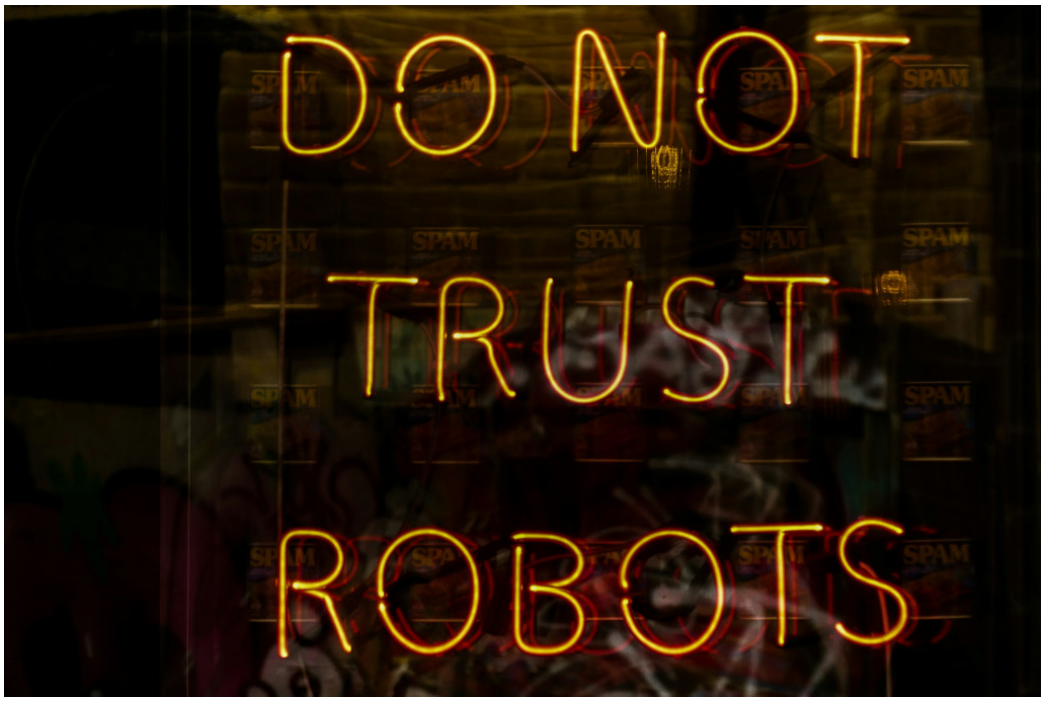
Issue 079: Trust

De Programmatica Ipsum

2025-04-07

Contents

Issue 079: Trust



By Adrian Kosmaczewski, April 7th, 2025

Welcome to the 79th issue of *De Programmatica Ipsum*, about *Trust*.

In this edition:

- We explore trust and its importance[1] in human society and culture.

[1] https://deprogrammaticaipsum.com/who-do-you-trust/

- In the Library section[2], we review "Geekonomics" by David Rice[3].
- In our Vidéothèque section[4], we watch a video on the Veritasium channel where Derek Muller hacks Linus Sebastian[5]'s phone.

Download this issue in DRM-free PDF[6] or EPUB[7] format, and read it on your preferred device.

We would like to thank our patrons who generously contribute every month (or have contributed in the past) to our work and help us run this magazine. Thank you so much! In alphabetical order: Adam Guest, Adrian Tineo Cabello, Benjamin Sheldon, Christopher Nascone, Colin Powell, Franz Lucien Moersdorf, Guillermo Ramos Álvarez, Jean-Paul de Vooght, Dr. Juande Santander-Vela, Patryk Matuszewski, Paul Hudson, Quico Moya, Roger Turner, Szymon Licau, and countless more leaving anonymous tips every month.

Enjoy this issue! Please subscribe to our free newsletter[8] to stay updated about new releases, share the articles on social media, or contribute[9] if you would like to support our work with a donation via Liberapay[10].

Cover photo by Nick Fewings[11] on Unsplash[12].

---

[2] https://deprogrammaticaipsum.com/category/library/

[3] https://deprogrammaticaipsum.com/david-rice/

[4] https://deprogrammaticaipsum.com/category/videotheque/

[5] https://deprogrammaticaipsum.com/derek-muller-linus-sebastian/

[6] https://deprogrammaticaipsum.com/pdf/issue-079-trust.pdf

[7] https://deprogrammaticaipsum.com/epub/issue-079-trust.epub

[8] https://deprogrammaticaipsum.com/newsletter/

[9] https://deprogrammaticaipsum.com/contribute/

[10] https://liberapay.com/akosma/donate

[11] https://unsplash.com/@jannerboy62?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[12] https://unsplash.com/photos/text-C2J92BO3qTw?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Who Do You Trust?

By Adrian Kosmaczewski, April 7th, 2025

Have you ever asked yourself why do we live in society, surrounded by others? Why have humans gathered in slowly-growing communities at the dawn of our civilization? It turns out that humans are terrible when it comes at surviving alone. We are not particularly strong or fast, which means that we can succumb at the attack of any predator. We cannot fly by flapping our arms, we cannot swim at great speed in the sea. Our bod-

ies suffer plenty of ailments and are prone to quite a few lethal illnesses. To add insult to injury, we carry a big, complex, and fragile organ inside our skulls that requires quite a few calories a day to function properly. Even worse, said organ is supposedly at the origin of a psyche (a novel concept in nature, apparently) with an annoying tendency towards anger, fear, and depression.

So, get together we do. Turns out, there were several advantages in doing so; to begin with, work specialization is a huge one. Some people are better at hunting, some others at growing vegetables, some others at teaching, some others at making clothes, some others at invoking the spirits of the forest. Getting together meant that we could be more than just the sum of the members of our communities. We could have more things. We could learn from others. And, yes, we could survive better.

All of this became possible because, at some point around a dozen millennia ago, we decided to trust each other.

Our modern society, however, seems to have forgotten this basic principle. We forgot that we were able to use empathy as a superpower, and through the simple act of taking care of one another, to build real webs of trust. Instead, human communities of various scales are crippled today with decaying infrastructure, rampant corruption, abysmal crime statistics, lack of professional, economical, and educational opportunities, war, political or religious prosecution, or even worse, with full-blown, state-driven, and technologically-enabled annihilation.

Faced with such a situation, many human beings (among which you can include the author of these words) decided to just pack up and move to other valleys, where the grass is supposedly greener; not all can afford or are allowed to do so, however. On the extreme opposite side of this spectrum, billionaires on a quest against empathy plan on ~~polluting~~ colonizing other planets to escape the mayhem they have created. Somewhere in the middle, the 0.1% of mankind who cannot (yet) afford a spaceship orchestrate their retreats to secure locations ranging from a private island in the Pacific Ocean, to Dubai, or to a secluded and tax-friendly ski resort in the Swiss Alps.

As a result, we are all either fleeing our society, or dreaming of doing it someday. So much for trusting each other. Even worse, this inconvenient feeling named mistrust has translated to each facet of our human experience, including, as you can imagine, our software.

Tim Burton's "Batman" movie[13] features a scene where the Joker, played by the im-

[13] https://en.wikipedia.org/wiki/Batman_(1989_film)

mense Jack Nicholson, and with Prince's music in the background, is throwing dollar bills to the same Gotham City crowd he is planning to kill *en masse*. He then asks the trillion-dollar question: "Who do you trust?"[14]

As a testimony to the brilliance of Burton, it is hard not to see in this short clip an allegory of our current world.

No Small Vulnerabilities

During the last decade, a troublesome one if we compare it to not-so-recent history, we can enumerate quite a few occasions in which software dropped its status from the pinnacle of human creativity, to the bottomless pit of the most abject hatred for mankind. And we, software workers and software practitioners, are the only ones to blame for this.

Let us recap a few highlights, beginning with 2015, when Volkswagen was caught tweaking its own diesel car software to cheat during emission tests. Somebody wrote, tested, and deployed that code.

Then during the same year, Uber engineers inserted special instructions in their apps to evade law enforcement in cities where it faced regulatory issues, identifying officials and preventing them from hailing Uber rides. Somebody wrote, tested, and deployed that code.

Boeing's faulty (and even non-documented) Maneuvering Characteristics Augmentation System software caused two deadly crashes (Lion Air Flight 610 in 2018, and Ethiopian Airlines Flight 302 in 2019), killing 346 people in total. Somebody wrote, tested, and deployed that code.

From 2017 to 2020, Apple intentionally slowed down older iPhones via software updates, without informing users, to push them toward buying new models. By the way, you might want to collect the 20 USD that Apple owes you[15] because of a privacy lawsuit settlement related to Siri early this year. Somebody wrote, tested, and deployed that code.

Speaking about Apple, let us not forget about the aptly named `trustd` apocalypse[16] which prevented macOS Big Sur users of opening third-party applications in November

[14] https://www.youtube.com/watch?v=50OJ0atfipo

[15] https://www.wired.com/story/apple-95-million-siri-privacy-lawsuit/

[16] https://www.sentinelone.com/blog/what-happened-to-my-mac-apples-ocsp-apocalypse/

2020.

That same year, allegedly Russian hackers compromised software made by a company named SolarWinds, inserting backdoors that effectively worked against numerous government agencies and private companies (effectively for the hackers, that is). According to the SEC filing by SolarWinds, this supply chain attack affected approximately 18'000 organizations. We will never know the real number, and I do not think we really want to know it.

And let us not even get started on the Facebook and Cambridge Analytica affair, shall we. As we publish this article, we are still living the consequences of Mark Zuckerberg's childish inability and unwillingness to accept the rules of a trusting society. Instead, he prefers to prevent the release of a book[17] that candidly reveals the inner workings of a truly psychotic and untrustable organization, ruled by psychotic and untrustable billionaires.

What about exploits? They have become a yearly tradition like having birthday cakes; take *any* piece of software (open-source or not) or *any* hardware gizmo, ask a few security experts to scrutinize it for a while, and then give a fancy name to whatever vulnerability they find. Thus were born Heartbleed[18] and FLUSH+RELOAD[19] in 2014, ARMageddon[20] and DRAMA[21] in 2016, Spectre[22] and Meltdown[23] in 2018, PACMAN[24] in 2022, Downfall[25] in 2023, and ZenHammer[26] in 2024.

We are still waiting for the winner of 2025. These exploits have become such a common fixture in our daily life that they not only feature a USENIX presentation or a paper on arXiv[27], but even their own website with their own domain name, with a nice logo or plush mascot designed specially for them. If you cannot trust our software, at least support us on Patreon or buy our swag.

---

[17] https://en.wikipedia.org/wiki/Careless_People
[18] https://en.wikipedia.org/wiki/Heartbleed
[19] https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/yarom
[20] https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/lipp
[21] https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/pessl
[22] https://en.wikipedia.org/wiki/Spectre_(security_vulnerability)
[23] https://en.wikipedia.org/wiki/Meltdown_(security_vulnerability)
[24] https://dl.acm.org/doi/10.1145/3470496.3527429
[25] https://en.wikipedia.org/wiki/Downfall_(security_vulnerability)
[26] https://github.com/comsec-group/zenhammer
[27] https://www.wired.com/story/inside-arxiv-most-transformative-code-science/

No Plausible Deniability

The word "trust" has been very much abused through the years. These days we have the Zero-Trust Architecture[28], and a journalists' association called "The Trust Project"[29]. There are "antitrust lawsuits" which suggests the existence of a business meaning for the word "trust"[30]. During the late 1950s Johnny Carson presented a TV show called "Who Do You Trust?"[31]; and yeah, all US dollar bills feature the well-known "In God We Trust" motto.

Never mind that Sibert, Porras, and Lindell had warned us[32], already in 1995, that the Intel 80x86 processor architecture was not to be trusted from a security standpoint. Apparently Bill Gates forgot to read that paper, and then five years late he had to come up with a thing he called "Trustworthy Computing"[33] because Back Orifice[34] was undermining trust in his valuable Windows operating system.

Thankfully, Joanna Rutkowska did read Sibert, Porras, and Lindell's paper, and included a severe and cold-blooded definition of the concept of trust in a 2015 paper called "Intel x86 considered harmful"[35]. Sensitive souls beware:

> The word "trusted" is a sneaky and confusing term: many people get a warm fuzzy feeling when they read it, and it is treated as a good thing. In fact the opposite is true. Anything that is "trusted" is a potentially lethal enemy of any secure system. Any component that we (are forced to) consider "trusted" is an ideal candidate to compromise the whole system, should this trusted component turn out to be buggy or backdoored. That property (i.e. the ability to destroy the system's security) is in fact the definition of the term "trusted".

An IBM engineer had said already in the 1970s that a computer could never be held accountable[36], so I do not see what is so surprising here; particularly now that the UK

---

[28]https://en.wikipedia.org/wiki/Zero_trust_architecture

[29]https://thetrustproject.org/

[30]https://en.wikipedia.org/wiki/Trust_(business)

[31]https://www.youtube.com/watch?v=tSnw10XS2rE

[32]https://dl.acm.org/doi/10.5555/882491.884240

[33]https://web.archive.org/web/20150626172158/http://archive.wired.com/techbiz/media/news/2002/01/49826

[34]https://en.wikipedia.org/wiki/Back_Orifice

[35]https://blog.invisiblethings.org/papers/2015/x86_harmful.pdf

[36]https://simonwillison.net/2025/Feb/3/a-computer-can-never-be-held-accountable/

government is weakening safety worldwide[37] with its actions, the current software crisis is, just like most economic crisis, one caused by mistrust.

Ken Thompson[38], of Unix, C, UTF-8, and Go fame, declared during his 1983 ACM Turing Award speech called "Reflections on Trusting Trust"[39] that

> You can't trust code that you did not totally create yourself. (Especially code from companies that employ people like me.) No amount of source-level verification or scrutiny will protect you from using untrusted code. In demonstrating the possibility of this kind of attack, I picked on the C compiler. I could have picked on any program-handling program such as an assembler, a loader, or even hardware microcode. As the level of program gets lower, these bugs will be harder and harder to detect. A well-installed microcode bug will be almost impossible to detect.

These words seem oddly prophetic when we remember the XZ Utils backdoor[40] discovered merely one year ago. Remember to thank Andres Freund for performing what Ken Thompson considered almost impossible 40 years ago.

We proposed an oath for software developers[41] a few years ago, which included the following statement that we think perfectly summarizes our feelings in the matter of trust:

> I will remember that I do not merely create a system or implement an algorithm, but I create systems for the highest benefit of society, who will have to use it and who will store their most confidential information within. My responsibility includes these related problems, if I am to solve adequately the problem at hand.

Nothing Left For Me To Do

Let us quote Sting[42] to finish this admittedly depressing article:

> You could say I lost my faith in science and progress
> You could say I lost my belief in the Holy Church

[37] https://blog.thenewoil.org/how-the-uk-is-weakening-safety-worldwide
[38] https://en.wikipedia.org/wiki/Ken_Thompson
[39] https://dl.acm.org/doi/10.1145/358198.358210
[40] https://en.wikipedia.org/wiki/XZ_Utils_backdoor
[41] https://deprogrammaticaipsum.com/primum-non-nocere/
[42] https://en.wikipedia.org/wiki/If_I_Ever_Lose_My_Faith_in_You

> You could say I lost my sense of direction
> And you could say all of this and worse but
> If I ever lose my faith in you
> There'd be nothing left for me to do

Empathy and trust in one another are the fragile glue that keeps our society in an ever-fragile equilibrium. We must acknowledge the fact that there is no gizmo, AI, protocol, library, programming language (no, not even Rust), platform, or mobile application to replace them, and in this acknowledgement resides our chance for survival as a species. When it comes to software, the answer to the question "Who Do You Trust?" should be a strong, resounding, and unanimous voice saying: "all of us".

In other words, TL;DR: Please give a shit.

Cover photo by Azzedine Rouichi[43] on Unsplash[44].

[43]https://unsplash.com/@rouichi?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[44]https://unsplash.com/photos/a-rock-formation-in-the-middle-of-a-desert-f_C_lFxqThI?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Derek Muller & Linus Sebastian

By Adrian Kosmaczewski, April 7th, 2025

Previous articles in this magazine have explored real-life examples highlighting the insecurity of our modern communications infrastructure. Regular readers might remember the anecdote[45] of an *impromptu* hacking lesson in the Universidad de Buenos Aires in the year 2000, where our teacher simply intercepted a phone call made by one of the students with a small handheld scanner. The same article goes on to describe how I was able to use a software tool called "CaptureNet" to sniff packets on port 1863 (used by MSN Messenger back in the day) and thus secretly read all the exchanges between my work colleagues in 2001.

[45] https://deprogrammaticaipsum.com/the-weakest-link/

Scary stuff, but here we are 24 years later, and the telecom trust situation has not improved. If anything, it has gotten worse, and therefore it has become a matter of national security, as shown by this month's Vidéothèque movie, "Exposing The Flaw In Our Phone System"[46], narrated and experienced by Derek Muller and Linus Sebastian.

I do not think we need an introduction of their respective channels, but here it goes anyway: "Veritasium"[47] is the brainchild of Derek Muller[48], a PhD in physics education, and a prolific science communicator. His channel regularly features science-related videos, with total viewership counts in the billions. A quick review of the most popular videos on Veritasium brings hallmarks like the 96 million black balls[49] on a reservoir, Derek waterproofing[50] himself with aerogel, the story of the man who accidentally killed the most people[51] in history, or a discussion about parallel worlds[52].

On the other hand, Linus Sebastian[53] is a Canadian YouTuber hosting "Linus Tech Tips"[54] since 2008, again with millions of subscribers and billions of views, and featuring videos about computer hardware and software.

Following the classic Veritasium style, Derek kicks off his video with a review of historical fragilities in the phone system, including Steve Jobs' and Steve Wozniak's attempt to call the Pope while impersonating Henry Kissinger[55], using a small device of their own creation called "Blue Box"[56].

Derek then proceeds to explain the current Signalling System No. 7[57] protocol, or SS7, used by global telecommunication companies worldwide to route calls and service information. And here lies the heart of the video: the protocol works like a "walled garden", where, as soon as given access to, any telecom operator can freely use, without further checks.

The problem is, not all operators are trustworthy. Actually, getting access to the net-

[46] https://www.youtube.com/watch?v=wVyu7NB7W6Y
[47] https://www.youtube.com/@veritasium
[48] https://en.wikipedia.org/wiki/Derek_Muller
[49] https://www.youtube.com/watch?v=uxPdPpi5W4o
[50] https://www.youtube.com/watch?v=GcdB5bFwio4
[51] https://www.youtube.com/watch?v=IV3dnLzthDA
[52] https://www.youtube.com/watch?v=kTXTPe3wahc
[53] https://en.wikipedia.org/wiki/Linus_Sebastian
[54] https://www.youtube.com/@LinusTechTips
[55] https://en.wikipedia.org/wiki/Henry_Kissinger
[56] https://en.wikipedia.org/wiki/Blue_box
[57] https://en.wikipedia.org/wiki/Signalling_System_No._7

work is surprisingly simple and cheap, even for an individual!

The case of Sheikha Latifa bint Mohammed Al Maktoum[58], daughter of the Prime Minister of the United Arab Emirates, is a frightening example of the vulnerability of the SS7 system. She made the headlines in 2018 when her failed escape to India was thwarted by the intervention of the FBI, who used the geolocation features of the SS7 protocol to pinpoint her location on a boat 50 miles away from the shores of Goa.

Using precisely those same flaws in the SS7 protocol, Derek (with the help of security experts Karsten Nohl[59] and Alexandre De Oliveira[60]) proceeds to "hack" Linus' smartphone, intercepting calls and SMS messages.

Precisely what I saw with my own eyes in 2000 in that fateful classroom in Buenos Aires.

A bewildered Linus tries to understand what is going on, and in minute 17:30[61] asks:

> So, the most important question I have now then is, what did you need to steal from me, in order to become me? Like, is this something you can social engineer out of my carrier, is this something that, I would need to accidentally leak a screenshot of my IMEI...?

Derek's response is as lethal as it is short:

> At the very simplest, all what we need is your phone number. That's it.

The face of Linus at minute 17:52[62] and his final reaction says it all:

> This is why we can't have nice things.

To add insult to injury, among the SMS messages that Derek receives on behalf of Linus, there is a two-factor authentication code for Linus' YouTube channel, providing full administrative access to the contents of one of the most popular on the platform.

What can we do to protect ourselves from this threat? Unfortunately, as Derek says, not much.

Karsten Nohl explains that the 5G protocol includes fixes to the well-known flaws of the SS7 system, but the costly (and hence slow) rollout of these capabilities means that

---

[58] https://en.wikipedia.org/wiki/Latifa_bint_Mohammed_Al_Maktoum_(born_1985)
[59] https://en.wikipedia.org/wiki/Karsten_Nohl
[60] https://ieeexplore.ieee.org/author/37089305673
[61] https://youtu.be/wVyu7NB7W6Y?t=1050
[62] https://youtu.be/wVyu7NB7W6Y?t=1072

the current state of things will most probably remain untouched for a decade, or even longer than that.

You have been warned: lower your expectations of trust, and protect yourself. With a bit of luck, the hack I witnessed in the year 2000 will not be possible anymore in 2045. But let us be honest, I am not holding my breath.

Watch this month's Vidéothèque movie, "Exposing The Flaw In Our Phone System" by Derek Muller & Linus Sebastian, on YouTube[63].

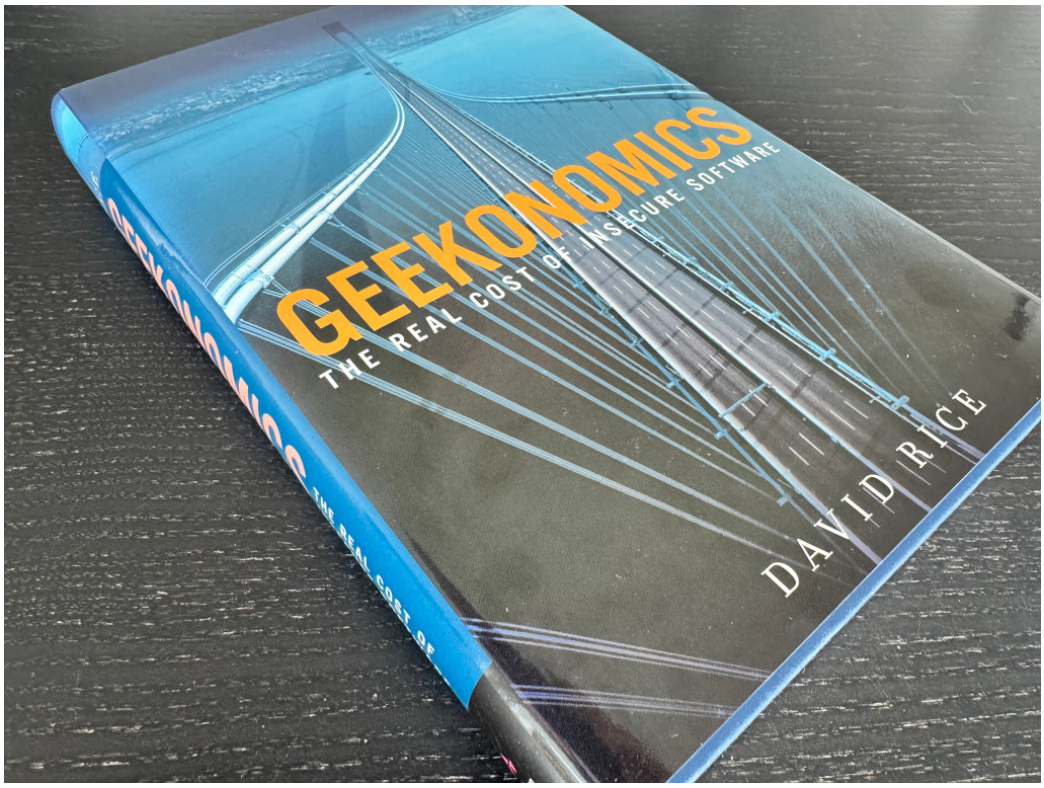
In the meantime, you might want to stop using SMS codes for your two-factor authentication, and use time-based one-time passwords[64] instead (sadly, not all online services have watched this month's Vidéothèque movie). Remember to switch to encrypted messaging apps like Signal[65] to chat with your family, colleagues, and friends as well.

In other words, TL;DR: Please give a shit.

Cover snapshot chosen by the author.

[63] https://www.youtube.com/watch?v=wVyu7NB7W6Y
[64] https://en.wikipedia.org/wiki/Time-based_one-time_password
[65] https://signal.org/

David Rice



By Adrian Kosmaczewski, April 7th, 2025

Our biggest problem is not the lack of books explaining in detail the fragility of our software-driven world; it is the fact that nobody reads them. Of course, every so often one of those titles rises to the top of The New York Times bestseller list, some celebrity adds it to their list of favorite books, the author takes a year promoting the book in a few talk shows here and there, and might even give some sold-out conference talks outside

their home country. They might even make a living out of said book. But as soon as a new shiny gizmo appears on the horizon, all the concerns raised by their work fade into obscurity, the state of the world degrades a bit more, and we are back in the starting blocks scratching our armpits and screaming like monkeys.

This has happened quite a few times in our modern literary history. We can enumerate some examples: "Cybersecurity and Cyberwar: What Everyone Needs to Know"[66] (2014) by Peter W. Singer and Allan Friedman; "Click Here to Kill Everybody: Security and Survival in a Hyper-connected World"[67] (2018) by Bruce Schneier; "The Art of Invisibility"[68] (2017) by the late Kevin Mitnick; "Code Version 2.0"[69] (2006) by Lawrence Lessig; "The Age of Surveillance Capitalism"[70] (2019) by Shoshana Zuboff; and finally the subject of this month's Library article, David Rice's 2007 "Geekonomics: The Real Cost of Insecure Software"[71].

The catchy title surfed over the popularity of a 2005 bestseller called "Freakonomics: A Rogue Economist Explores the Hidden Side of Everything"[72] by Steven Levitt and Stephen J. Dubner, which has since evolved into a complete franchise including more books, films, podcasts, and who knows what more.

The *-onomics* suffix has since become a common fixture of any book, software, website, or think tank that brings into the layman realm the complexity of our modern world. Hence, we have now Leadonomics[73], Quickonomics[74], Clearnomics[75], Priceonomics[76] ("In Data We Trust"), Growth-onomics[77], Eco-nomics[78] (pay attention to the dash in the name), DBnomics[79], Date-onomics[80], Stronomics[81] (whatever that is),

---

[66]https://whateveryoneneedstoknow.com/display/10.1093/wentk/9780199918096.001.0001/isbn-9780199918096

[67]https://www.schneier.com/books/click-here/

[68]https://www.mitnicksecurity.com/the-art-of-invisibility-mitnick-security

[69]https://lessig.org/product/codev2/

[70]https://en.wikipedia.org/wiki/The_Age_of_Surveillance_Capitalism

[71]https://www.oreilly.com/library/view/geekonomics-the-real/9780321477897/

[72]https://en.wikipedia.org/wiki/Freakonomics

[73]https://leadnomics.com/

[74]https://quickonomics.com/

[75]https://www.clearnomics.com/

[76]https://priceonomics.com/

[77]https://growth-onomics.com/

[78]https://archive.org/details/economicswhateve0000stro

[79]https://db.nomics.world/

[80]https://thepowermoves.com/date-onomics/

[81]https://stronomics.com/

IBM Turbonomic[82], Egg-onomics[83] (love the pun), and other[84] idiocies[85]. Do not worry, we will not rename this magazine "Programmeronomics" or anything like that, no matter how big the temptation is.

"Geekonomics", then, provides an insightful yet frightening view of the fragility of the world in 2007. Let us think about that for a minute: this book was released before the 2008 financial crisis, before the iPhone, before Uber and Airbnb, before Cambridge Analytica, before Zero-Trust architectures, before the pandemic, before ChatGPT. I know for a fact that many younger readers of this magazine were barely able to read when this book came out.

So, here we are in the future, 18 years later after the publication of this book, and yes, our world has visibly deteriorated (not only environmentally, but politically, socially, and economically), and this situation is, to a large degree, driven by software.

The author, David Rice, worked in the US government (for what it is worth), taught at James Madison University, and served as Executive Director of The Monterey Group, a consulting firm of which I could not find any current references. He was hired by Apple in 2011[86] to lead their security efforts, where he is still employed at the time of this writing according to his LinkedIn profile[87].

"Geekonomics" exposes and develops five major ideas.

First, that software is a public hazard; this includes, but is not limited to, web applications running with PHP 5.0[88] that never got any serious security review since 2004, like for example this little-known thing that nobody uses called Facebook.

Second, that security is an afterthought in the software market, and let us be honest; we have all witnessed some manager dropping security reviews or activities because of budget constraints.

Third, that end-user license agreements or EULAs are a cancer. As Microsoft states in

---

[82]https://www.ibm.com/products/turbonomic

[83]https://onpasture.com/2013/12/02/small-farm-egg-onomics/

[84]https://nft-onomics.com/

[85]https://www.economist.com/finance-and-economics/2024/07/11/trumponomics-would-not-be-as-bad-as-most-expect

[86]https://appleinsider.com/articles/11/01/24/apple_hires_former_nsa_navy_analyst_as_security_czar

[87]https://www.linkedin.com/in/david-rice-7b3686/

[88]https://museum.php.net/php4/

the OEM Windows 11 EULA[89],

> **Disclaimer.** Neither Microsoft, nor the device manufacturer or installer, gives any other express warranties, guarantees, or conditions. Microsoft and the device manufacturer and installer exclude all implied warranties and conditions, including those of merchantability, fitness for a particular purpose, and non-infringement.

Fourth, that cyberattacks and downtime are costing us lives, freedom, work, and sanity (the author prefers to use the more business-palatable expression "billions of dollars" to explain catastrophe after catastrophe, but the core idea is the same.)

Fifth, the author reminds us that there are other sectors that *do* have regulations, like, you know, the automotive or health industries, as explained in the section titled "A Matter of Trust" of chapter 4:

> Trust matters when it comes to systems; physical, digital, or otherwise. And mistrust in infrastructure has significant consequences. Trust derives from consistent stable performance, which in turn is derived from standards of design, construction, and skill.(...)
> There also happens to be considerable oversight, standards, and regulations placed on each of these elements. Car accidents certainly happen and will continue to happen regardless of standards and regulations, but the safety odds are *with* the drivers. On the Internet, the odds are decidedly against software users.

Finally, the author proposes some concrete solutions to have a more secure software ecosystem, like introducing legal liability for software defects (using the framework of Tort Law[90] as a basis) and actually implementing economic incentives for companies to *have* to start giving a shit about security.

Needless to say, it seems like this book was never published. Like it does not exist. People read this book, nodded in approval and shook their heads in dismay, left five stars on Amazon, and went back to play with their nerf guns across cubicles.

I have often denounced in the pages of this magazine the shameful and cowardly complicit action of many of my peers in this state of things. I do not see hordes of software developers dropping their cozy jobs in questionable organizations involved in privacy

[89] https://www.microsoft.com/content/dam/microsoft/usetm/documents/windows/11/oem-(pre-installed)/UseTerms_OEM_Windows_11_English.pdf

[90] https://en.wikipedia.org/wiki/Tort

violations, massacres, corruption, or software defects; I do not see them joining worker unions, supporting their peers who have been laid off; I do not see them denouncing in public the atrocities committed thanks to the very software they have been tasked to write.

Approving this collective pact of mediocrity and ignominy, programmers who have had their ethic neuron surgically removed through PlayStation abuse, scream the famous mantra “let us not talk politics here” all over the world, jumping yet again into a heated debate on Reddit about “tabs versus spaces” or “Rust versus Go”.

As a member of the software industry, each one of us has an ethical duty to release privacy-sensitive, secure, and maintainable code. In that order. “Geekonomics” will certainly give you enough arguments to convince your manager and to get that security budget approved. And if they do not, I hope you will know what to do next.

In other words, TL;DR: Please give a shit.

Cover photo by the author.