SOFTWARE IS A LONG CON

I had a conversation with a bridge engineer one evening not long ago. I said, “Bridges, they are nice, and vital, but they fall down a lot.”

He looked at me with a well-worn frustration and replied, “Falling down is what bridges do. It’s the fate of all bridges to fall down, if you don’t understand that, you don’t understand bridges.”

“Ok, I do understand that,” I replied. “But they fall down a lot. Maybe if we stepped back and looked at how we’re building bridges —”

“You can’t build a bridge that doesn’t fall down. That’s just not how bridges work”

I took a deep breath. “What if you could build a bridge that didn’t fall down as often?”

“Not practical — it’s too hard, and besides, people want bridges.” By now, he was starting to look bored with the conversation.

“I bet if you slowed down how you build bridges, you could make ones that lasted decades, even in some cases, centuries. You might have to be thoughtful, set more realistic expectations, do a lot more of the design of a bridge before you start building it, but..”

He interrupted me again. “Look, you’re not a bridge engineer, so you don’t really understand how bridges work, but people want bridges now. So no one is going to build a bridge like that,

even if it were possible, and I’m not saying it is.”

“But people get hurt, sometimes die, on these bridges.”

“Bridges fall down. Sometimes people are on them when they do. That’s not my fault as a bridge engineer, that’s literally how gravity works,” he said.

“I know there will always be accidents and problems with bridges, but I really do think that you could build them with careful planning, and maybe shared standards and even regulations in such a way that bridge collapses could be rare. Some of the problems with bridges are faults we’ve known about for decades, but they still get built into bridges all the time.”

He took a deep breath, and pinned me with a stare. “Even if we could, and it’s still entirely possible that no one can build these mythical bridges you’re talking about, that would slow down the building of bridges. People need bridges to get places. No one could afford to build bridges that slowly, and people would complain.” He stretched out the –plaaaain in complain, in a way that made clear this was the end of the argument and he’d won.

“They might not complain if they didn’t fall off bridges so often,” I mumbled.

He heard me. “Unlike you, people know that bridges fall down.”

Just then, a friend of mine, also a writer, also interested in bridges, stopped by.

“Hey guys!” he said. “So it looks like there’s a crew of Russian bridge destroyers with hammers and lighters who are running around in the middle of the night setting fires to bridges and knocking off braces with hammers. They started in Ukraine but they’re spreading around the world now, and we don’t know if our bridges are safe. They’ve studied bridges carefully and they seem to be good at finding where they’re most

flammable and which braces to knock off with their hammer."

We both regarded my friend a long moment, letting it sink in. I turned back to the bridge engineer and said, "Maybe we need to make them out of non-flammable material and rivet them instead of using exposed braces and clamps."

But he was already red in the face, eyes wide with anger and fear. "GET THE RUSSIANS!" he screamed.

OK, obviously it's not bridges I'm talking about, it's software. And that other writer is Wired's Andy Greenberg, who wrote a piece not that long ago on Russian hacking.

Greenberg's detailed and riveting story focuses largely on the politics of hacking, and the conflict between an increasingly imperialist Russia, and Ukraine, with an eye towards what it means for America. For people who respond to such attacks, like FireEye and Crowdstrike, these kinds of events are bread and butter. They have every reason to emphasize the danger Russia (or a few years ago, China) pose to the USA. It's intense, cinematic stuff.

It's also one of a long sequence of stories in this vein. These stories, some of which I've written over the years, show that our computers and our networks are the battlegrounds for the next great set of political maneuvers between nation-states. We the people, Americans, Russians, whomever, are just helpless victims for the coming hacker wars. We have Cyber Commands and Cyber attack and Cyber defense units, all mysterious, all made romantic and arcane by their fictional counterparts in popular media.

But there's another way to look at it. Computer systems are poorly built, badly maintained, and often locked in a maze of vendor contracts and outdated spaghetti code that amounts to a death spiral. This is true of nothing else we buy.

Our food cannot routinely poison us. Our

electronics cannot blow up, and burn down our houses. If they did, we could sue the pants off whomever sold us the flawed product. But not in the case of our software.

The Software Is Provided "As Is", Without Warranty of Any Kind

This line is one of the most common in software licenses. In developed nations, it is a uniquely low standard. I cannot think of anything infrastructural that is held to such a low standard. Your restaurants are inspected. Your consumer purchases enveloped in regulations and liability law. Your doctors and lawyers must be accredited. Your car cannot stop working while it is going down the freeway and kill you without consequences, except maybe if it's caused by a software bug.

It is to the benefit of software companies and programmers to claim that software as we know it is the state of nature. They can do stupid things, things we know will result in software vulnerabilities, and they suffer no consequences because people don't know that software could be well-written. Often this ignorance includes developers themselves. We've also been conditioned to believe that software rots as fast as fruit. That if we waited for something, and paid more, it would still stop working in six months and we'd have to buy something new. The cruel irony of this is that despite being pushed to run out and buy the latest piece of software and the latest hardware to run it, our infrastructure is often running on horribly configured systems with crap code that can't or won't ever be updated or made secure.

People don't understand their computers. And this lets people who do understand computers mislead the public about how they work, often without even realizing they are doing it.

Almost every initial attack comes through a phishing email. Not initial attack on infrastructure — initial attacks on everything — begins with someone clicking on an attachment or

a link they shouldn't. This means most attacks rely on a shocking level of digital illiteracy and bad IT policy, allowing malware to get to end-user computers, and failing to train people to recognize when they are executing a program.

From there, attackers move laterally through systems that aren't maintained, or written in code so poor it should be a crime, or more often, both. The code itself isn't covered by criminal law, or consumer law, but contract law. The EULAs, or End User Licensing Agreements (aka the contracts you agree to in order to use software), which are clicked through by infrastructure employees are as bad, or worse, as the ones we robotically click through everyday.

There are two reasons why I gave up reporting on hacking attacks and data breach. One is that Obama's Department of Justice had moved their policies towards making that kind of coverage illegal, as I've written about here. But the other, more compelling reason, was that they have gotten very very boring. It's always the same story, no one is using sophistication, why would you bother? It's dumb to burn a zero day when you can send a phishing mail. It's dumb to look for an advanced zero day when you can just look for memory addressing problems in C, improperly sanitized database inputs, and the other programatic problems we solved 20 years ago or more.

Programmers make the same mistakes over and over again for decades, because software companies suffer no consequences when they do. Like pollution and habitat destruction, security is an externality. And really, it's not just security, it's whether the damn things work at all. Most bugs don't drain our bank accounts, or ransom our electrical grids. They just make our lives suck a little bit more, and our infrastructure fail a little more often, even without any hackers in sight.

When that happens with a dam, or a streetlight, or a new oven, we demand that the people who

provided those things fix the flaws. If one of those things blows up and hurt someone, the makers of those things are liable for the harm they have caused. Not so if any of these things happen because of software. You click through our EULA, and we are held harmless no matter how much harm we cause.

When I became a reporter, I decided I never wanted my career to become telling the same story over and over again. And this is, once again, always the same story. It's a story of software behaving badly, some people exploiting that software to harm other people, and most people not knowing they could have it better. I'm glad people like Andy Greenberg and others at my old Wired home, the good folks at Motherboard and Ars Technica, and others, are telling these stories. It's important that we know how often the bridges burned down.

But make no mistake, as long as we blame the people burning the bridges and not the people building them, they will keep burning down.

And shit software will still remain more profitable than software that would make our lives easier, better, faster, and safer. And yeah, we would probably have to wait a few more months to get it. It might even need a better business model than collecting and selling your personal information to advertisers and whomever else comes calling.

I could keep writing about this, there's a career's worth of pieces to write about how bad software is, and how insecure it makes us, and I have written many of those pieces. But like writing about hackers compromising terrible systems, I don't want to write the same thing telling you that software is the problem, not the Chinese or the Russians or the boogeyman de jour.

You, the person reading this, whether you work in the media or tech or unloading container ships or selling falafels, need to learn how computers work, and start demanding they work

better for you. Not everything, not how to write code, but the basics of digital and internet literacy.

Stop asking what the Russians could do to our voting machines, and start asking why our voting machines are so terrible, and often no one can legally review their code.

Stop asking who is behind viruses and ransomware, and ask why corporations and large organizations don't patch their software.

Don't ask who took the site down, ask why the site was ever up with a laundry list of known vulnerabilities.

Start asking lawmakers why you have to give up otherwise inalienable consumer rights the second you touch a Turing machine.

Don't ask who stole troves of personal data or what they can do with it, ask why it was kept in the first place. This all goes double for the journalists who write about these things — you're not helping people with your digital credulity, you're just helping intel services and consultants and global defense budgets and Hollywood producers make the world worse.

And for the love of the gods, stop it with emailing attachments and links. Just stop. Do not send them, do not click on them. Use Whatsapp, use Dropbox, use a cloud account or hand someone a USB if you must, but stop using email to execute programs on your computer.

Thanks to my Patrons on Patreon, who make this and my general living possible. You can support this more of work at Patreon.

Image CC Skez