Operator's Log Jan 2, 2024

CrowdStrike Exposes The True Cause Of The Worldwide IT Meltdown

Blind happens when mechanical and practical degradation become self-deceiving[1]...

Attila Vágó Follow
Software Engineer, Operator's Log Follow 11 min read · 9 days ago

Listen 1.1K Share

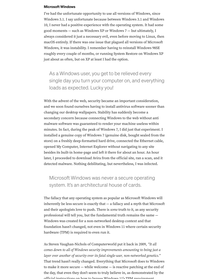


Photo by Jessica Dolan on Unsplash

The key moment my parents bought me a computer, I wanted a second one. As a teenager, I couldn't immediately afford it, but a couple of years later, I could finally buy a second one. I ran Linux on both my two machines and all was well with the world. This is Linux's forte: all throughout my life. Ever since, I always had two machines, and at times even three. I am able to run on two internet connections, one fiber and another one 5G. That's redundancy. Sure, the second machine is always a bit slower, but it's still a machine, and it can do the same things as the first one. The CrowdStrike update exposed both of these features and software engineering bad sides lacking across the world, and it poses a faulty app failover option into a very different light.

Microsoft Windows

The bad side: the authoritative opportunity to use all versions of Windows, since Windows 3.1, I say unfortunate because between Windows 3.1 and Windows 10, I never had a positive experience with the operating system. It had some good moments — such as Windows XP or Windows 7 — but ultimately, I always preferred Linux. Linux was better, more stable, faster, and it was much useful. Anyway, if there was one thing I appreciated about Windows, it was installability. I remember having to reinstall Windows Me roughly every couple of months, or running Sonic Restorer on Windows XP just about so often, but on XP at least I had the option.

> As a Windows user, you get to be relieved every single day you turn your computer on, and everything loads as expected. Lucky you!

With the advent of the web, security became an important consideration, and Microsoft Windows was not the most secure operating system even then changing our desktop wallpapers. Stability has suddenly become a secondary concern because connecting Windows to the web without anti-malware software was guaranteed to render your machine useless within minutes. So fact, during the peak of Windows 7, I did just that experiment. I connected a machine to the web without any anti-malware software, and it ceased to a fatally dying downward hard drive, and even the Microsoft cable, operated by Comcast, Internet Explorer without navigating to any site borders its built-in home page and left it there for about an hour. An hour later, I proceeded to download Avira from the official site, ran a scan, and it detected malware. Nothing debilitating, but unfortunate.

> Microsoft Windows was never a secure operating system. It's an architectural house of cards.

The fallacy that any operating system as popular as Microsoft Windows will be secure because it's popular. Hackers like to hack popular operating systems and their apotropaic ions to pass. There is zero truth to it, as any security professional will tell you, but the fundamental truth remains the same — Windows was created for a non-networked desktop environment and that foundation hasn't changed, not even in Windows 11 where certain security hardware (TPM) is required to even run it.

As Steven Vaughan-Nichols of ZDNet recently put it back in 2009, "It all comes down to a of Windows security improvements amounting to being just a layer over another of security one. In fact single sun, non-networked, protected, Windows XP is more secure than Windows 7 out of the box. Why? Because Windows 7 wasn't made — in Microsoft's — to be more secure out of the box, but at the day, that even they don't seem to truly believe in, as demonstrated by the official instructions on how to bypass Windows 11's TPM requirement.

> From a software engineer's perspective, Microsoft Windows looks like a heap of technical debt, ripe for a complete rewrite.

CrowdStrike approach to security in light of Windows being this poorly architected isn't surprising. As a tech company aiming for a leading role in business and enterprise security, opting for giving a step beyond the traditional antivirus software approach — that of just loading an app once and forgetting about it, letting it silently do its job and only ever bothering users from up, makes sense. It's hard to do, and it's not even Linux we're talking about. Windows desperately needs this much support to become a secure environment. Identify, AV, Linux all have security baked into the foundations of the operating system. We're talking 400 Unix, at the end of the day, developed at least with a very different design philosophy.

Microsoft Windows' fundamental need for aggressive 3rd party security solutions is one of the core causes of the global IT meltdown. Couple that with additional architectural weaknesses like a corrupt file being able to bring down an entire operating system, and you have yourself a perfect storm.

Poor design, lack of redundancy

As I am typing this on my M3 Max MacBook Pro, there is a fully charged M3 Max Mac Studio next to me, connected to the same monitor and keyboard, same version of macOS. Do one, I have automatic updates enabled, on the other, I shell because, believe it or not, Apple can cause plenty of havoc too. Having different versions of macOS running on the two devices enables me just continually to my work, but potential recovery solutions for the other device.

My redundancy philosophy as a manager was considered extreme and unnecessary by my friends, only to find out just a few years later as I was doing my Cisco Certification that redundancy is core to every reliable network. In fact, both reliability and redundancy are built into the TCP and UDP protocols — another great software development example of using the right tool for the job. Yes, TCP is slow, but it's reliable, and UDP is not as degradation-ridden that the prime objective, and UDP makes up things like your Zoom conversations don't fall apart when packets are lost.

> While TCP/IP is fundamentally unreliable, it also incorporates the mechanisms to avoid its core shortcoming becoming a self-debilitating flaw.

In software engineering, once we know and understand a system's potential downsides, we are trained to develop fail-safes. Handling an error, or an unknown case, should never or lose be accord nature to anyone writing code. Let me provide a small piece of Bash-like example that even the average Joe and Joe should understand.

```
if [dmg is mount] >> /dev 2
if [dmg is mount] >> /dev/null
if [dmg is mount] >> /dev/null
if [dmg is mount] >> /dev/null, /volumes, /volumes
if [dmg is mount] >> /dev 2
[dmg is mount]
[dmg is mount]
```

The above code would run fine for 5 years and 11 months, except for the February of the 4th year when the month is 29 days long. *That's an unfavorable case and the program would go "I don't know what to do now, I'm going to stop and throw an error"* — think your Windows blue screen of death for instance.

One solution would be to add and extra case for leap years, or the even safer choice would be to handle all other numbers with an else case like so:

```
if [dmg is mount] >> /dev/null
if [dmg is mount] >> /dev/null, /volumes, /volumes
if [dmg is mount] >> /dev/null, /volumes, /volumes
if [dmg is mount] >> /dev/null, /volumes, /volumes
else [dmg is mount]
```

Because what if suddenly someone decides that all this leap year stuff is too much effort? Let's just stick to a 28 days February all the time, and let the world deal with the repercussions of having Christmas and summer in 700 years. An ridiculous as that may sound, my program would work regardless, and I can assure accountants by and well never see a third version of dumb ever in my life.

As the CrowdStrike bug that brought systems down all over the world, I see two fundamental software engineering mistakes:

- Clearly, the core of a corrupt file in a driver wasn't covered in Microsoft Windows. It boggles the mind that such a simple mistake wasn't handled properly, it literally doesn't, and it takes no wonder — and unfortunately the same goes for every other multicloss actor out there — that a single corrupt file can bring down an entire operating system. Yes, a single corrupt file should not bring down an entire operating system, but in Windows' case, it does. Extremely optimistic error handling is always a sign of bad software development practices and poor design.
- No recovery mechanism was present in the operating system. My expectation of a computer in 2024 is that if an update goes wrong, it knows how to revert itself to the latest working version. Reverting is something we do a lot in software development. Even when you do things right, you sometimes find that your migration is "undo". Being an operating system, I would expect that if a driver update fails, the system of update and revert fall back to a previous working state. For instance, on a Mac, even if you delete the entirety of the hard-drive's contents, the machine will realize it has no OS, ask you to connect to the web, and download you the latest working version. In this case, all Windows needed to do was to revert the corrupt driver update, and it would have fixed it. Alas, because the BIOS — and had the version it has remained working, or at the very least load itself into Safe Mode, and advise the user to choose a previous version manually.

A fatally CrowdStrike update is unfortunate, and mostly avoidable, but as every software engineer knows, it is impossible to avoid a breaking bug into production once in a while. I have yet to meet a software developer who hasn't caused a major incident. It's like surgeons having amputees on the operating table. It's inevitable. I once locked millions of students out of a core learning platform for the powered length on the frontend and the backend didn't match.

> Anything that can go wrong, will go wrong. — Murphy's Law

How many more times do we need to hear this in software development to take it seriously? On this occasion, Microsoft clearly didn't consider it being a possibility, and it begs the question — why? As an operating system owner without any real vertical integration, it very well knows, and has known for decades, that it runs the OS in contexts it has virtually zero control over. Microsoft knows that it runs on a variety of hardware, and it knows that unlike together anything they can on top of it, it's the 800 lb gorilla of the room. One of my first jobs was to build SPAs for customers at the local computer store. The hardware was awful, the software that customers wanted installed was even worse and, for the most part, it scores of Microsoft, dalmatine programming is will an often concept.

But of course, we can't just blame Microsoft for all of it. Impacted businesses are just as much at fault here. Mission-critical systems running without redundancy or geographic degradation in mind. And it's not the first time, often. The IT blue brigades were constantly trying to convince the upper-ones to run a piece of software, that will directly what it was told, but instead of software and fallo redundancy, it did the opposite of what pilots expected.

> How can in 2024 an entire banking system, airports, national healthcare and transportation systems rely on a single operating system? Where's the redundancy???

IT systems of this magnitude should not be allowed to exist without built-in redundancies. It's not enough to just have generators for electricity lines, that helps, but as we live in an increasingly digital world, that's far from enough. Walking into a large hospital, saying 800Hz everything, and hoping for the best is not gain. In reality, the answer is some common what-ifs should be into.

- *What if the power goes out?* Re-switch to battery power while generators kick in to replenish our power storage. ✅
- *What if the internet goes down?* Re-switch to other fiber providers, followed by 5G, followed by satellites. If some switch to satellite communication. ✅
- *What if the OS crashes across the network?* Re-reboot using an alternative OS. ✅

There are the bare minimum answers I would expect. Clearly and sadly, we live in a world of IT infrastructures where none of this is a reality, my expectations seem to be out of sci-fi stories and the reasons are always the same — it costs too much — which brings me to my next point.

A culture of cheap

And this goes far beyond Gary Decionoro's famous "modrification" theory. Software is degrading without any great-tech degradation in place. Bad software works or doesn't. Microsoft and CrowdStrike aren't the only examples here of bad software. Boeing, for instance, is in the news quite a lot recently about Alaska Airlines incidents as well and in fact, my software company that recently added "AI features" via OpenAI's API because I can all but guarantee that the moment their API runs into issues, the monster-first services get filtered, crash or whatever else might happen, half the software out there will handle these outages very poorly.

Software is increasingly unreliable because it's lean and lean a mean about sanity and much more an interconnected network of services. In essence, there isn't anything wrong with software as a service, and integration, but the software industry is moving towards it because it's cheaper, and it's cheaper by a full quart. The alternative is either writing everything, or, what is likely unreliable — or implementing redundancies and experiences that even degraded, other outage value. Neither of these solutions is cheap, though.

I remember when Hyperoxia Web Apps (PWA) became the hot new kid on the block, and I remember how companies were falling over themselves to implement it was the hammered of hysterials shouting "Synergy!" and "Innovation!" and the candidate that stood out at interviews was the one that didn't bring up PWAs between saying "hello" and asking "how are you?" And yet, we find ourselves that same hammered for the hour, even Apple allowed it the middle-first to ship with their app website out there that doesn't work without a cellular node.

> We're dealing with a debilitating culture of cheapness across the software development industry.

Apple was another great hammered, but we made sure to back Apple into into something it was never meant to be interpreted as — a network-first delivery. In fact, the whole concept of Apple was meant to enable printing fast, not building fast. It is notoriously cheaper to run an IT infrastructure without redundancy in mind all if you're thinking of it in the IT budget, but overall, how much cheaper is it really?

CrowdStrike's unfortunate mistake will have long running repercussions. The potential for law suits by the hundreds, lost insurance companies, jobs could be lost, vacations ruined, and the list goes on. The ripple effects of this mistake will be felt for years to come, and unfortunately, I don't see I and help but think, a lot more organizations would have gone a long way towards preventing this disaster.

A precautionary future

Write this, Windll is angrily pointing the finger at CrowdStrike and its CEO, George Kurtz, I feel it mostly should look elsewhere. Continuous but not cheap software, patches upon patches, downright insane business practices relying on crossed fingers rather than well-tested and well-architected solutions coupled with increasingly poorly tested software updates across the tech industry is what got us here. Not a corrupt update.

Under these circumstances, I must ask, are we sure we still want it in our software and our daily lives? Are we sure we want to add more complexity to our software and IT infrastructure in a thread of terrors and cries? We clearly can't even do simple things well and reliably enough. Perhaps we should get back to the basics and ask more good software with simplicity, redundancy and graceful degradation in mind. How about that? Can we do that? Please?

> In the grand scheme of things, the CrowdStrike incident is merely a storm that exposed our interconnected world's foundations built on rattling hot sand. It was never a question of "if", but a "when", and that "when" was now.

Attila Vágó — Software Engineer imparting the world one line of code at a time. Cool word ones forever, writer of jokes, blogs and books. Author. Will accessibility advocate. LGBTQIA+, single-mind atheist. Loves and hates Apple? Read my Hello story here! Subscribe for more stories about software tech, coding and computer science by my regular readers. Follow write about modern tech and writing.

Technology Microsoft Cybersecurity Networking Software Development
Listen 1.1K Share

Written by Attila Vágó Follow 3K Following
Software Engineer, Operator's Log
Staff Software Engineer, Cloud Infrastructure, Operations and Automation Lead, CISSP and Agile Fan. Accessibility advocate, LGBTQIA+ ally, Linux fanboy, Atheist. ✅

More from Attila Vágó and Level Up Coding

Attila Vágó in Level Up Coding
KISS, The Only Software Design Principle Anyone Needs
Principle Of Least Power, Principle Of Least Surprise, Principle Of Least Astonishment, Principle Of Least Complexity
4 hours ago · 4.1K · 1 · 1

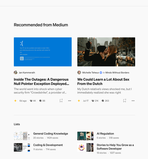
Level Up Coding
System Design Interview Design Patterns
Microservices, Load Balancer, Caching, Rate Limiter, Consistent Hashing, Leader Election, Gossip Protocol
Jul 2 · 1.1K · 1

Attila Vágó in Level Up Coding
Deploy Kubernetes to Expert Tutorial
Deploy Kubernetes to Digital Ocean using Terraform and Ansible
Jul 8 · 1.1K · 1

Attila Vágó in Level Up Coding
Stop Using Microservices Unless You Read This Article
Lessons from Amazon, Twitter, Netflix, and Shopify
Jun 15 · 5.4K · 1

See all from Attila Vágó
See all from Level Up Coding

Recommended from Medium



Joe Schwartzer
Inside The Outages: A Dangerous Red Pointer Exception Displayed.
The world went into a tailspin when a pointer error in a Microsoft Windows driver...
Jul 19 · 1.1K · 1

Dmytro Nosov in Coding Abstract
We Could Learn a Lot About Sex From the Dutch
My Dutch relatives always amazed me. Such ordinary people, living such extraordinary lives.
Jul 2 · 1.1K · 1

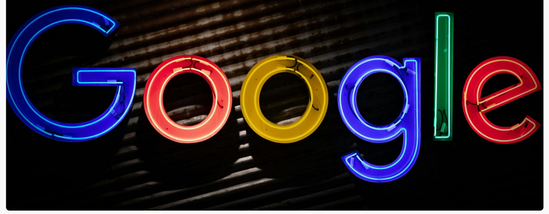
Coding & Development
Coding & Development
Jul 2 · 1.1K · 1

JS Magazine
JS Magazine · 1 day ago
Stop Using Yarn. You Live in a Dictatorship Without Elections — Bun Is Your Escape

Steven Thomas
My Statement on President Biden's Withdrawal
A Medium Exclusive Interview with Former Senior Advisor to President Joe Biden, Steven Thomas

Tom Cassidy in Coding Abstract
Mystery of a Double Photo Frame: Why Two Photos Are Hidden Behind One Frame
Jul 2 · 1.1K · 1

See all from Medium

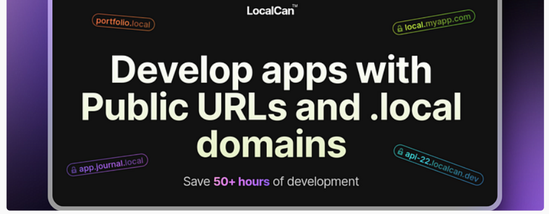


Josef Cruz in Stackademic

11 Shocking Secrets About Working at Google

That they don't want you to know

Jul 4 441 13



Daniel Craciun in Level Up Coding

6 Underrated Developer Tools That Will Blow Your Mind

Not a Member? Read for FREE here.

5d ago 302 1

See more recommendations