Remington Rand, 1946-1957*. Cambridge, Massachusetts.: MIT Press, 2005.

Oakes, Chris. "The Non-Refundable Refund." *Wired*, January 22, 1999. https://www.wired.com/1999/01/the-non-refundable-refund/. (accessed January 30, 2020).

Ockman, Sam. "Svlug Rally." Penguin Computing. http://web.archive.org/web/19990224201149/http:/www.penguincomputing.com/svlug-rally.html. (accessed December 19, 2018.)

Pagliey, Jose. "FBI Shuts Down Online Drug Market Silk Road." CNN. October 2, 2013. https://money.cnn.com/2013/10/02/technology/silk-road-shut-down/index.html. (accessed January 21, 2020).

Pavlicek, Russell C. *Embracing Insanity: Open Source Software Development.* Indianapolis, Indiana: SAMS, 2000.

Petersen, Julie. *Fiber Optics Illustrated Dictionary, Advanced and Emerging Communications Technologies Series* (Boca Raton, Florida: CRC Press, 2003).

Php.net. "History of Php." https://www.php.net/manual/en/history.php.php. (accessed April 2, 2019.)

Pittman, Tom. "Itty Bitty Computers and TinyBasic." Ittybittycomputers.com. Last modified July 10, 2017. http://www.ittybittycomputers.com/IttyBitty/TinyBasic/ (accessed April 12, 2018).

Planes, Alex. “The IPO That Inflated the Dot-Com Bubble.” *Motley Fool.* August 9, 2013. https://www.fool.com/investing/general/2013/08/09/the-ipo-that-inflated-the-dot-com-bubble.aspx. (accessed January 30, 2020).

Proffitt, Brian. “Google Waves Goodbye to Mysql in Favor of Mariadb.” ReadWrite.com. September 14, 2013. https://readwrite.com/2013/09/14/Google-waves-goodbye-to-mysql-in-favor-of-mariadb/#awesm=~omBpxaeC3cblHt. (accessed January 30, 2020).

Progress Report On the Moon Race. *Fortune,* March 1964,142-145.

Pursell, Carroll W. *The Machine in America: A Social History of Technology.* 2nd ed. Baltimore, Maryland: Johns Hopkins University Press, 2007.

Python.org. “History and License.” Last modified April 2, 2019. https://docs.python.org/3/license.html. (accessed January 30, 2020).

ReactOS.org. “Reactos History.” Last modified June 4, 2012. http://www.reactos.org/en/about_history.html. (accessed January 30, 2020).

Red Hat. “Red Hat Training and Certification through the Years.” 2014. http://www.redhat.com/promo/cert15years/history/?intcmp=70160000000ciHuAAI. (Accessed March 14, 2015.)

Results for: OPENSOURCE.ORG.” https://whois.icann.org/en/lookup?name=www.opensource.org. (accessed December 17, 2018.)

Revolution OS, directed by J.T.S. Moore. Wonderview Productions, 2002. DVD. Wonderview Productions, 2012.

Rheingold, Howard. *The Virtual Community: Homesteading On the Electronic Frontier*. Rev. ed. Cambridge, Massachusetts: MIT Press, 2000.

Rice, Xan. “The Looting of Kenya.” *The Guardian*, August 31, 2007. https://www.theguardian.com/world/2007/aug/31/kenya.topstories3. (accessed January 30, 2020).

Rosen, Lawrence E. *Open Source Licensing: Software Freedom and Intellectual Property Law*. Upper Saddle River, New Jersey: Prentice Hall, 2005.

Rosenbaum, Ron. “Secrets of the Little Blue Box.” *Esquire*, October, 1971, 117-125,222-227.

Rosenzweig, Roy. *Clio Wired: The Future of the Past in the Digital Age*. New York, New York: Columbia University Press, 2011.

Salomon, David. *Assemblers and Loaders.* Ellis Horwood Series in Computers and Their Applications. New York, New York: Ellis Horwood, 1992.

Salus, Peter H. *A Quarter Century of UNIX.* Reading, Massachusetts: Addison-Wesley Pub. Co., 1994.

Salus, Peter H. *The Daemon, the Gnu, and the Penguin: How Free and Open Source Software is Changing the World.* Marysville, Washington.: Reed Media Services, 2004.

Sanger, Larry, “The Early History of Nupedia and Wikipedia: A Memoir,” in *Open Sources 2.0: The Continuing Evolution.* Beijing: O'Reilly, 2006.

Sangster, Andrew. “Organizations Using Python.” Python.org. July 2, 2013. https://wiki.python.org/moin/OrganizationsUsingPython. (accessed January 30, 2020).

Schein, Edgar H. and DeLisi, Peter S. *DEC Is Dead, Long Live DEC: The Lasting Legacy of Digital Equipment Corporation.* San Francisco, California: Berrett-Koehler, 2003.

Seltzer, Wendy. "Openlaw." Harvard Berkman Center. https://cyber.harvard.edu/openlaw/. (accessed April 10, 2019.)

SETI@home. "About Seti@home." https://setiathome.berkeley.edu/sah_about.php. (accessed April 13, 2019.)

SETI@home. "Porting SETI@home." https://setiathome.berkeley.edu/sah_porting.php. (accessed April 13, 2019.)

Sewell, William Hamilton. *Logics of History: Social Theory and Social Transformation*. Chicago Studies in Practices of Meaning. Chicago, Illinois: University of Chicago Press, 2005.

Share.org. "A Brief History of SHARE: It's not an acronym, it's what we do." https://www.share.org/d/do/11532. (accessed November 17, 2017.)

Shugerman, Emily. "Two Members of Alt-Right Accused of Making White Supremacist Hand Signs in White House After Receiving Press Passes." April 29, 2017. https://www.independent.co.uk/news/world/americas/us-politics/white-power-hand-symbol-cassandra-fairbanks-mike-cernovich-alt-right-white-house-a7709446.html. (accessed January 21, 2020).

Silicon Valley Linux Users Group. "Windows Refund Day Monday, February 15, 1999." http://www.svlug.org/events/refund/. (accessed December 19, 2018.)

Singh, Simon. *The Code Book*. New York, New York: Anchor Books, 1999.

Smith, Tim, and François Flückiger. "Licensing the Web." CERN. http://home.web.cern.ch/topics/birth-web/licensing-web. (accessed August 15, 2015.)

Smith, Tim, and François Flückiger. "Public Releases of Web Software." CERN. http://home.web.cern.ch/topics/birth-web/licensing-web. (Accessed November 3, 2017.)

Solution for Slashdot Effect? *Wired.* October 1, 2004. https://www.wired.com/2004/10/solution-for-slashdot-effect/. (accessed January 30, 2020).

Spence, Thomas. "The Hacker's Voice...." *Whole Earth Software Catalog*, 1984.

Spyglass Signs Agreement with NCSA to Enhance and Broadly Relicense Mosaic Graphical Browser for the Internet. Champaign, Illinois: Spyglass, Inc., 1994.

Stoll, Clifford. *The Cuckoo's Egg: Tracking a Spy through the Maze of Computer Espionage.* New York, New York: Doubleday, 1989.

Suárez-Potts, Louis. "Interview: Frank Hecker," Open Office, May 1, 2001, http://www.openoffice.org/editorial/ec1May.html. (accessed April 20, 2015.)

The Computer History Museum. "PDP-1: Spacewar!." http://www.computerhistory.org/pdp-1/spacewar!/(accessed January 30, 2020).

The Computer History Museum. "PDP-1: Time Sharing." http://www.computerhistory.org/pdp-1/timesharing/. (accessed November 17, 2017.)

The Computer History Museum. "Snowflake." http://www.computerhistory.org/pdp-1/0b08375e1559f5d4ebc67e29fa359da1/ (accessed November 17, 2017.)

The Digital Millennium Copyright Act of 1998 U.S. Copyright Office Summary. Washington, D.C.: U.S. Copyright Office, December 1998.

The Internet: A Historical Encyclopedia. 3 vols. Santa Barbara, California: ABC-CLIO, 2005.

The Open Source Initiative. "History of the OSI." September 2012. http://opensource.org/history. (accessed March 14, 2015.)

Thomas, C. William. "The Rise and Fall of Enron." *Journal of Accountancy*. April 1, 2002, 41-45, 47-48.https://www.journalofaccountancy.com/issues/2002/apr/theriseandfallofenron.html. (accessed January 30, 2020).

Thurrot, Paul. "Windows Refund Day Turns Into Sad Joke." *IT Pro Today*. February 15, 1999. https://www.itprotoday.com/windows-8/windows-refund-day-turns-sad-joke. (accessed January 30, 2020).

Timeline: The U.S. Government and Cybersecurity. *Washington Post*. May 16, 2003. http://www.washingtonpost.com/wp-dyn/articles/A50606-2002Jun26.html. (accessed January 30, 2020).

Torvalds, Linus. *Just for Fun: The Story of an Accidental Revolutionary*. New York, New York: Harper Business, 2001.

Torvalds, Linus. "Linux's History." Carnegie Mellon University. July 31, 1992. Accessed March 14, 2015. https://www.cs.cmu.edu/~awb/linux.history.html.

Touretzky, David. "Gallery of CSS Descramblers," last modified February 13, 2008, http://www.cs.cmu.edu/~dst/DeCSS/Gallery/. (accessed April 20, 2015.)

Turner, Fred. *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism*. Chicago, Illinois: University of Chicago Press, 2006.

Ulick, Jake. "Not Your Father's Nasdaq." CNN. March 19, 2002. https://money.cnn.com/2002/03/19/markets/nasdaq/index.htm. (accessed January 30, 2020).

Valloppillil, Vinod. Open Source Software: A (New?) Development Methodology. Redmond, Washington: Microsoft, 1998. https://www.gnu.org/software/fsfe/projects/ms-vs-eu/halloween1.html#_Toc427495764. (accessed January 30, 2020).

Vaughan-Nichols, Steven. "The Truth About Goobuntu: Google's In-House Desktop Ubuntu Linux." Ziff Davis. August 29, 2012. https://www.zdnet.com/article/the-truth-about-goobuntu-Googles-in-house-desktop-ubuntu-linux/. (accessed January 30, 2020).

Warren, Tom. "Microsoft Will Ship a Full Linux Kernel in Windows 10." *The Verge*. May 6, 2019. https://www.theverge.com/2019/5/6/18534687/microsoft-windows-10-linux-kernel-feature. (accessed January 21, 2020).

Watson, Thomas J. *Father, Son and Co.: My Life at IBM and Beyond.* 1990. Bantam Trade ed. New York, New York: Bantam Books, 2000

Watzinger, Martin, Thomas Fackler, Markus Nagler, and Monika Schnitzer. "How Antitrust Enforcement Can Spur Innovation: Bell Labs and the 1956 Consent Decree." Paper presented at the Center for Economic Studies (CES), the IFO

Institute Ludwig Maximilian University of Munich, Munich, Germany, February 27, 2017. http://www.cesifo-group.de/ifoHome.html (accessed March 20, 2018).

Weber, Steve. *The Success of Open Source.* Cambridge, Massachusetts: Harvard University Press, 2004.

Weins, Kyle, and Elizabeth Chamberlain. "John Deere Just Swindled Farmers Out of Their Right to Repair." *Wired.com.* September 19, 2018. https://www.wired.com/story/john-deere-farmers-right-to-repair/. (accessed January 21, 2020).

Weizenbaum, Joseph. "Computer Power and Human Reason: Science and the Compulsive Programmer." Santa Anna College. https://www.sac.edu/AcademicProgs/Business/ComputerScience/Pages/Hester_James/HACKER.htm. (accessed July 17, 2018.)

Whitehurst, Jim. "IBM Closes Landmark Acquisition of Red Hat for $34 Billion; Defines Open, Hybrid Cloud Future." Redhat.com. July 9, 2019. https://www.redhat.com/en/about/press-releases/ibm-closes-landmark-acquisition-red-hat-34-billion-defines-open-hybrid-cloud-future. (accessed January 21, 2020).

WikiLeaks. "WikiLeaks: Tor." https://www.wikileaks.org/wiki/WikiLeaks:Tor. (accessed March 14, 2015.)

Wikimedia. "Reusing Content." Last modified March 14, 2015. http://commons.wikimedia.org/wiki/Commons:Reusing_content_outside_Wikimedia. (accessed March 14, 2015.)

Wikipedia. " March 14, 2015. http://en.wikipedia.org/wiki/Wikipedia. (accessed March 14, 2015.)

Wilbur, Marcia. *A Decade of the DMCA*. Middletown, Delaware: lulu.com, 2009.

Williams, Sam. *Free as in Freedom: Richard Stallman's Crusade for Free Software.* Sebastopol, California: O'Reilly, 2002.

Wine.org. "Wine Application Database." https://appdb.winehq.org/. (accessed December 19, 2018.)

Wine.org. "Wine History." https://wiki.winehq.org/Wine_History. (accessed December 19, 2018.)

Woolf, Nicky. "An Anti-Trump Campaign Sparks a Civil War Among Anonymous Hackers." *Business Insider*. March 24, 2016. https://www.businessinsider.com/anonymous-hackers-donald-trump-2016-3. (accessed January 21, 2020).

Zawinski, Jamie. "Emacs Timeline." JWZ.org. March 8, 1999. https://www.jwz.org/doc/emacs-timeline.html. (accessed January 30, 2020).

Zetter, Kim. "Hacker's Arrest Prompts Protest Against Adobe." *PC World*. July 23, 2001. https://www.pcworld.idg.com.au/article/43670/hacker_arrest_prompts_protest_against_adobe/. (accessed January 30, 2020).

VITA

Christopher Campbell is from Delaware County, Pennsylvania. He received his Bachelor of Arts degree (2012) from West Chester University, and his Master of Arts degree (2015) from Lehigh University. As an undergraduate, he studied Business and History. His Master's degree is in History with a concentration in Technology. Christopher enjoys researching environmental, technology, and public history. He has also worked in Information Technologies for the past several decades, most recently as the Chief Information Officer of a midsized private healthcare firm in Philadelphia. He has been using UNIX since the 1980s and holds technical certifications on multiple platforms, including Linux, where he is a Red Hat Certified Engineer.