Free as in Freedom (2.0) - Richard Stallman and the Free Software Revolution

Sam Williams, Richard M. Stallman

Copyright © Sam Williams 2002; Copyright 2010 Richard M. Stallman
Published under the GNU Free Documentation License. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License."

Contents

Contents

This is *Free as in Freedom 2.0: Richard Stallman and the Free Soft-ware Revolution* , a revision of *Free as in Freedom: Richard Stallman's Crusade for Free Software*.

The cover photograph of Richard Stallman is by Peter Hinely. The PDP-10 photograph in Chapter 7 is by Rodney Brooks. The photo-graph of St. IGNUcius in Chapter 8 is by Stian Eikeland

Foreword by Richard M. Stallman

I have aimed to make this edition combine the advantages of my knowledge and Williams' interviews and outside viewpoint. The reader can judge to what extent I have achieved this.

I read the published text of the English edition for the first time in 2009 when I was asked to assist in making a French translation of *Free as in Freedom*. It called for more than small changes.

Many facts needed correction, but deeper changes were also needed. Williams, a non-programmer, blurred fundamental technical and legal distinctions, such as that between modifying an existing program's code, on the one hand, and implementing some of its ideas in a new program, on the other. Thus, the first edition said that both Gosmacs and GNU Emacs were developed by modifying the original PDP-10 Emacs, which in fact neither one was. Likewise, it mistakenly described Linux as a "version of Minix." SCO later made the same false claim in its infamous lawsuit against IBM, and both Torvalds and Tanenbaum rebutted it.

The first edition over dramatized many events by projecting spurious emotions into them. For instance, it said that I "all but shunned" Linux in 1992, and then made a "a dramatic about-face" by deciding in 1993 to sponsor Debian GNU/Linux. Both my interest in 1993 and my lack of interest in 1992 were pragmatic means to pursue the same end: to complete the GNU system. The launch of the GNU Hurd kernel in 1990 was also a pragmatic move directed at that same end.

For all these reasons, many statements in the original edition were mistaken or inco-herent. It was necessary to correct them, but not straightforward to do so with integrity short of a total rewrite, which was undesirable for other reasons. Using explicit notes for the corrections was suggested, but in most chapters the amount of change made explicit notes prohibitive. Some errors were too pervasive or too in-grained to be cor-rected by notes. Inline or footnotes for the rest would have overwhelmed the text in some places and made the text hard to read; footnotes would have been skipped by readers tired of looking down for them. I have therefore made corrections directly in the text.

However, I have not tried to check all the facts and quotations that are outside my knowledge; most of those I have simply carried forward on Williams' authority.

Williams' version contained many quotations that are critical of me. I have preserved all these, adding rebuttals when appropriate.I have not deleted any quotation, except in chapter 11where I have deleted some that were about open source and did not pertain to my life or work. Likewise I have preserved (and sometimes commented on) most of Williams' own interpretations that criticized me, when they did not represent misunder-standing of facts or technology, but I have freely corrected inaccurate assertions about my work and my thoughts and feelings. I have preserved his personal impressions when presented as such, and "I" in the text of this edition always refers to Williams except in notes labeled "RMS:".

In this edition, the complete system that combines GNU and Linux is always "GNU/Linux," 10 and "Linux" by itself always refers to Torvalds' kernel, except in quotations where the other usage is marked with "[sic]".

See ‹http://www.gnu.org/gnu/gnu-linux-faq.html› for more explanation of why it is erroneous and unfair to call the whole system "Linux."

I would like to thank John Sullivan for his many useful criticisms and suggestions. 11

Preface by Sam Williams 12

This summer marks the 10th anniversary of the email exchange that set in motion the 13
writing of *Free as in Freedom: Richard Stallman's Crusade for Free Software* and, by
extension, the work prefaced here, *Richard Stallman and the Free Software Revolution*.

Needless to say, a lot has changed over the intervening decade. 14

Originally conceived in an era of American triumphalism, the book's main storyline - 15
about one man's Jeremiah-like efforts to enlighten fellow software developers as to the
ethical, if not economic, shortsightedness of a commercial system bent on turning the
free range intellectual culture that gave birth to computer science into a rude agglom-
eration of proprietary gated communities - seems almost nostalgic, a return to the days
when the techno-capitalist system seemed to be working just fine, barring the criticism
of a few outlying skeptics.

Now that doubting the system has become almost a common virtue,it helps to look at 16
what narrative threads, if any, remained consistent over the last ten years.

While I don't follow the software industry as closely as I once did, one thing that leaps 17
out now, even more than it did then, is the ease with which ordinary consumers have
proven willing to cede vast swaths of private information and personal user liberty in
exchange for riding a top the coolest technology "platform" or the latest networking
trend.

A few years ago, I might have dubbed this the "iPod Effect," a shorthand salute to 18
Apple co-founder Steve Jobs' unrivaled success in getting both the music industry and
digital music listeners to put aside years of doubt and mutual animosity to rally around a
single, sexy device - the Apple iPod - and its restrictive licensing regime, iTunes. Were
I pitching the story to a magazine or newspaper nowadays, I'd probably have to call it
the "iPad Effect" or maybe the "Kindle Effect" both in an attempt to keep up with the
evolving brand names and to acknowledge parallel, tectonic shifts in the realm of daily
journalism and electronic book publishing.

Lest I appear to be gratuitously plugging the above-mentioned brand names, RMS sug- 19
gests that I offer equal time to a pair of websites that can spell out their many disad-
vantages, especially in the realm of software liberty. I have agreed to this suggestion
in the spirit of equal time. The web sites he recommends are DefectiveByDesign.org
and BadVista.org.

Regardless of title, the notion of corporate brand as sole guarantor of software quality 20
in a swiftly changing world remains a hard one to dislodge, even at a time when most
corporate brands are trading at or near historic lows.

Ten years ago, it wasn't hard to find yourself at a technology conference listening in 21
on a conversation (or subjected to direct tutelage) in which some old-timer, Richard
Stallman included, offered a compelling vision of an alternate possibility. It was the job
of these old-timers, I ultimately realized, to make sure we newbies in the journalism

game recognized that the tools we prided ourselves in finally knowing how to use - Microsoft Word, PowerPoint, Internet Explorer, just to name a few popular offerings from a single oft-cited vendor - were but a pale shadow of towering edifice the original architects of the personal computer set out to build.

Nowadays, it's almost as if the opposite situation is at hand. The edifice is now a sprawling ecosystem, a jungle teeming with ideas but offering only a few stable niches for sustainable growth. While one can still find plenty of hackers willing to grumble about, say, Vista's on going structural flaws, Apple's dictatorial oversight of the iPhoneApp Store or Google's shifting definition of the word "evil" - each year brings with it a fresh crop of "digital native" consumers willing to trust corporate guidance in this Hobbesian realm. Maybe that's because many of the problems that once made using your desktop computer such a teeth-grinding experience have largely been paved over with the help of free software. 22

Whatever. As consumer software reliability has improved, the race to stay one step ahead of consumer taste has put application developers in an even tighter embrace with moneyed interests. I'm not saying that the hacker ethos no longer exists or that it has even weakened in any noticeable way. I'm just saying that I doubt the programmer who generated the Facebook algorithm that rewrites the "info" pages so that each keyword points to a sponsored page, with an 80-percent semantic error rate to boot, spends much time in his new Porsche grousing about what the program really could have achieved if only the "suits" hadn't gotten in the way. 23

True, millions of people now run mostly free software on their computers with many running free software exclusively. From an ordinary consumer perspective, however, terms like "software" and "computer" have become increasingly distant. Many 2010-era cell phones could give a 2000-era laptop a run for its money in the functionality department. And yet, when it comes time to make a cell phone purchase, how many users lend any thought to the computer or software operating system making that functionality possible? The vast majority of modern phone users base their purchasing decisions almost entirely on the number of applications offered, the robustness of the network and, most important of all, the monthly service plan. Getting a consumer in this situation to view his or her software purchase through the lens of personal liberty, as opposed to personal convenience, is becoming, if not more difficult, certainly a more complex endeavor. 24

Given this form of pessimistic introduction, why should anyone want go on and read this book? 25

I can offer two major reasons. 26

The first reason is a personal one. As noted in the Epilogue of *Free as in Freedom*, Richard and I parted on less than cordial terms shortly before the publication of that book. The fault, in large part, was mine. Having worked with Richard to make sure that my biographical sketch didn't run afoul of free software principles - an effort that, I'm proud to say, made *Free as in Freedom* one of the first works to employ the GNU Free Documentation License (GFDL) as a copyright mechanism - I abruptly ended the 27

cooperative relationship when it came time to edit the work and incorporate Richard'
lengthy list of error corrections and requests for clarification.

Though able to duck behind my own principles of authorial independence and journal- 28
istic objectivity, I have since come to lament not begging the book's publisher - O'Reilly
and Associates - for additional time. Because O'Reilly had already granted my one ma-
jor stipulation - the GFDL - and had already put up with a heavy stream of last-minute
changes on my part, however, I was hesitant to push my luck.

In the years immediately following the publication of *Free as in Freedom*, I was able to 29
justify my decision by noting that the GFDL, just like the GNU General Public License
in the software realm, makes it possible for any reader to modify the book and resell it
as a competitive work. As Ernest Hemingway once put it, "the first draft of anything is
shit." If Stallman or others within the hacker community saw *Free as in Freedom* as a
first draft at best, well, at least I had spared them the time and labor of generating their
own first draft.

Now that Richard has indeed delivered what amounts to a significant rewrite, I can only 30
but remain true to my younger self and endorse the effort. Indeed, I salute it. My only
remaining hope is that, seeing as how Richard's work doesn't show any sign of slowing,
additional documentation gets added to the mix.

Before moving on to the next reason, I should note that one of the pleasant by-products 31
of this book is a re-opening of email communication channels between Richard and my-
self. The resulting communication has reacquainted me with the razor-sharp Stallman
writing style.

An illustrative and perhaps amusing anecdote for anyone out there who has wrangled 32
with Richard in text: In the course of discussing the passage in which I observe and
document the process of Richard losing his cool amid the rush hour traffic of Kihei, Maui,
a passage that served as the basis for Chapter 7 ("A Brief Journey through Hacker Hell")
in the original book, I acknowledged a common complaint among the book's reviewers
- namely, that the episode seemed out of place, a fragment of magazine-style profile
interrupting a book-length biography. I told Richard that he could discard the episode for
that reason alone but noted that my decision to include it was based on two justifications.
First, it offered a glimpse of the Stallman temper, something I'd been warned about but
had yet to experience in a first hand manner. Second, I felt the overall scene possessed
a certain metaphorical value. Hence the chapter title. Stallman, to my surprise, agreed
on both counts. His concern lay more in the two off-key words. At one point I quote
him accusing the lead driver of our two-vehicle caravan with "deliberately" leading us
down a dead-end street, an accusation that, if true, suggested a level of malice outside
the bounds of the actual situation. Without the benefit of a recorded transcript - I only
had a notebook at the time, I allowed that it was likely I'd mishandled Stallman's actual
wording and had made it more hurtful than originally intended.

On a separate issue, meanwhile, Stallman questioned his quoted use of the word "fuck- 33
ing." Again, I didn't have the moment on tape, but I wrote back that I distinctly recalled
an impressive display of profanity, a reminder of Richard's New York roots, and was

willing to stand by that memory.

An email response from Richard, received the next day, restated the critique in a way that forced me to go back and re-read the first message. As it turned out, Stallman wasn't so much objecting to the "fuck" as the "-ing" portion of the quote. 34

"Part of the reason I doubt [the words] is that they involve using fucking as an adverb," Stallman wrote. "I have never spoken that way. So I am sure the words are somewhat altered." 35

"Touch'e". 36

The second reason a person should feel compelled to read this book cycles back to the opening theme of this preface - how different a future we face in 2010 compared to the one we were still squinting our eyes to see back in 2000. I ll be honest: Like many Americans (and non-Americans), my world view was altered by the events of September 11, 2001, so much so that it wasn't much longer after the publication of *Free as in Freedom* that my attention drifted sharply away from the free software movement and Stallman's efforts to keep it on course. While I have managed to follow the broad trends and major issues, the day-to-day drama surrounding software standards, software copyrights and software patents has become something I largely skip over - the Internet news equivalent of the Water Board notes in the local daily newspaper, in other words. 37

[RMS: The September 2001 attacks, not mentioned later in the book, deserve brief comment here. Far from "changing everything," as many proclaim, the attacks have, in fact, changed very little in the U.S.: There are still scoundrels in power who hate our freedoms. The only major difference is that they can now cite "terrorists" as an excuse for laws to take them away. See the political notes on stallman.org for more about this.] 38

This is a lamentable development in large part because, ten years in, I finally see the maturing 21st century in what I believe to be a clear light. Again, if this were a pitch letter to some editor, I'd call it "The Process Century." 39

By that I mean I we stand at a rare point in history where, all cynicism aside, the power to change the world really does delegate down to the ordinary citizen's level. The catch, of course, is that the same power that belongs to you also belongs to everyone else. Wherein past eras one might have secured change simply by winning the sympathies of a few well-placed insiders, today's reformer must bring into alignment an entire vector field of competitive ideas and interests. In short, being an effective reformer nowadays requires more than just titanic stamina and a willingness to cry out in the wilderness for a decade or more, it requires knowing how to articulate durable, scalable ideas, how to beat the system at its own game. 40

On all counts, I would argue that Richard M. Stallman, while maybe not the archetype, is at the very least an ur-type of the successful reformer just described. 41

While some might lament a future in which every problem seems to take a few decades of committee meetings and sub-committee hearings just to reach the correction stage, 42

I, for one, see the alternative - a future so responsive to individual or small group action that some self-appointed actor finally decides to put that responsiveness to the test- as too chilling to contemplate.

In short, if you are the type of person who, like me, hopes to seethe 21st century follow a less bloody course than the 20th century, the Water Board - in its many frustrating guises - is where that battle is currently being fought. As hinted by the Virgil-inspired epigraph introducing the book's first chapter, I've always held out hope that this book might in some way become a sort of epic poem for the Internet Age. Built around a heroic but flawed central figure, its authorial stamp should be allowed to blur with age.

43

On that note, I would like to end this preface the same way I always end this preface - with a request for changes and contributions from any reader wishing to improve the text. Appendix B - GNU Free Documentation License offers a guide on your rights as a reader to submit changes, make corrections, or even create your own spin-off version of the book. If you prefer to simply run the changes through Richard or myself, you can find the pertinent contact information on the Free Software Foundation web site. In the meantime, good luck and enjoy the book!

44

45

Sam Williams

Staten Island,

USA

Chapter 1 - For Want of a Printer

```
I fear the Greeks. Even when they bring gifts.
                              ---Virgil
                           The Aeneid
```

The new printer was jammed, again.

Richard M. Stallman, a staff software programmer at the Massachusetts Institute of Technology's Artificial Intelligence Laboratory (AI Lab), discovered the malfunction the hard way. An hour after sending off a 50-page file to the office laser printer, Stallman, 27, broke off a productive work session to retrieve his documents. Upon arrival, he found only four pages in the printer's tray. To make matters even more frustrating, the four pages belonged to another user, meaning that Stallman's print job and the unfinished portion of somebody else's print job were still trapped somewhere within the electrical plumbing of the lab's computer network.

Waiting for machines is an occupational hazard when you're a software programmer, so Stallman took his frustration with a grain of salt. Still, the difference between waiting for a machine and waiting on a machine is a sizable one. It wasn't the first time he'd been forced to stand over the printer, watching pages print out one by one. As a person who spent the bulk of his days and nights improving the efficiency of machines and the software programs that controlled them, Stallman felt a natural urge to open up the machine, look at the guts, and seek out the root of the problem.

Unfortunately, Stallman's skills as a computer programmer did not extend to the mechanical-engineering realm. As freshly printed documents poured out of the machine, Stallman had a chance to reflect on other ways to circumvent the printing jam problem.

How long ago had it been that the staff members at the AI Lab had welcomed the new printer with open arms? Stallman wondered. The machine had been a donation from the Xerox Corporation. A cutting edge prototype, it was a modified version of a fast Xerox photocopier. Only instead of making copies, it relied on software data piped in over a computer network to turn that data into professional-looking documents. Created by engineers at the world-famous Xerox Palo Alto Research Facility, it was, quite simply, an early taste of the desktop-printing revolution that would seize the rest of the computing industry by the end of the decade.

Driven by an instinctual urge to play with the best new equipment, programmers at the AI Lab promptly integrated the new machine into the lab's sophisticated computing infrastructure. The results had been immediately pleasing. Unlike the lab's old printer, the new Xerox machine was fast. Pages came flying out at a rate of one per second, turning a 20-minute print job into a 2-minute print job. The new machine was also more precise. Circles came out looking like circles, not ovals. Straight lines came out looking like straight lines, not low-amplitude sine waves.

It was, for all intents and purposes, a gift too good to refuse.

Free Software Foundation

Once the machine was in use, its flaws began to surface. Chief among the draw- 55
backs was the machine's susceptibility to paper jams. Engineering-minded program-
mers quickly understood the reason behind the flaw. As a photocopier, the machine
generally required the direct oversight of a human operator. Figuring that these human
operators would always be on hand to fix a paper jam, if it occurred, Xerox engineers
had devoted their time and energies to eliminating other pesky problems. In engineer-
ing terms, user diligence was built into the system.

In modifying the machine for printer use, Xerox engineers had changed the user- 56
machine relationship in a subtle but profound way. Instead of making the machine
subservient to an individual human operator, they made it subservient to an entire net-
worked population of human operators. Instead of standing directly over the machine,
a human user on one end of the network sent his print command through an extended
bucket brigade of machines, expecting the desired content to arrive at the targeted des-
tination and in proper form. It wasn't until he finally went to check up on the final output
that he realized how little of it had really been printed.

Stallman was hardly the only AI Lab denizen to notice the problem, but he also thought 57
of a remedy. Years before, for the lab's previous printer, Stallman had solved a similar
problem by modifying the software program that regulated the printer, on a small PDP-
11machine, as well as the Incompatible Timesharing System that ran on the main PDP-
10 computer. Stallman couldn't eliminate paper jams, but he could insert software code
that made the PDP-11 check the printer periodically, and report jams back to the PDP-
10. Stallman also inserted code on the PDP-10 to notify every user with a waiting print
job that the printer was jammed. The notice was simple, something along the lines
of "The printer is jammed, please fix it," and because it went out to the people with
the most pressing need to fix the problem, chances were that one of them would fix it
forthwith.

As fixes go, Stallman's was oblique but elegant. It didn't fix the mechanical side of the 58
problem, but it did the next best thing by closing the information loop between user and
machine. Thanks to a few additional lines of software code, AI Lab employees could
eliminate the 10 or 15 minutes wasted each week in running back and forth to check
on the printer. In programming terms, Stallman's fix took advantage of the amplified
intelligence of the overall network.

"If you got that message, you couldn't assume somebody else would fix it," says Stall- 59
man, recalling the logic. "You had to go to the printer. A minute or two after the printer
got in trouble, the two or three people who got messages arrive to fix the machine. Of
those two or three people, one of them, at least, would usually know how to fix the
problem."

Such clever fixes were a trademark of the AI Lab and its indigenous population of 60
programmers. Indeed, the best programmers at the AI Lab disdained the term pro-
grammer, preferring the more slangy occupational title of hacker instead. The job title
covered a host of activities - everything from creative mirth making to the improvement
of existing software and computer systems. Implicit within the title, however, was the
old-fashioned notion of Yankee ingenuity. For a hacker, writing a software program

that worked was only the beginning. A hacker would try to display his cleverness (and impress other hackers) by tackling an additional challenge: to make the program particularly fast, small, powerful, elegant, or somehow impressive in a clever way.[1]

Companies like Xerox made it a policy to donate their products(and software) to places where hackers typically congregated. If hackers used these products, they might go to work for the company later on. In the 60s and early 70s, they also sometimes developed programs that were useful for the manufacturer to distribute to other customers.

When Stallman noticed the jamming tendency in the Xerox laser printer, he thought of applying the old fix or "hack" to this printer. In the course of looking up the Xerox laser-printer software, however, Stallman made a troubling discovery. The printer didn't have any software, at least nothing Stallman or a fellow programmer could read. Until then, most companies had made it a form of courtesy to publish source-code files-readable text files that documented the individual software commands that told a machine what to do. Xerox, in this instance, had provided software files only in compiled, or binary, form. If programmers looked at the files, all they would see was an endless stream of ones and zeroes - gibberish.

There are programs, called "disassemblers," to convert the ones and zeroes into low-level machine instructions, but figuring out what those instructions actually "do" is a long and hard task, known as "reverse engineering." To reverse engineer this program could have taken more time than five years' worth of jammed printouts. Stallman wasn't desperate enough for that, so he put the problem aside.

Xerox's unfriendly policy contrasted blatantly with the usual practices of the hacker community. For instance, to develop the program for the PDP-11 that ran the old printer, and the program for another PDP-11 that handled display terminals, the AI Lab needed a cross-assembler program to build PDP-11 programs on the PDP-10 main computer. The lab's hackers could have written one, but Stallman, a Harvard student, found such a program at Harvard's computer lab. That program was written to run on the same kind of computer, the PDP-10, albeit with a different operating system. Stallman never knew who had written the program, since the source code did not say. But he brought a copy back to the AI Lab. He then altered the source code to make it run on the AI Lab's Incompatible Timesharing System (ITS). With no muss and little fuss, the AI Lab got the program it needed for its software infrastructure. Stallman even added a few features not found in the original version, making the program more powerful. "We wound up using it for several years," Stallman says.

From the perspective of a 1970s-era programmer, the transaction was the software equivalent of a neighbor stopping by to borrow a power tool or a cup of sugar from a neighbor. The only difference was that in borrowing a copy of the software for the AI Lab, Stallman had done nothing to deprive anyone else of the use of the program. If anything, other hackers gained in the process, because Stallman had introduced additional features that other hackers were welcome to borrow back. For instance, Stallman

---

[1]For more on the term "hacker," see Appendix A - Hack, Hackers, and Hacking.

recalls a programmer at the private engineering firm, Bolt, Beranek & Newman, borrowing the program. He made it run on Twenex and added a few additional features, which Stallman eventually reintegrated into the AI Lab's own source-code archive. The two programmers decided to maintain a common version together, which had the code to run either on ITSor on Twenex at the user's choice.

"A program would develop the way a city develops," says Stallman, recalling the software infrastructure of the AI Lab. "Parts would get replaced and rebuilt. New things would get added on. But you could always look at a certain part and say, 'Hmm, by the style, I see this part was written back in the early 60s and this part was written in themid-1970s.'" 66

Through this simple system of intellectual accretion, hackers at the AI Lab and other places built up robust creations. Not every programmer participating in this culture described himself as a hacker, but most shared the sentiments of Richard M. Stallman. If a program or software fix was good enough to solve your problems, it was good enough to solve somebody else's problems. Why not share it out of a simple desire for good karma? 67

This system of cooperation was being undermined by commercial secrecy and greed, leading to peculiar combinations of secrecy and co-operation. For instance, computer scientists at UC Berkeley had built up a powerful operating system called BSD, based on the Unix system they had obtained from AT&T. Berkeley made BSD available for the cost of copying a tape, but would only give these tapes to schools that could present a $50,000 source license obtained from AT&T. The Berkeley hackers continued to share as much as AT&T let them, but they had not perceived a conflict between the two practices. 68

Likewise, Stallman was annoyed that Xerox had not provided the source-code files, but not yet angry. He never thought of asking Xerox for a copy. "They had already given us the laser printer," Stallman says. "I could not say they owed us something more. Besides, I took for granted that the absence of source code reflected an intentional decision, and that asking them to change it would be futile." 69

Good news eventually arrived: word had it that a scientist at the computer-science department at Carnegie Mellon University had a copy of the laser printer source code. 70

The association with Carnegie Mellon did not augur well. In 1979, Brian Reid, a doctoral student there, had shocked the community by refusing to share his text-formatting program, dubbed Scribe. This text formatter was the first to have mark-up commands oriented to-wards the desired semantics (such as "emphasize this word" or "this paragraph is a quotation") rather than low-level formatting details("put this word in italics" or "narrow the margins for this paragraph"). Instead Reid sold Scribe to a Pittsburgh-area software company called Unilogic. His graduate-student career ending, Reid says he simply was looking for a way to unload the program on a set of developers that would take pains to keep it from slipping into the public domain.(Why one would consider such an outcome particularly undesirable is not clear.) To sweeten the deal, Reid also agreed 71

to insert a set of time-dependent functions - "time bombs" in software-programmer parlance - that deactivated freely copied versions of the program after a 90-day expiration date. To avoid deactivation, users paid the software company, which then issued a code that defused the internal time-bomb anti-feature.

For Stallman, this was a betrayal of the programmer ethos, pure and simple. Instead of honoring the notion of share-and-share alike, Reid had inserted a way for companies to compel programmers to pay for information access. But he didn't think deeply about the question, since he didn't use Scribe much. 72

Unilogic gave the AI Lab a gratis copy to use, but did not remove or mention the time bomb. It worked, for a while; then one day a user reported that Scribe had stopped working. System hacker Howard Cannon spent hours debugging the binary until he found the time-bomb and patched it out. Cannon was incensed, and wasn't shy about telling the other hackers how mad he was that Unilogic had wasted his time with an intentional bug. 73

Stallman had a Lab-related reason, a few months later, to visit the Carnegie Mellon campus. During that visit, he made a point of looking for the person reported to have the printer software source code. By good fortune, the man was in his office. 74

In true engineer-to-engineer fashion, the conversation was cordial but blunt. After briefly introducing himself as a visitor from MIT, Stallman requested a copy of the laser-printer source code that he wanted to modify. To his chagrin, the researcher refused. 75

"He told me that he had promised not to give me a copy," Stallman says. 76

Memory is a funny thing. Twenty years after the fact, Stallman's mental history tape is blank in places. Not only does he not remember the motivating reason for the trip or even the time of year during which he took it, he also has no recollection of who was on the other end of the conversation. According to Reid, the person most likely to have fielded Stallman's request is Robert Sproull, a former Xerox PARC researcher and current director of Sun Laboratories, a research division of the computer-technology conglomerate Sun Microsystems. During the 1970s, Sproull had been the primary developer of the laser-printer software in question while at Xerox PARC. Around 1980, Sproull took a faculty research position at Carnegie Mellon where he continued his laser-printer work amid other projects. 77

When asked directly about the request, however, Sproull draws a blank. "I can't make a factual comment," writes Sproull via email. "I have absolutely no recollection of the incident." 78

"The code that Stallman was asking for was leading-edge, state-of-the-art code that Sproull had written in the year or so before going to Carnegie Mellon," recalls Reid. If so, that might indicate a mis-understanding that occurred, since Stallman wanted the source for the program that MIT had used for quite some time, not some newer version. But the question of which version never arose in the brief conversation. 79

In talking to audiences, Stallman has made repeated reference to the incident, noting 80

that the man's unwillingness to hand over the source code stemmed from a nondisclosure agreement, a contractual agreement between him and the Xerox Corporation giving the signatory access to the software source code in exchange for a promise of secrecy. Now a standard item of business in the software industry, the nondisclosure agreement, or NDA, was a novel development at the time, a reflection of both the commercial value of the laser printer to Xerox and the information needed to run it. "Xerox was at the time trying to make a commercial product out of the laser printer," recalls Reid. "They would have been insane to give away the source code."

For Stallman, however, the NDA was something else entirely. It was a refusal on the part of some CMU researcher to participate in a society that, until then, had encouraged software programmers to regard programs as communal resources. Like a peasant whose centuries-old irrigation ditch had grown suddenly dry, Stallman had followed the ditch to its source only to find a brand-spanking-new hydroelectric dam bearing the Xerox logo. [81]

For Stallman, the realization that Xerox had compelled a fellow programmer to participate in this newfangled system of compelled secrecy took a while to sink in. In the first moment, he could only seethe refusal in a personal context. "I was so angry I couldn't think of a way to express it. So I just turned away and walked out without another word," Stallman recalls. "I might have slammed the door. Who knows? All I remember is wanting to get out of there. I went to his office expecting him to cooperate, so I had not thought about how I would respond if he refused. When he did, I was stunned speechless as well as disappointed and angry." [82]

Twenty years after the fact, the anger still lingers, and Stallman presents the event as one that made him confront an ethical issue, though not the only such event on his path. Within the next few months, a series of events would befall both Stallman and the AI Lab hacker community that would make 30 seconds worth of tension in a remote Carnegie Mellon office seem trivial by comparison. Nevertheless, when it comes time to sort out the events that would transform Stallman from a lone hacker, instinctively suspicious of centralized authority, to a crusading activist applying traditional notions of liberty, equality, and fraternity to the world of software development, Stallman singles out the Carnegie Mellon encounter for special attention. [83]

"It was my first encounter with a nondisclosure agreement, and it immediately taught me that nondisclosure agreements have victims," says Stallman, firmly. "In this case I was the victim. [My lab and I]were victims." [84]

Stallman later explained, "If he had refused me his cooperation for personal reasons, it would not have raised any larger issue. I might have considered him a jerk, but no more. The fact that his refusal was impersonal, that he had promised in advance to be uncooperative, not just to me but to anyone whatsoever, made this a larger issue." [85]

Although previous events had raised Stallman's ire, he says it wasn't until his Carnegie Mellon encounter that he realized the events were beginning to intrude on a culture he had long considered sacrosanct. He said, "I already had an idea that software should be shared, but I wasn't sure how to think about that. My thoughts weren't clear and [86]

organized to the point where I could express them in a concise fashion to the rest of the world. After this experience, I started to recognize what the issue was, and how big it was."

As an elite programmer at one of the world's elite institutions, Stallman had been perfectly willing to ignore the compromises and bargains of his fellow programmers just so long as they didn't interfere with his own work. Until the arrival of the Xerox laser printer, Stallman had been content to look down on the machines and programs other computer users grimly tolerated.

Now that the laser printer had insinuated itself within the AI Lab's network, however, something had changed. The machine worked fine, barring the paper jams, but the ability to modify software according to personal taste or community need had been taken away. From the viewpoint of the software industry, the printer software represented a change in business tactics. Software had become such a valuable asset that companies no longer accepted the need to publicize source code, especially when publication meant giving potential competitors a chance to duplicate something cheaply. From Stallman's viewpoint, the printer was a Trojan Horse. After a decade of failure, software that users could not change and redistribute - future hackers would use the term "proprietary" software - had gained a foothold inside the AI Lab through the sneakiest of methods. It had come disguised as a gift.

That Xerox had offered some programmers access to additional gifts in exchange for secrecy was also galling, but Stallman takes pains to note that, if presented with such a quid pro quo bargain at a younger age, he just might have taken the Xerox Corporation up on its offer. The anger of the Carnegie Mellon encounter, however, had a firming effect on Stallman's own moral lassitude. Not only did it give him the necessary anger to view such future offers with suspicion, it also forced him to turn the situation around: what if a fellow hacker dropped into Stallman's office someday and it suddenly became Stallman's job to refuse the hacker's request for source code?

"When somebody invited me to betray all my colleagues in that way, I remembered how angry I was when somebody else had done that to me and my whole lab," Stallman says. "So I said, 'Thank you very much for offering me this nice software package, but I can't accept it on the conditions that you're asking for, so I'm going to do without it.'"

It was a lesson Stallman would carry with him through the tumultuous years of the 1980s, a decade during which many of his MIT colleagues would depart the AI Lab and sign nondisclosure agreements of their own. They may have told themselves that this was a necessary evil so they could work on the best projects. For Stallman, however, the NDA called the moral legitimacy of the project into question. What good is a technically exciting project if it is meant to be withheld from the community?

As Stallman would quickly learn, refusing such offers involved more than personal sacrifice. It involved segregating himself from fellow hackers who, though sharing a similar distaste for secrecy, tended to express that distaste in a more morally flexible fashion. Refusing another's request for source code, Stallman decided, was not only a betrayal of the scientific mission that had nurtured software development since the end of World

War II, it was a violation of the Golden Rule, the baseline moral dictate to do unto others as you would have them do unto you.

Hence the importance of the laser printer and the encounter that resulted from it. Without it, Stallman says, his life might have followed a more ordinary path, one balancing the material comforts of a commercial programmer with the ultimate frustration of a life spent writing invisible software code. There would have been no sense of clarity, no urgency to address a problem others weren't addressing. Most importantly, there would have been no righteous anger, an emotion that, as we soon shall see, has propelled Stallman's career as surely as any political ideology or ethical belief. 93

"From that day forward, I decided this was something I could never participate in," says Stallman, alluding to the practice of trading personal liberty for the sake of convenience - Stallman's description of the NDA bargain - as well as the overall culture that encouraged such ethically suspect deal-making in the first place. "I decided never to make other people victims as I had been a victim." 94

Chapter 2 - 2001: A Hacker's Odyssey   95

The New York University computer-science department sits inside Warren Weaver Hall,   96
a fortress-like building located two blocks east of Washington Square Park. Industrial-
strength air-conditioning vents create a surrounding moat of hot air, discouraging loiter-
ers and solicitors alike. Visitors who breach the moat encounter another formidable bar-
rier, a security check-in counter immediately inside the building's single entryway.

Beyond the security checkpoint, the atmosphere relaxes somewhat. Still, numerous   97
signs scattered throughout the first floor preach the dangers of unsecured doors and
propped-open fire exits. Taken as a whole, the signs offer a reminder: even in the
relatively tranquil confines of pre-September 11, 2001, New York, one can never be too
careful or too suspicious.

The signs offer an interesting thematic counterpoint to the growing number of visitors   98
gathering in the hall's interior atrium. A few look like NYU students. Most look like
shaggy-haired concert-goers milling outside a music hall in anticipation of the main act.
For one brief morning, the masses have taken over Warren Weaver Hall, leaving the
nearby security attendant with nothing better to do but watch Ricki Lake on TV and
shrug her shoulders toward the nearby auditorium whenever visitors ask about "the
speech."

Once inside the auditorium, a visitor finds the person who has forced this temporary   99
shutdown of building security procedures. The person is Richard M. Stallman, founder
of the GNU Project, original president of the Free Software Foundation, winner of
the 1990 MacArthur Fellowship, winner of the Association of Computing Machinery's
Grace Murray Hopper Award (also in 1990), co-recipient of the Takeda Foundation's
2001 Takeda Award for Social/Economic Betterment, and former AI Lab hacker. As
announced over a host of hacker-related web sites, including the GNU Project's own
‹http://www.gnu.org› site, Stallman is in Manhattan, his former hometown, to deliver a much
anticipated speech in rebuttal to the Microsoft Corporation's recent campaign against
the GNU General Public License.

The subject of Stallman's speech is the history and future of the free software move-   100
ment. The location is significant. Less than a month before, Microsoft senior vice
president Craig Mundie appeared at the nearby NYU Stern School of Business, deliv-
ering a speech blasting the GNU General Public License, or GNU GPL, a legal device
originally conceived by Stallman 16 years before. Built to counteract the growing wave
of software secrecy overtaking the computer industry - a wave first noticed by Stallman
during his 1980 troubles with the Xerox laser printer - the GPL has evolved into a cen-
tral tool of the free software community. In simplest terms, the GPL establishes a form
of communal ownership - what today's legal scholars now call the "digital commons"
- through the legal weight of copyright. The GPL makes this irrevocable; once an au-
thor gives code to the community in this way, that code can't be made proprietary by
anyone else. Derivative versions must carry the same copyright license, if they use a
substantial amount of the original source code. For this reason, critics of the GPL have
taken to calling it a "viral" license, suggesting inaccurately that it spreads itself to every

software program it touches.[2]

In an information economy increasingly dependent on software and increasingly be- 101
holden to software standards, the GPL has become the proverbial "big stick." Even
companies that once derided it as "software socialism" have come around to recognize
the benefits. Linux, the kernel developed by Finnish college student Linus Torvalds in
1991, is licensed under the GPL, as are most parts of the GNU system: GNU Emacs,
the GNU Debugger, the GNU C Compiler, etc. Together, these tools form the com-
ponents of the free software GNU/Linux operating system, developed, nurtured, and
owned by the worldwide hacker community. Instead of viewing this community as a
threat, high-tech companies like IBM, Hewlett Packard, and Sun Microsystems have
come to rely upon it, selling software applications and services built to ride atop the
ever-growing free software infrastructure.[3]

They've also come to rely upon it as a strategic weapon in the hacker community's 102
perennial war against Microsoft, the Redmond, Washington-based company that has
dominated the PC-software marketplace since the late 1980s. As owner of the popular
Windows operating system, Microsoft stands to lose the most in an industry-wide shift to
the GPL license. Each program in the Windows colossus is covered by copyrights and
contracts (End User License Agreements, or EULAs) asserting the proprietary status
of the executable, as well as the underlying source code that users can't get anyway.
Incorporating code protected by the "viral" GPL into one of these programs is forbidden;
to comply with the GPL's requirements, Microsoft would be legally required to make
that whole program free software. Rival companies could then copy, modify, and sell
improved versions of it, taking away the basis of Microsoft's lock over the users.

Hence the company's growing concern over the GPL's rate of adoption. Hence the 103
recent Mundie speech blasting the GPL and the "open source" approach to software
development and sales. (Microsoft does not even acknowledge the term "free soft-
ware," preferring to use its attacks to direct attention towards the apolitical "open source"
camp described in chapter 11, and away from the free software movement.) And hence
Stallman's decision to deliver a public rebuttal to that speech on the same campus here
today.

20 years is a long time in the software industry. Consider this: in 1980, when Richard 104
Stallman was cursing the AI Lab's Xerox laser printer, Microsoft, which dominates the
worldwide software industry, was still a privately held startup. IBM, the company then
regarded as the most powerful force in the computer hardware industry, had yet to
introduce its first personal computer, thereby igniting the current low-cost PC market.

---

[2]Actually, the GPL's powers are not quite that potent: just putting your code in the same computer
with a GPL-covered program does not put your code under the GPL. "To compare something to a virus
is very harsh," says Stallman. "A spider plant is a more accurate comparison; it goes to another place if
you actively take a cutting." For more information on the GNU General Public License,
visit <http://www.gnu.org/copyleft/gpl.html>.

[3]Although these applications run on GNU/Linux, it does not follow that they are themselves free
software. On the contrary, most of them applications are proprietary software, and respect your
freedom no more than Windows does. They may contribute to the success of GNU/Linux, but they
don't contribute to the goal of freedom for which it exists.

Many of the technologies we now take for granted - the World Wide Web, satellite television, 32-bit video-game consoles - didn't even exist. The same goes for many of the companies that now fill the upper echelons of the corporate establishment, companies like AOL, Sun Microsystems, Amazon.com, Compaq, and Dell. The list goes on and on.

Among those who value progress above freedom, the fact that the high-technology marketplace has come so far in such little time is cited both for and against the GNU GPL. Some argue in favor of the GPL, pointing to the short lifespan of most computer hardware platforms. Facing the risk of buying an obsolete product, consumers tend to flock to companies with the best long-term survival. As a result, the software marketplace has become a winner-take-all arena.[4]

Such admonitions serve as the backdrop for Stallman's speech today. Less than a month after their utterance, Stallman stands with his back to one of the chalk boards at the front of the room, edgy to begin.

If the last two decades have brought dramatic changes to the software marketplace, they have brought even more dramatic changes to Stallman himself. Gone is the skinny, clean-shaven hacker who once spent his entire days communing with his beloved PDP-10. In his place stands a heavy-set middle-aged man with long hair and rabbinical beard, a man who now spends the bulk of his time writing and answering email, haranguing fellow programmers, and giving speeches like the one today. Dressed in an aqua-colored T-shirt and brown polyester pants, Stallman looks like a desert hermit who just stepped out of a Salvation Army dressing room.

The crowd is filled with visitors who share Stallman's fashion and grooming tastes. Many come bearing laptop computers and cellular modems, all the better to record and

105

106

107

108

---

[4]See Shubha Ghosh, "Revealing the Microsoft Windows Source Code," Gi-galaw.com (January, 2000),
<http://www.gigalaw.com/.
› The proprietary software environment, they say, leads to monopoly abuse and stagnation. Strong companies suck all the oxygen out of the marketplace for rival competitors and innovative startups. Others argue just the opposite. Selling software is just as risky, if not more risky, than buying software, they say. Without the legal guarantees provided by proprietary software licenses, not to mention the economic prospects of a privately owned "killer app" (i.e., a break-through technology that launches an entirely new market),[5]
The mutual success of GNU/Linux and Windows over the last 10years suggests that both sides on this question are sometimes right. However, free software activists such as Stallman think this is a side issue. The real question, they say, isn't whether free or proprietary software will succeed more, it's which one is more ethical.
Nevertheless, the battle for momentum is an important one in the software industry. Even powerful vendors such as Microsoft rely on the support of third-party software developers whose tools, programs, and computer games make an underlying software platform such as Windows more attractive to the mainstream consumer. Citing the rapid evolution of the technology marketplace over the last 20 years, not to mention his own company's impressive track record during that period, Mundie advised listeners to not get too carried away by the free software movement's recent momentum:
> Two decades of experience have shown that an economic model that protects intellectual property and a business model that recoups research and development costs can create impressive economic benefits and distribute them very broadly.[7]

transmit Stallman's words to a waiting Internet audience. The gender ratio is roughly 15 males to 1 female, and 1 of the 7 or 8 females in the room comes in bearing a stuffed penguin, the official Linux mascot, while another carries a stuffed teddy bear.

Agitated, Stallman leaves his post at the front of the room and takes a seat in a front-row chair, tapping commands into an already-opened laptop. For the next 10 minutes Stallman is oblivious to the growing number of students, professors, and fans circulating in front of him at the foot of the auditorium stage. ₁₀₉

Before the speech can begin, the baroque rituals of academic formality must be observed. Stallman's appearance merits not one but two introductions. Mike Uretsky, co-director of the Stern School's Center for Advanced Technology, provides the first. ₁₁₀

"The role of a university is to foster debate and to have interesting discussions," Uretsky says. "This particular presentation, this seminar falls right into that mold. I find the discussion of open source particularly interesting." ₁₁₁

Before Uretsky can get another sentence out, Stallman is on his feet waving him down like a stranded motorist. ₁₁₂

"I do free software," Stallman says to rising laughter. "Open source is a different movement. ₁₁₃

"The laughter gives way to applause. The room is stocked with Stallman partisans, people who know of his reputation for verbal exactitude, not to mention his much publicized 1998 falling out with the open source software proponents. Most have come to anticipate such outbursts the same way radio fans once waited for Jack Benny's trademark, "Now cut that out!" phrase during each radio program. ₁₁₄

Uretsky hastily finishes his introduction and cedes the stage to Edmond Schonberg, a professor in the NYU computer-science department. As a computer programmer and GNU Project contributor, Schonberg knows which linguistic land mines to avoid. He deftly summarizes Stallman's career from the perspective of a modern-day programmer. ₁₁₅

"Richard is the perfect example of somebody who, by acting locally, started thinking globally [about] problems concerning the un-availability of source code," says Schonberg. "He has developed a coherent philosophy that has forced all of us to reexamine our ideas of how software is produced, of what intellectual property means, and of what the software community actually represents."[8] ₁₁₆

Schonberg welcomes Stallman to more applause. Stallman takes a moment to shut off his laptop, rises out of his chair, and takes the stage. ₁₁₇

At first, Stallman's address seems more Catskills comedy routine than political speech. "I'd like to thank Microsoft for providing me the opportunity to be on this platform," Stallman wisecracks. "For the past few weeks, I have felt like an author whose book was fortuitously banned somewhere." ₁₁₈

---

[8]If this were to be said today, Stallman would object to the term "intellectual property" as carrying bias and confusion.
See ‹http://www.gnu.org/philosophy/not-ipr.html›.

Free Software Foundation

For the uninitiated, Stallman dives into a quick free software warm-up analogy. He 119 likens a software program to a cooking recipe. Both provide useful step-by-step instructions on how to complete a desired task and can be easily modified if a user has special desires or circumstances. "You don't have to follow a recipe exactly," Stallman notes. "You can leave out some ingredients. Add some mushrooms, 'cause you like mushrooms. Put in less salt because your doctor said you should cut down on salt - whatever."

Most importantly, Stallman says, software programs and recipes are both easy to share. 120 In giving a recipe to a dinner guest, a cook loses little more than time and the cost of the paper the recipe was written on. Software programs require even less, usually a few mouse-clicks and a modicum of electricity. In both instances, however, the person giving the information gains two things: increased friendship and the ability to borrow interesting recipes in return.

"Imagine what it would be like if recipes were packaged inside black boxes," Stallman 121 says, shifting gears. "You couldn't see what ingredients they're using, let alone change them, and imagine if you made a copy for a friend. They would call you a pirate and try to put you in prison for years. That world would create tremendous outrage from all the people who are used to sharing recipes. But that is exactly what the world of proprietary software is like. A world in which common decency towards other people is prohibited or prevented."

With this introductory analogy out of the way, Stallman launches into a retelling of the 122 Xerox laser-printer episode. Like the recipe analogy, the laser-printer story is a useful rhetorical device. With its parable-like structure, it dramatizes just how quickly things can change in the software world. Drawing listeners back to an era before Amazon.com one-click shopping, Microsoft Windows, and Oracle databases, it asks the listener to examine the notion of software ownership free of its current corporate logos.

Stallman delivers the story with all the polish and practice of a local district attorney 123 conducting a closing argument. When he gets to the part about the Carnegie Mellon professor refusing to lend him a copy of the printer source code, Stallman pauses.

"He had betrayed us," Stallman says. "But he didn't just do it to us. Chances are he did 124 it to you."

On the word "you," Stallman points his index finger accusingly at an unsuspecting mem- 125 ber of the audience. The targeted audience member's eyebrows flinch slightly, but Stallman's own eyes have moved on. Slowly and deliberately, Stallman picks out a second listener to nervous titters from the crowd. "And I think, mostly likely, he did it to you, too," he says, pointing at an audience member three rows behind the first.

By the time Stallman has a third audience member picked out, the titters have given 126 away to general laughter. The gesture seems a bit staged, because it is. Still, when it comes time to wrap up the Xerox laser-printer story, Stallman does so with a showman's flourish. "He probably did it to most of the people here in this room - except a few, maybe, who weren't born yet in 1980," Stallman says, drawing more laughs. "[That's]

because he had promised to refuse to cooperate with just about the entire population of the planet Earth."

Stallman lets the comment sink in for a half-beat. "He had signed a nondisclosure agreement," Stallman adds. 127

Richard Matthew Stallman's rise from frustrated academic to political leader over the last 20 years speaks to many things. It speaks to Stallman's stubborn nature and prodigious will. It speaks to the clearly articulated vision and values of the free software movement Stallman helped build. It speaks to the high-quality software programs Stallman has built, programs that have cemented Stallman's reputation as a programming legend. It speaks to the growing momentum of the GPL, a legal innovation that many Stallman observers see as his most momentous accomplishment. 128

Most importantly, it speaks to the changing nature of political power in a world increasingly beholden to computer technology and the software programs that power that technology. 129

Maybe that's why, even at a time when most high-technology stars are on the wane, Stallman's star has grown. Since launching the GNU Project in 1984,[9] Stallman has been at turns ignored, satirized, vilified, and attacked-both from within and without the free software movement. Through it all, the GNU Project has managed to meet its milestones, albeit with a few notorious delays, and stay relevant in a software marketplace several orders of magnitude more complex than the one it entered 18 years ago. So too has the free software ideology, an ideology meticulously groomed by Stallman himself. 130

To understand the reasons behind this currency, it helps to examine Richard Stallman both in his own words and in the words of the people who have collaborated and battled with him along the way. The Richard Stallman character sketch is not a complicated one. If any person exemplifies the old adage "what you see is what you get," it's Stallman. 131

"I think if you want to understand Richard Stallman the human being, you really need to see all of the parts as a consistent whole," advises Eben Moglen, legal counsel to 132

---

[9]The acronym GNU stands for "GNU's not Unix." In another portion of the May 29, 2001, NYU speech, Stallman summed up the acronym's origin:

We hackers always look for a funny or naughty name for a program, because naming a program is half the fun of writing the program. We also had a tradition of recursive acronyms, to say that the program that you're writing is similar to some existing program... I looked for a recursive acronym for Something Is Not UNIX. And I tried all 26 letters and discovered that none of them was a word. I decided to make it a contraction. That way I could have a three-letter acronym, for Something's Not UNIX. And I tried letters, and I came across the word "GNU." That was it.

Although a fan of puns, Stallman recommends that software users pronounce the "g" at the beginning of the acronym (i.e., "gah-new").Not only does this avoid confusion with the word "gnu," the name of the African antelope, Connochaetes gnou, it also avoids confusion with the adjective "new." "We've been working on it for 17 years now, so it is not exactly new any more," Stallman says.

Source: author notes and online transcript of "Free Software: Freedom and Cooperation," Richard Stallman's May 29, 2001, speech at New York University,

`<http://www.gnu.org/events/rms-nyu-2001-transcript.txt>`.

the Free Software Foundation and professor of law at Columbia University Law School. "All those personal eccentricities that lots of people see as obstacles to getting to know Stallman really 'are' Stallman: Richard's strong sense of personal frustration, his enormous sense of principled ethical commitment, his inability to compromise, especially on issues he considers fundamental. These are all the very reasons Richard did what he did when he did."

Explaining how a journey that started with a laser printer would eventually lead to a sparring match with the world's richest corporation is no easy task. It requires a thoughtful examination of the forces that have made software ownership so important in today's society. It also requires a thoughtful examination of a man who, like many political leaders before him, understands the malleability of human memory. It requires an ability to interpret the myths and politically laden codewords that have built up around Stallman over time. Finally, it requires an understanding of Stallman's genius as a programmer and his failures and successes in translating that genius to other pursuits. 133

When it comes to offering his own summary of the journey, Stallman acknowledges the fusion of personality and principle observed by Moglen. "Stubbornness is my strong suit," he says. "Most people who attempt to do anything of any great difficulty eventually get discouraged and give up. I never gave up." 134

He also credits blind chance. Had it not been for that run-in over the Xerox laser printer, had it not been for the personal and political conflicts that closed out his career as an MIT employee, had it not been for a half dozen other timely factors, Stallman finds it very easy to picture his life following a different career path. That being said, Stallman gives thanks to the forces and circumstances that put him in the position to make a difference. 135

"I had just the right skills," says Stallman, summing up his decision for launching the GNU Project to the audience. "Nobody was there but me, so I felt like, 'I'm elected. I have to work on this. If not me, who?'" 136

Chapter 3 - A Portrait of the Hacker as a Young Man 141

Richard Stallman's mother, Alice Lippman, still remembers the moment she realized 142 her son had a special gift.

"I think it was when he was eight," Lippman recalls. 143

The year was 1961, and Lippman, a recently divorced single mother, was whiling away 144 a weekend afternoon within the family's tiny one-bedroom apartment on Manhattan's Upper West Side. Leafing throug ha copy of Scientific American, Lippman came upon her favorite section, the Martin Gardner-authored column titled "Mathematical Games." A substitute art teacher at the time, Lippman enjoyed Gardner's column for the brain-teasers it provided. With her son already ensconced in a book on the nearby sofa, Lippman decided to take a crack at solving the week's feature puzzle.

"I wasn't the best person when it came to solving the puzzles," she admits. "But as an 145 artist, I found they really helped me work through conceptual barriers."

Lippman says her attempt to solve the puzzle met an immediate brick wall. About to 146 throw the magazine down in disgust, Lippman was surprised by a gentle tug on her shirt sleeve.

"It was Richard," she recalls, "He wanted to know if I needed any help." 147

Looking back and forth, between the puzzle and her son, Lippman says she initially 148 regarded the offer with skepticism. "I asked Richard if he'd read the magazine," she says. "He told me that, yes, he had and what's more he'd already solved the puzzle. The next thing I know, he starts explaining to me how to solve it."

Hearing the logic of her son's approach, Lippman's skepticism quickly gave way to 149 incredulity. "I mean, I always knew he was a bright boy," she says, "but this was the first time I'd seen anything that suggested how advanced he really was."

Thirty years after the fact, Lippman punctuates the memory with a laugh. "To tell you the 150 truth, I don't think I ever figured out how to solve that puzzle," she says. "All I remember is being amazed he knew the answer."

Seated at the dining-room table of her second Manhattan apartment - the same spa- 151 cious three-bedroom complex she and her son moved to following her 1967 marriage to Maurice Lippman, now deceased - Alice Lippman exudes a Jewish mother's mixture of pride and bemusement when recalling her son's early years. The nearby dining-room credenza offers an eight-by-ten photo of Stallman glowering in full beard and doctoral robes. The image dwarfs accompanying photos of Lippman's nieces and nephews, but before a visitor can make too much of it, Lippman makes sure to balance its prominent placement with an ironic wisecrack."

Richard insisted I have it after he received his honorary doctorate at the University of 152 Glasgow," says Lippman. "He said to me, 'Guess what, mom? It's the first graduation

I ever attended.'"[10]

Needless to say, the essay caused some concern. Neither Tracy nor I had discussed   153
the software her company would use nor had we discussed the type of copyright [license] that would govern the e-book's usage. I mentioned the *Technology Review*

---

eight and didn't let up until her son's high-school graduation in 1970. "I'd call him for dinner, and he'd never hear me. I'd have to call him 9 or 10 times just to get his attention. He was totally immersed." Stallman, for his part, remembers things in a similar fashion, albeit with a political twist.

"I enjoyed reading," he says. "If I wanted to read, and my mother told me to go to the kitchen and eat or go to sleep, I wasn't going to listen. I saw no reason why I couldn't read. No reason why she should be able to tell me what to do, period. Essentially, what I had read about, ideas such as democracy and individual freedom, I applied to myself. I didn't see any reason to exclude children from these principles." The belief in individual freedom over arbitrary authority extended to school as well. Two years ahead of his classmates by age 11, Stallman endured all the usual frustrations of a gifted public-school student. It wasn't long after the puzzle incident that his mother attended the first in what would become a long string of parent-teacher conferences.

"He absolutely refused to write papers," says Lippman, recalling an early controversy. "I think the last paper he wrote before his senior year in high school was an essay on the history of the number system in the west for a fourth-grade teacher." To be required to choose a specific topic when there was nothing he actually wanted to write about was almost impossible for Stallman, and painful enough to make him go to great lengths to avoid such situations.

Gifted in anything that required analytical thinking, Stallman gravitated toward math and science at the expense of his other studies. What some teachers saw as single-mindedness, however, Lippman saw as impatience. Math and science offered simply too much opportunity to learn, especially in comparison to subjects and pursuits for which her son seemed less naturally inclined. Around age 10 or 11, when the boys in Stallman's class began playing a regular game of touch football, she remembers her son coming home in a rage. "He wanted to play so badly, but he just didn't have the coordination skills," Lippman recalls. "It made him so angry."

The anger eventually drove her son to focus on math and science all the more. Even in the realm of science, however, her son's impatience could be problematic. Poring through calculus textbooks by age seven, Stallman saw little need to dumb down his discourse for adults. Sometime, during his middle-school years, Lippman hired a student from nearby Columbia University to play big brother to her son. The student left the family's apartment after the first session and never came back. "I think what Richard was talking about went over his head," Lippman speculates.

Another favorite maternal memory dates back to the early 1960s, shortly after the puzzle incident. Around age seven, two years after the divorce and relocation from Queens, Richard took up the hobby of launching model rockets in nearby Riverside Drive Park. What started as aimless fun soon took on an earnest edge as her son began recording the data from each launch. Like the interest in mathematical games,the pursuit drew little attention until one day, just before a major NASA launch, Lippman checked in on her son to see if he wanted to watch.

"He was fuming," Lippman says. "All he could say to me was, 'But I'm not published yet.' Apparently he had something that he really wanted to show NASA." Stallman doesn't remember the incident, but thinks it more likely that he was anguished because he didn't have anything to show.

Such anecdotes offer early evidence of the intensity that would become Stallman's chief trademark throughout life. When other kids came to the table, Stallman stayed in his room and read. When other kids played Johnny Unit as, Stallman played spaceman. "I was weird," Stallman says, summing up his early years succinctly in a 1999 interview. "After a certain age, the only friends I had were teachers."[12] Stallman was not ashamed of his weird characteristics, distinguishing them from the social ineptness that he did regard as a failing. However, both contributed together to his social exclusion.

Although it meant courting more run-ins at school, Lippman decided to indulge her son's passion. By age 12, Richard was attending science camps during the summer and private school during the school year. When a teacher recommended her son enroll in the Columbia Science Honors Program, a post-Sputnik program designed for gifted middle- and high-school students in New York City, Stallman added to his

article and asked if she could give me information on her company's e-book policies. Tracy promised to get back to me.

Eager to get started, I decided to call Stallman anyway and mention the book idea to him. When I did, he expressed immediate interest and immediate concern. "Did you read my essay on e-books?" he asked.

154

When I told him, yes, I had read the essay and was waiting to hear back from the publisher, Stallman laid out two conditions: he didn't want to lend support to an e-book licensing mechanism he fundamentally opposed, and he didn't want to come off

155

---

extracurriculars and was soon commuting uptown to the Columbia University campus on Saturdays.

Dan Chess, a fellow classmate in the Columbia Science Honors Program, recalls Richard Stallman seeming a bit weird even among the students who shared a similar lust for math and science. "We were all geeks and nerds, but he was unusually poorly adjusted," recalls Chess, now a mathematics professor at Hunter College. "He was also smart as shit. I've known a lot of smart people, but I think he was the smartest person I've ever known."

Seth Breidbart, a fellow Columbia Science Honors Program alumnus, offers bolstering testimony. A computer programmer who has kept in touch with Stallman thanks to a shared passion for science fiction and science-fiction conventions, he recalls the 15-year-old, buzz-cut-wearing Stallman as "scary," especially to a fellow 15-year-old.

"It's hard to describe," Breidbart says. "It wasn't like he was unapproachable. He was just very intense. [He was] very knowledgeable but also very hardheaded in some ways."

Such descriptions give rise to speculation: are judgment-laden adjectives like "intense" and "hardheaded" simply a way to describe traits that today might be categorized under juvenile behavioral disorder? A December, 2001, *Wired* magazine article titled "The Geek Syndrome" paints the portrait of several scientifically gifted children diagnosed with high-functioning autism or Asperger Syndrome. In many ways, the parental recollections recorded in the *Wired* article are eerily similar to the ones offered by Lippman. Stallman also speculates about this. In the interview for a 2000 profile for the *Toronto Star*, Stallman said he wondered if he were "borderline autistic." The article inaccurately cited the speculation as a certainty.[13]

Such speculation benefits from the fast and loose nature of most so-called "behavioral disorders" nowadays, of course. As Steve Silberman, author of "The Geek Syndrome," notes, American psychiatrists have only recently come to accept Asperger Syndrome as a valid umbrella term covering a wide set of behavioral traits. The traits range from poor motor skills and poor socialization to high intelligence and an almost obsessive affinity for numbers, computers, and ordered systems.[14]

In an effort to drive that image home, Moglen reflects on a shared moment in the spring of 2000. The success of the VA Linux IPO was still resonating in the business media, and a half dozen issues related to free software were swimming through the news. Surrounded by a swirling hurricane of issues and stories each begging for comment, Moglen recalls sitting down for lunch with Stallman and feeling like a castaway dropped into the eye of the storm. For the next hour, he says, the conversation calmly revolved around a single topic: strengthening the GPL.

"We were sitting there talking about what we were going to do about some problems in Eastern Europe and what we were going to do when the problem of the ownership of content began to threaten free software," Moglen recalls. "As we were talking, I briefly thought about how we must have looked to people passing by. Here we are, these two little bearded anarchists, plotting and planning the next steps. And, of course, Richard is plucking the knots from his hair and dropping them in the soup and behaving in his usual way. Anybody listening in on our conversation would have thought we were crazy, but I knew: I knew the revolution's right here at this table. This is what's making it happen. And this man is the person making it happen."

Moglen says that moment, more than any other, drove home the elemental simplicity of the Stallman style.

"It was funny," recalls Moglen. "I said to him, 'Richard, you know, you and I are the two guys who didn't make any money out of this revolution.' And then I paid for the lunch, because I knew he didn't have the money to pay for it."[135]

as lending support. "I don't want to participate in anything that makes me look like a hypocrite," he said.

For Stallman, the software issue was secondary to the copyright issue. He said he was willing to ignore whatever software the publisher or its third-party vendors em-

156

What I'd love to do, however, is read more - and I don't
think I'm alone. Do you think there is more information
and/or sources out there to expand and update your
interview and adapt it into more of a profile of Stallman?
Perhaps including some more anecdotal information about
his personality and background that might really interest
and enlighten readers outside the more hardcore
programming scene?

Tracy ended the email with a request that I give her a call to discuss the idea further. I did just that. Tracy told me her company was launching a new electronic book line, and it wanted stories that appealed to an early-adopter audience. The e-book format was 30,000words, about 100 pages, and she had pitched her bosses on the idea of profiling a major figure in the hacker community. Her bosses liked the idea, and in the process of searching for interesting people to profile,she had come across my BeOpen interview with Stallman. Hence her email to me.

That's when Tracy asked me: would I be willing to expand the interview into a full-length feature profile?

My answer was instant: yes. Before accepting it, Tracy suggested I put together a story proposal she could show her superiors. Two days later, I sent her a polished proposal. A week later, Tracy sent me a follow up email. Her bosses had given it the green light.

I have to admit, getting Stallman to participate in an e-book project was an afterthought on my part. As a reporter who covered the open source beat, I knew Stallman was a stickler. I'd already received a half dozen emails at that point upbraiding me for the use of "Linux" instead of "GNU/Linux."

Then again, I also knew Stallman was looking for ways to get his message out to the general public. Perhaps if I presented the project to him that way, he would be more receptive. If not, I could always rely upon the copious amounts of documents, interviews, and recorded online conversations Stallman had left lying around the Internet and do an unauthorized biography.

During my research, I came across an essay titled "Freedom - Or Copyright?" Written by Stallman and published in the June, 2000,edition of the MIT *Technology Review*, the essay blasted e-books for an assortment of software sins. Not only did readers have to use proprietary software programs to read them, Stallman lamented, but the methods used to prevent unauthorized copying were overly harsh. Instead of downloading a transferable HTML or PDF file, readers downloaded an encrypted file. In essence, purchasing an e-book meant purchasing a nontransferable key to unscramble the encrypted content. Any attempt to open a book's content without an authorized key constituted a criminal violation of the Digital Millennium Copyright Act, the 1998 law designed to bolster copyright enforcement on the Internet. Similar penalties held for readers who converted a book's content into an open file format, even if their only intention was to read the book on a different computer in their home. Unlike a normal book, the reader no longer held the right to lend, copy, or resell an e-book. They only had the right to read it on an authorized machine, warned Stallman:

We still have the same old freedoms in using paper books.But if e-books replace printed books, that exception will do little good. With "electronic ink," which makes it possible to download new text onto an apparently printed piece of paper, even newspapers could become ephemeral. Imagine:no more used book stores; no more lending a book to your friend; no more borrowing one from the public library - no more "leaks" that might give someone a chance to read without paying. (And judging from the ads for Microsoft Reader, no more anonymous purchasing of books either.)This is the world publishers have in mind for us.[136]

[10]One of the major background sources for this chapter was the interview "Richard Stallman: High

ployed just so long as the company specified within the copyright that readers were free to make and distribute verbatim copies of the e-book's content. Stallman pointed to Stephen King's *The Plant* as a possible model. In June, 2000, King announced on his official web site that he was self-publishing *The Plant* in serial form. According to the announcement, the book's total cost would be $13, spread out over a series of $1 installments. As long as at least 75% of the readers paid for each chapter, King promised to continue releasing new installments. By August, the plan seemed to be working, as King had published the first two chapters with a third on the way.

"I'd be willing to accept something like that," Stallman said. "As long as it also permitted   900

School Misfit, Symbol of Free Software, MacArthur-Certified Genius" by Michael Gross, author of the 1999 book Talking About My Generation , a collection of interviews with notable personalities from the so-called "Baby Boom" generation. Although Stallman did not make it into the book, Gross published the interview as an online supplement to the book's web site. The URL for the interview has changed several times since I first came across it. According to various readers who have gone searching for it, you can now find the interview at

`‹http://www.mgross.com/MoreThgsChng/interviews/stallman1.html.`

`›`

Such comments reflect the sense of humor that comes with raising a child prodigy. Make no mistake,

for every story Lippman hears and reads about her son's stubbornness and unusual behavior, she can deliver at least a dozen in return.

"He used to be so conservative," she says, throwing up her hands in mock exasperation. "We used

to have the worst arguments right here at this table. I was part of the first group of public city school teachers that struck to form a union, and Richard was very angry with me. He saw unions as corrupt. He was also very opposed to social security. He thought people could make much more money investing it on their own. Who knew that within 10 years he would become so idealistic? All I remember is his stepsister coming to me and saying, 'What is he going to be when he grows up? A fascist?'"[11]

As a single parent for nearly a decade - she and Richard's father, Daniel Stallman, were married in 1948,

divorced in 1958, and split custody of their son afterwards - Lippman can attest to her son's aversion to authority. She can also attest to her son's lust for knowledge. It was during the times when the two forces intertwined, Lippman says, that she and her son experienced their biggest battles.

"It was like he never wanted to eat," says Lippman, recalling the behavior pattern that set in around age

Epilogue from Sam Williams: Crushing Loneliness

[RMS: Because this chapter is so personally from Sam Williams, I have indicated all changes to the text with square brackets or ellipses, and I have made such changes only to clear up technical or legal points,and to remove passages that I found to be hostile and devoid of information. I have also added notes labeled 'RMS:' to respond to certain points. Williams has also changed the text of this chapter; changes made by Williams are not explicitly indicated.]

Writing the biography of a living person is a bit like producing a play. The drama in front of the curtain often pales in comparison to the drama backstage.

In *The Autobiography of Malcolm X*, Alex Haley gives readers a rare glimpse of that backstage drama. Stepping out of the ghostwriter role, Haley delivers the book's epilogue in his own voice. The epilogue explains how a freelance reporter originally dismissed as a "tool" and"spy" by the Nation of Islam spokesperson managed to work through personal and political barriers to get Malcolm X's life story on paper.

While I hesitate to compare this book with *The Autobiography of Malcolm X*, I do owe a debt of gratitude to Haley for his candid epilogue. Over the last 12 months, it has served as a sort of instruction manual on how to deal with a biographical subject who has built an entire career on being disagreeable. [RMS: I have built my career on saying no to things others accept without much question, but if I sometimes seem or am disagreeable, it is not through specific intention.] From the outset, I envisioned closing this

verbatim copying." [RMS: As I recall, I also raised the issue of encryption; the text two paragraphs further down confirms this. I would not have agreed to publish the book in a way that *required* a non-free program to read it.]

I forwarded the information to Tracy. Feeling confident that she and I might be able to work out an equitable arrangement, I called up Stallman and set up the first interview for the book. Stallman agreed to the interview without making a second inquiry into the status issue. Shortly after the first interview, I raced to set up a second interview (this one in Kihei), squeezing it in before Stallman headed off on a 14-day vacation to Tahiti. [RMS: That was not a pure vacation; I gave a speech there too.]   901

It was during Stallman's vacation that the bad news came from Tracy. Her company's legal department didn't want to adjust its [license] notice on the e-books. Readers who wanted to make their books transferable would [first have to crack the encryption code, to be able to convert the book to a free, public format such as HTML. This would be illegal and they might face criminal penalties.]   902

With two fresh interviews under my belt, I didn't see any way to write the book without resorting to the new material. I quickly set up a trip to New York to meet with my agent and with Tracy to see if there was a compromise solution.   903

When I flew to New York, I met my agent, Henning Guttman. It was our first face-to-   904

---

biography with a similar epilogue, both as an homage to Haley and as a way to let readers know how this book came to be.

The story behind this story starts in an Oakland apartment, winding its way through the various locales mentioned in the book - Silicon Valley, Maui, Boston, and Cambridge. Ultimately, however, it is a tale of two cities: New York, New York, the book-publishing capital of the world, and Sebastopol, California, the book-publishing capital of Sonoma County.

The story starts in April, 2000. At the time, I was writing stories for the ill-fated web site BeOpen.com. One of my first assignments was a phone interview with Richard M. Stallman. The interview went well, so well that Slashdot (‹http://www.slashdot.org›), the popular "news for nerds" site owned by VA Software, Inc. (formerly VA LinuxSystems and before that, VA Research), gave it a link in its daily list of feature stories. Within hours, the web servers at BeOpen were heating up as readers clicked over to the site.

For all intents and purposes, the story should have ended there. Three months after the interview, while attending the O'Reilly Open Source Conference in Monterey, California, I received the following email message from Tracy Pattison, foreign-rights manager at a large New York publishing house:

To: ‹sam@BeOpen.com›

Subject: RMS Interview Date: Mon, 10 Jul 2000 15:56:37 - 0400

Dear Mr. Williams,

I read your interview with Richard Stallman on BeOpen
with great interest. I've been intrigued by RMS and his
work for some time now and was delighted to find your
piece which I really think you did a great job of capturing
some of the spirit of what Stallman is trying to do with
GNU-Linux and the Free Software Foundation.

face meeting, and Henning seemed pessimistic about our chances of forcing a compromise, at least on the publisher's end. The large, established publishing houses already viewed the e-book format with enough suspicion and weren't in the mood to experiment with copyright language that made it easier for readers to avoid payment. As an agent who specialized in technology books, however,Henning was intrigued by the novel nature of my predicament. I told him about the two interviews I'd already gathered and the promise not to publish the book in a way that made Stallman "look like a hypocrite." Agreeing that I was in an ethical bind, Henning suggested we make that our negotiating point.

Barring that, Henning said, we could always take the carrot-and-stick approach. The carrot would be the publicity that came with publishing an e-book that honored the hacker community's internal ethics. The stick would be the risks associated with publishing an e-book that didn't. Nine months before Dmitry Sklyarov became an Internet *cause célèbre*, we knew it was only a matter of time before an enterprising programmer revealed how to hack e-books. We also knew that a major publishing house releasing an [encrypted] e-book on Richard M. Stallman was the software equivalent of putting "Steal This E-Book" on the cover.

After my meeting with Henning, I called Stallman. Hoping to make the carrot more enticing, I discussed a number of potential compromises. What if the publisher released the book's content under a[dual] license, something similar to what Sun Microsystems had done with Open Office, the free software desktop applications suite? The publisher could then release DRM-restricted[137] versions of the e-book under [its usual] format, taking advantage of all the bells and whistles that went with the e-book software, while releasing the copyable version under a less aesthetically pleasing HTML format.

Stallman told me he didn't mind the [dual-license] idea, but he did dislike the idea of making the freely copyable version inferior to the restricted version. Besides, he said [on second thought, this case was different precisely because he had] a way to control the outcome. He could refuse to cooperate.

[RMS: The question was whether it would be wrong for me to agree to the restricted version. I can endorse the free version of Sun's Open Office, because it is free software and much better than nothing,while at the same time I reject the non-free version. There is no self- contradiction here, because Sun didn't need or ask my approval for the non- free version; I was not responsible for its existence. In this case, if I had said yes to the non-freely-copyable version, the onus would fall on me.]

I made a few more suggestions with little effect. About the only thing I could get out of Stallman was a concession [RMS: i.e., a further compromise] that the e-book's [license] restrict all forms of file sharing to "noncommercial redistribution."

Before I signed off, Stallman suggested I tell the publisher that I'd promised Stallman that the work would be [freely sharable]. I told Stallman I couldn't agree to that statement [RMS: though it was true,since he had accepted my conditions at the outset] but that I

905

906

907

908

909

910

---

[137]RMS: Williams wrote "commercial" here, but that is a misnomer, since it means "connected with business." All these versions would be commercial if a company published them.

did view the book as unfinishable without his cooperation. Seemingly satisfied,Stallman hung up with his usual sign-off line: "Happy hacking."

Henning and I met with Tracy the next day. Tracy said her company was willing to publish copyable excerpts in a unencrypted format but would limit the excerpts to 500 words. Henning informed her that this wouldn't be enough for me to get around my ethical obligation to Stallman. Tracy mentioned her own company's contractual obligation to online vendors such as Amazon.com. Even if the company decided to open up its e-book content this one time, it faced the risk of its partners calling it a breach of contract. Barring a change of heart in the executive suite or on the part of Stallman, the decision was up tome. I could use the interviews and go against my earlier agreement with Stallman, or I could plead journalistic ethics and back out of the verbal agreement to do the book. 911

Following the meeting, my agent and I relocated to a pub on Third Ave. I used his cell phone to call Stallman, leaving a message when nobody answered. Henning left for a moment, giving me time to collect my thoughts. When he returned, he was holding up the cell phone. 912

"It's Stallman," Henning said. 913

The conversation got off badly from the start. I relayed Tracy's comment about the publisher's contractual obligations. 914

"So," Stallman said bluntly. "Why should I give a damn about their contractual obligations?" 915

Because asking a major publishing house to risk a legal battle with its vendors over a 30,000-word e-book is a tall order, I suggested. [RMS: His unstated premise was that I couldn't possibly refuse this deal for mere principle.] 916

"Don't you see?" Stallman said. "That's exactly why I'm doing this. I want a signal victory. I want them to make a choice between freedom and business as usual." 917

As the words "signal victory" echoed in my head, I felt my attention wander momentarily to the passing foot traffic on the sidewalk. Coming into the bar, I had been pleased to notice that the location was less than half a block away from the street corner memorialized in the 1976 Ramones song, "53rd and 3rd," a song I always enjoyed playing in my days as a musician. Like the perpetually frustrated street hustler depicted in that song, I could feel things falling apart as quickly as they had come together. The irony was palpable. After weeks of gleefully recording other people's laments, I found myself in the position of trying to pull off the rarest of feats: a Richard Stallman compromise. When I continued hemming and hawing, pleading the publisher's position and revealing my growing sympathy for it,Stallman, like an animal smelling blood, attacked. 918

"So that's it? You're just going to screw me? You're just going to bend to their will?" 919

[RMS: The quotations show that Williams' interpretation of this conversation was totally wrong. He compares me to a predator, but I was only saying no to the deal he was badgering me to accept.I had already made several compromises, some described 920

above; I just refused to compromise my principles entirely away. I often do this; people who aren't satisfied say I "refused to compromise at all, "but that is an exaggeration; see ‹http://www.gnu.org/philosophy/compromise.html›. Then I feared he was going to disregard the conditions he had previously agreed to, and publish the book with DRM despite my refusal. What I smelled was not his "blood" but possible betrayal.]

I brought up the issue of a dual-copyright again.   921

"You mean license," Stallman said curtly.   922

"Yeah, license. Copyright. Whatever," I said, feeling suddenly like a wounded tuna   923
trailing a rich plume of plasma in the water.

"Aw, why didn't you just fucking do what I told you to do!" he shouted. [RMS: I think this   924
quotation was garbled, both because using "fucking" as an adverb was never part of
my speech pattern, and because the words do not fit the circumstances. The words he
quotes are a rebuke to a disobedient subordinate. I felt he had an ethical obligation, but
he was not my subordinate, and I would not have spoken to him as one. Using notes
rather than a recorder, he could not reliably retain the exact words.]

I must have been arguing on behalf of the publisher to the very end, because in my   925
notes I managed to save a final Stallman chestnut: "I don't care. What they're doing
is evil. I can't support evil. Goodbye." [RMS: It sounds like I had concluded that he
would never take no for an answer, and the only way to end the conversation without
accepting his proposition was to hang up on him.]

As soon as I put the phone down, my agent slid a freshly poured Guinness to me.   926
"I figured you might need this," he said with a laugh. "I could see you shaking there
towards the end."

I was indeed shaking. The shaking wouldn't stop until the Guinness was more than   927
halfway gone. It felt weird, hearing myself characterized as an emissary of "evil." [RMS:
My words as quoted criticize the publisher, not Williams personally. If he took it person-
ally, perhaps that indicates he was starting to take ethical responsibility for the deal he
had pressed me to accept.] It felt weirder still, knowing that three months before, I was
sitting in an Oakland apartment trying to come up with my next story idea. Now, I was
sitting in a part of the world I'd only known through rock songs, taking meetings with
publishing executives and drinking beer with an agent I'd never even laid eyes on until
the day before. It was all too surreal, like watching my life reflected back as a movie
montage.

About that time, my internal absurdity meter kicked in. The initial shaking gave way to   928
convulsions of laughter. To my agent, I must have looked like a another fragile author
undergoing an untimely emotional breakdown. To me, I was just starting to appreciate
the cynical beauty of my situation. Deal or no deal, I already had the makings of a
pretty good story. It was only a matter of finding a place to tell it. When my laughing
convulsions finally subsided, I held up my drink in a toast.

"Welcome to the front lines, my friend," I said, clinking pints with my agent. "Might as   929
well enjoy it."

If this story really were a play, here's where it would take a momentary, romantic inter-
lude. Disheartened by the tense nature of our meeting, Tracy invited Henning and me
to go out for drinks with her and some of her coworkers. We left the bar on Third Ave.,
headed down to the East Village, and caught up with Tracy and her friends.

Once there, I spoke with Tracy, careful to avoid shop talk. Our conversation was pleas-
ant, relaxed. Before parting, we agreed to meet the next night. Once again, the con-
versation was pleasant, so pleasant that the Stallman e-book became almost a distant
memory.

When I got back to Oakland, I called around to various journalist friends and acquain-
tances. I recounted my predicament. Most upbraided me for giving up too much ground
to Stallman in the pre-interview negotiation. [RMS: Those who have read the whole
book know that I would never have dropped the conditions.] A former j-school professor
suggested I ignore Stallman's "hypocrite" comment and just write the story. Reporters
who knew of Stallman's media-savviness ex-pressed sympathy but uniformly offered
the same response: it's your call.

I decided to put the book on the back burner. Even with the interviews, I wasn't making
much progress. Besides, it gave me a chance to speak with Tracy without running
things past Henning first.By Christmas we had traded visits: she flying out to the west
coast once, me flying out to New York a second time. The day before New Year's Eve,
I proposed. Deciding which coast to live on, I picked New York. By February, I packed
up my laptop computer and all my research notes related to the Stallman biography,
and we winged our way to JFK Airport. Tracy and I were married on May 11. So much
for failed book deals.

During the summer, I began to contemplate turning my interview notes into a magazine
article. Ethically, I felt in the clear doing so,since the original interview terms said nothing
about traditional print media. To be honest, I also felt a bit more comfortable writing
about Stallman after eight months of radio silence. Since our telephone conversation
in September, I'd only received two emails from Stallman.Both chastised me for using
"Linux" instead of "GNU/Linux" in a pair of articles for the web magazine *Upside Today*.
Aside from that, I had enjoyed the silence. In June, about a week after the New York
University speech, I took a crack at writing a 5,000-word magazine-length story about
Stallman. This time, the words flowed. The distance had helped restore my lost sense
of emotional perspective, I suppose.

In July, a full year after the original email from Tracy, I got a call from Henning. He
told me that O'Reilly & Associates, a publishing house out of Sebastopol, California,
was interested in the running the Stallman story as a biography. [RMS: I have a vague
memory that I suggested contacting O'Reilly, but I can't be sure after all these years.]
The news pleased me. Of all the publishing houses in the world, O'Reilly, the same
company that had published Eric Raymond's *The Cathedral and the Bazaar*, seemed
the most sensitive to the issues that had killed the earlier e-book. As a reporter, I had
relied heavily on the O'Reilly book *Open Sources* as a historical reference. I also knew
that various chapters of the book, including a chapter written by Stallman, had been