
# Platform Key Disclosure

**Description:** The obligatory follow-up on the massive CrowdStrike event: How do CrowdStrike's users feel? Are they switching or staying? How does CrowdStrike explain what happened, and does it make sense? How much blame should they receive? An update on how Entrust will be attempting to keep its customers from changing certificate authorities. Firefox appears not to be blocking third-party tracking cookies when it claims to be. How hiring remote workers can come back to bite you in the you-know-what. Did Google really want to kill off third-party cookies or are they actually happy? And is there any hope of ending abusive tracking? Auto-updating anything is fraught with danger. Why do we do it, and is there no better solution? And what serious mistake did a security firm discover that compromises the security of nearly 850 PC makes and models?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-985.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-985-lq.mp3

---

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. The post mortem on the CrowdStrike flaw. Actually, CrowdStrike explained how it happened. I think you'll enjoy Steve's reaction to that. Firefox is apparently not doing what it says it's doing when it comes to tracking cookies. We'll talk about that. And then a flaw that makes nearly 850 different PC makes and models insecure. It's all coming up next on Security Now!.

**Leo Laporte:** This is Security Now! with Steve Gibson, Episode 985, recorded Tuesday, July 30th, 2024: Platform Key Disclosure.

It's time for Security Now!, the show where we cover the latest news in the security front. And, boy, there's always a lot of news in the security front with this guy right here. He is our security weatherman, Mr. Steve Gibson. Hi, Steve.

**Steve Gibson:** And the outlook is cloudy with precipitation.

**Leo:** Chance of disaster, yes.

**Steve:** Yes. Remember duck and cover? Well, anyway. We did not, well, no, we did not have a new disaster since 10 days ago when, you know, we had one that pretty much made the record books. But we do have a really interesting discovery. And what's really even more worrisome is that it's a rediscovery. Today's podcast is titled "Platform Key Disclosure" for Security Now! #985, this glorious last podcast of July and the second-to-the-last podcast, the penultimate podcast where you are in the studio, Leo.

**Leo:** Yes, it is.

**Steve:** Rather than in your new attic bunker.

**Leo:** Can an attic be a bunker, though, really?

**Steve:** Oh, that's a good point, yeah.

**Leo:** I think I'm actually more exposed.

**Steve:** [Crosstalk] light tower, the lighthouse, yeah, you'll be the first to go.

**Leo:** Yes.

**Steve:** But really, sometimes you think maybe that's the best; right? End it now.

**Leo:** Oh, I always think that. I don't - the worst thing is a prolonged, slow, agonizing, suffering death.

**Steve:** What we know as life.

**Leo:** Or life, as it's known. That's the worst thing. Oh, we're being...

**Steve:** When you're a CIO, and you're - anyway. So we've got a bunch of stuff to talk about.

**Leo:** Yes.

**Steve:** We've got of course the obligatory follow-up on the massive CrowdStrike event. How do CrowdStrike users feel? Are they switching or staying? How does CrowdStrike explain now what happened? And does that make any sense? How much blame should they receive? We've also got an update on how Entrust will be attempting to retain its customers and keeping them from wandering off to other certificate authorities. Firefox just - no one understands what's going on, exactly, but it appears not to be blocking third-party tracking cookies when it claims to be. Also we're going to look at how hiring remote workers can come back to bite you in the you-know-what.

**Leo:** Oh, don't tell me that. That's all remote workers now. Ai ai ai ai ai.

**Steve:** Yeah, a security company got a rude awakening, and they learned something about just how determined North Korea is to get into our britches. Also, did Google really

want to kill off third-party cookies, or are they maybe actually happy about what happened? And is there any hope of ending abusive tracking? Auto-updating anything is obviously fraught with danger. We just survived some. Why do we do it, and is there no better solution?

And what serious mistake did a security firm discover that compromises the security of nearly 850 PC makes and models? This is another wakeup call. And I'll be further driving home the point of why, despite Microsoft's best intentions - assuming that we agree they have only the best of intentions, I know that opinions differ widely there - they can't, they can't keep Recall data safe. It's not necessarily their fault, it's just not a safe ecosystem that they're trying to do this in. So anyway, we have a fun Picture of the Week and a great podcast ahead.

**Leo:** Actually, that's a good topic. We could talk about could any computing system make Recall a safe thing? Probably not; right? It's just the nature of computing systems.

**Steve:** Yup. We have not come up with it yet.

**Leo:** Yeah. There's no such thing as a secure, perfectly secure operating system. Do we have a - I didn't even look. Do we have a Picture of the Week this week?

**Steve:** We do indeed. This one was just sort of irresistible. I know what they meant. But it's just sort of fun what actually transpired.

**Leo:** I just saw it.

**Steve:** Yeah. Now, I think, and you'll probably recognize this, too, the signage, I think it's a Barnes & Noble. It's sadly been quite a while since I've walked into an actual bookstore and filled my lungs with that beautiful scent of paper pulp. Used to love it. I grew up in the San Mateo Public Library.

**Leo:** Yeah, me, too. I love that, yeah.

**Steve:** They just would sort of nod to me was I would walk in, "Hello, Steve. Enjoy your time in the stacks." Anyway, so what we have here is a sign over a rack of books. The sign reads: "Large Print Audio Books." And of course I gave that the caption: "That's right, because large print is what one looks for in an audio book." Now, many of our clever listeners who received this already via email a couple of hours ago, they actually responded with what my first caption was, which was variations of, "Do you think they actually meant loud audio books?"

**Leo:** Okay. That could be. Wow.

**Steve:** And somebody took it more seriously and said, well, you know, Steve, somebody who's visually impaired might need large print on the instructions for how to play the audio book. And that's, you know...

**Leo:** That makes sense, yeah.

**Steve:** That's a point. Actually, what we know is the case is that large print books are on the upper few shelves.

**Leo:** That's right, and the audio's down there.

**Steve:** And the audio books are down below. And they put them both on the same sign, creating this little bit of fun for us and our audience. So thank you for whomever sent that to me. It's much appreciated. And I do have many more goodies to come.

Okay. So I want to share the note today that I mentioned last week. This was somebody who was forced to DM me because - I don't remember why. But he wrote the whole thing, then I think he created a Twitter account in order to send it to me. Now, I'll take this moment to mention I just was forced to turn off DMs, incoming DMs from unknown senders, or Twitter people who I don't follow. And of course I famously follow no one. So, and the reason is, when I went there today, I had a hard time finding any actual new direct messages to me from among the crap. It was so many people wanting to date me, and I don't think any of them actually do. You know, and I am wearing a ring, and I'm proud of that. That, and foreign language DMs that I can't read. And I finally thought, well, what am I, you know, why? What? No. So I just turned it off.

So I'm sorry, but I will still post the show notes from @SGgrc on Twitter. I've had a lot of people who thanked me for, even though I now have email up and running - and this system's working beautifully. 7,500 of our listeners received the show notes and a summary of the podcast and a thumbnail of the Picture of the Week and the ability to get the full sizes and everything several hours ago. But I just - it no longer works as an - I can't do open DMs. And I don't know why because it's been great for so long, Leo. I mean, I haven't had any problem. But maybe, I mean, the only thing I can figure is that the spam level on Twitter is just going way up.

**Leo:** Yeah. You've just been lucky, really.

**Steve:** I think I've just been lucky. Maybe I've just been sort of off the radar because I don't do much on Twitter except send out the little weekly tweet. So anyway, this is what someone wrote, really good piece that I wanted to share.

He said: "Hi, Steve. I'm writing to you from New South Wales, Australia. I don't really use Twitter, but here I am sending you a DM. Without a doubt you'll be mentioning the CrowdStrike outage in your Security Now! this week. I thought to give you an industry perspective here." And just to explain, I didn't get this from him until, as I do every Tuesday, I went over to Twitter to collect DMs and found this. So it didn't make it into last week's podcast, though I wish it had, so I'm sharing it today.

He said: "I work in cybersecurity at a large enterprise organization with the CrowdStrike Falcon agent deployed across the environment. Think approximately 20,000 endpoints." He said: "Around 2:00 to 3:00 p.m. Sydney, the BSOD wave hit Australia. The impact cannot be overstated. All Windows systems in our network started BSOD'ing at once. This is beyond a horrible outage. CrowdStrike will need to be held accountable and show they can improve their software stability, rather than sprucing AI mumbo jumbo," he said, "if they want to remain a preferred provider.

"Okay," he says. "In defense of CrowdStrike, their Falcon EDR tool is nothing short of amazing. The monitoring features of Falcon are at the top. It monitors file creation, network processes, registry edits, DNS requests, process executions, scripts, https requests, logons, logoffs, failed logons and so much more. The agent also allows for containing infected hosts, blocking indicators of compromise, opening a command shell on the host, collecting forensic data, submitting files to a malware sandbox, automation of workflows, an API, Powershell/ Python/Go libraries. It integrates with threat intel feeds and more and more.

"Most importantly," he says, "CrowdStrike has excellent customer support. Their team is helpful, knowledgeable, and responsive to questions or feature requests. Despite this disastrous outage, we are a more secure company for using CrowdStrike. They have saved our butts numerous times. I'm sure other enterprises feel the same. Full disclosure: I do not work for CrowdStrike, but I do have one of their T-shirts."

He says: "Why am I telling you this? Because the second-in- line competitor to CrowdStrike Falcon is Microsoft Defender for Endpoint (MDE)." He says: "MDE is not even close to offering CrowdStrike Falcon's level of protection. Even worse, Microsoft's customer support is atrocious to the point of being absurd. I've interacted with Microsoft's security support numerous times. They were unable to answer even the most basic questions about how their own product operated, and often placed me in an endless loop of escalating my problem to another support staff, forcing me to re-explain my problem to the point where I gave up asking for their help. I even caught one of their support users using ChatGPT to respond to my emails. And this is with an enterprise-level support plan."

He says: "As the dust starts to settle after this event, I imagine Microsoft will begin a campaign of aggressively selling Defender for Endpoint. Falcon is often more expensive than MDE, since Microsoft provides significant discounts depending on the other Microsoft services a customer consumes. Sadly, I imagine many executive leadership teams will be dumping CrowdStrike after this outage and signing on with MDE. Relying on Microsoft for endpoint security will further inflate the single point of failure balloon that is Microsoft, leaving us all less secure in the long run." And then he signs off: "Finally, I'm a big fan of the show. I started listening around 2015. As a result of listening to your show, I switched to a career in cybersecurity. Thank you, Leo and Steve."

So I wanted to share that because that's some of the best feedback I've had from somebody who really has some perspective here. And it all rings 100% true to me. This is the correct way for a company like CrowdStrike to survive and to thrive. That is, by really offering value. They're offering dramatically more value and functionality than Microsoft, so the income they're earning is actually deserved.

One key bit of information we're missing is whether all of these Windows systems, servers, and networks that are constantly being overrun with viruses and ransomware and costing their enterprises tens of millions of dollars to restore and recover - and remember, you know, those are the things that we're normally talking about here every week - are they protected with CrowdStrike, or is CrowdStrike saving those systems that would otherwise fall? You know, if CrowdStrike is actually successfully blocking enterprise-wide catastrophe, hour by hour and day by day, then that significantly factors into the risk/reward calculation. Our CrowdStrike user wrote: "They have saved our butts numerous times." And he said: "And I'm sure other enterprises feel the same."

Well, that is a crucially important fact that is easily missed. It may well be that right now corporate CIOs are meeting with their other C-suite executives and boards and reminding them that while, yes, what happened 10 days ago was bad, but even so it's worth it because this same system had previously prevented, say, I don't know, for example, 14 separate known instances of internal and external network breach, any of which may

have resulted in all servers and workstations being irreversibly encrypted, public humiliation for the company, and demands for illegal ransom payments to Russia.

So if that hasn't happened because, as our listener wrote, "CrowdStrike saved our butts numerous times," then the very rational decision might well be to stick with this proven solution in the knowledge that CrowdStrike will have definitely learned a very important and valuable lesson and will be arranging to never allow anything like this to happen again.

Now, if it never happens again, then remaining with this superior solution is the obvious win. But if it ever should happen again, then in retrospect remaining with them will have been difficult to justify, and I, you know, you could imagine not being surprised if people were fired over their decision not to leave CrowdStrike after this first major incident. But even so, if CrowdStrike's customers are able to point to the very real benefits they have been receiving on an ongoing basis for years, from their use of this somewhat, okay, can be dangerous system, then even so it might be worth remaining with it.

Before we talk about CrowdStrike's response, I want to share another interesting and important piece of feedback from a listener and also something that I found on Reddit. So our listener Dan Moutal sent, he wrote: "I work at a company that uses CrowdStrike, and I thought you would appreciate some insight. Thankfully, we were only minimally affected as many of our Windows users were at a team-building event with their laptops powered down and not working, and our servers primarily run Linux. So only a handful of workstations were affected. However, the recovery was hampered by the common knowledge, which turned out to be false, that the BitLocker recovery key would be needed to boot into Safe Mode.

"When you try to boot into Safe Mode in Windows, you are asked for the BitLocker recovery key. Most knowledgeable articles, even from Microsoft, state that you need to enter the BitLocker key at this point." He says: "But this is not required. It's just not obvious and not well known how to bypass this."

He says: "Here's what we discovered over the weekend: Cycle through the blue-screen error, when the host continues to crash, until you get to the recovery screen. Perform the following steps. First, navigate to Troubleshoot > Advanced Options > Startup Settings. Press Restart. Skip the first BitLocker recovery key prompt by pressing Escape. Skip the second BitLocker recovery key prompt by selecting 'Skip This Drive' at the bottom right. Navigate to Troubleshoot > Advanced Options > Command Prompt. Then enter bcdedit /set and then {default) safe boot minimal, then press Enter. Close the command prompt window by clicking the X in the top right. This will return you back to the blue screen, which is the Windows RE main menu. Select Continue. Your PC will now reboot. It may cycle two to three times. But then your PC should boot into Safe Mode."

He said: "I confirmed this worked and allowed us to recover a few systems where we did not have the BitLocker keys available." He says: "I think Microsoft deserves a lot of blame for the poor recovery process when Safe Mode is needed. They should not be asking for BitLocker keys if they're not needed. At the bare minimum, they need to make this knowledge much more well known so system admins who don't have BitLocker keys handy can still boot into Safe Mode when disaster strikes.

"I also want to send a shout-out to CrowdStrike's technical support team. I'm sure they were swamped by support requests on Friday, but despite that we were talking to them after waiting on hold for only 10 minutes, and they were very helpful. Most vendors are not quick or useful on a good day, let alone on a day when they are the cause of a massive outage. CrowdStrike is an expensive product, but it is clear that a large chunk of that expense pays for a large and well-trained support staff." So that was from a listener of ours.

**Leo:** Those are really excellent points. I mean...

**Steve:** Aren't they? Yes.

**Leo:** Yeah, yeah. I mean, so much worse to get bit by ransomware than a temporary boot flaw.

**Steve:** Yes, exactly. So for our listeners to say, yes, you know, this was not good, that first listener had 20,000 endpoints. But he said, "Still, it has saved our butts." Well, "saved our butts" must mean that he has evidence that something bad would have happened to them, had CrowdStrike not blocked it. So what's that worth? You know? It's worth a lot.

Okay. So on Reddit I found a posting. Oh, and by the way, Leo, I just should mention, I don't normally have our monitor screen up so that I know if you're there or not. So just a note to your control room.

**Leo:** I'm here, by the way.

**Steve:** Nice to hear your voice. So this post on Reddit said: "Just exited a meeting with CrowdStrike. You can remediate all of your endpoints from the cloud." There's news. He said: "If you're thinking, 'That's impossible, how?'" he says, "this was also the first question I asked, and they gave a reasonable answer.

"To be effective, CrowdStrike services are loaded very early in the boot process" - which of course is what we talked about last week - "and they communicate directly with CrowdStrike. This communication is used to tell CrowdStrike to quarantine windows\system32\drivers\crowdstrike\ and then the infamous c-00000291*.sys file."

So he said: "To do this, you must first opt-in" - that is, for this cloud-based recovery - "by submitting a request via the support portal, providing your customer IDs and request to be included in cloud remediation. At the time of the meeting," he says, you know, when he was in this meeting - "the average wait time for inclusion was less than one hour. Once you receive email indicating that you have been included, you simply have your users reboot their computers."

**Leo:** Oh.

**Steve:** In other words, it's a self-repair of this problem.

**Leo:** That seems sensible, yeah.

**Steve:** Yeah. He said: "CrowdStrike noted that sometimes the boot process completes too quickly for the client to get the update, and a second or third try is needed; but it is working for nearly all of their affected users. At the time of the meeting, they had remediated more than 500,000 endpoints this way." He says: "It was suggested to use a wired connection when possible since WiFi-connected users have the most frequent

trouble with this approach, probably because WiFi connectivity becomes available later in the boot process, after a crash will have occurred." He says: "This also works with home and remote users since all they need is an Internet connection, any Internet connection. The point is, they do not need to be and should not be VPN'd into the corporate network."

So anyway, I thought that was interesting since essentially we have another of those race conditions. We've talked about those recently; right? In this case it's one where we're hoping that the CrowdStrike network-based update succeeds before the crash can occur.

Okay. So with all of that, what more do we know today than we did a week ago at the time of last week's podcast? The questions that were on everyone's mind were variations of "How could this possibly have been allowed to happen in the first place?" "How could CrowdStrike not have had measures in place to prevent this?" And even, you know, "Staggering the release of the buggy file would have limited the scale of the damage. Why wasn't that, at least, part of their standard operating procedure?"

For last week's podcast we had no answers to any of those questions. Among the several possibilities I suggested were that they did have some pre-release testing system in place; and, if so, then it must have somehow failed. And that's the explanation that the industry has received from them since that time. I have no doubt about its truth, since CrowdStrike executives will be repeating that under oath shortly.

Last week we shared what little was known, which CrowdStrike had published by that point under the title "What Happened?" But they weren't yet saying how. And we also had that statement that I shared from George Kurtz, CrowdStrike's founder and CEO. This week we do have their "What Happened?" which is followed by "What Went Wrong and Why?" And, okay. Despite the fact that it contains a bunch of eye-crossing jargon which sounds like gobbledygook, I think it's important for everyone to hear what CrowdStrike said. So here's what they have said to explain this. And in order to create the context that's necessary, we learn a lot more about the innards of what's going on.

They said: "CrowdStrike delivers security content configuration updates to our sensors." And when they say "sensors," they're talking about basically a kernel driver. So some of it is built into the kernel driver, or delivered with driver updates, and the other is real-time. So whenever you hear me say "sensors," you know, it's not anything physical, though it sounds like it is. It is software running in the kernel. They said: "In two ways: Sensor Content that is shipped with our sensor directly, and Rapid Response Content that is designed to respond to the changing threat landscape at operational speed." They wrote: "The issue on Friday involved a Rapid Response Content update with an undetected error.

"Sensor Content provides a wide range of capabilities to assist in adversary response. It is always part of a sensor release and not dynamically updated from the cloud. Sensor Content includes on-sensor AI and machine-learning models, and comprises code written expressly to deliver long-term reusable capabilities for CrowdStrike's threat detection engineers. These capabilities include Template Types," which this figures strongly in the response, "which have pre-defined fields for threat detection engineers to leverage in Rapid Response Content. Template Types are expressed in code. All Sensor Content, including Template Types, go through an extensive QA process, which includes automated testing, manual testing, validation, and rollout.

"The sensor release process begins with automated testing, both prior to and after merging into our code base. This includes unit testing, integration testing, performance testing, and stress testing. This culminates in a staged sensor rollout process that starts with dogfooding internally at CrowdStrike, followed by early adopters. It's then made

generally available to customers. Customers then have the option of selecting which parts of their fleet should install the latest sensor release ('N'), or one version older ('N-1') or two versions older ('N-2') through Sensor Update Policies."

Now, okay. To be clear, all of that refers to essentially the driver, the so-called "sensor." So for that stage they are doing incremental rollout, early adopter testing and so forth. Unfortunately, not for the rapid response stuff.

So they said: "The event of Friday, July 19th, was not triggered by Sensor Content, which is only delivered with the release of an updated Falcon sensor." Meaning, you know, updated static drivers. They said: "Customers have complete control over the deployment of the sensor, which includes Sensor Content and Template Types. Rapid Response Content is used to perform a variety of behavioral pattern-matching operations on the sensor using a highly optimized engine." Right, because you don't want to slow the whole Windows operating system down as it takes time out to analyze everything that's going on.

So they said: "Rapid Response Content is a representation of fields and values, with associated filtering. This Rapid Response Content is stored in a proprietary binary format that contains configuration data. It is not code or a kernel driver." But I'll just note, unfortunately it is interpreted. And how much time have we spent about interpreters going wrong on this podcast?

They said: "Rapid Response Content is delivered as 'Template Instances,' which are instantiations of a given Template Type. Each Template Instance maps to specific behaviors for the sensor to observe, detect, or prevent. Template Instances have a set of fields that can be configured to match the desired behavior. In other words, Template Types represent a sensor capability that enables new telemetry and detection, and their runtime behavior is configured dynamically by the Template Instance - in other words, specific Rapid Response Content.

"Rapid Response Content provides visibility and detections on the sensor without requiring sensor code changes. This capability is used by threat detection engineers to gather telemetry, identify indicators of adversarial behavior, and perform detections and preventions. Rapid Response Content is behavioral heuristics, separate and distinct from CrowdStrike's on-sensor AI prevention and detection capabilities. Rapid Response Content is delivered as content configuration updates to the Falcon sensor. There are three primary systems: the Content Configuration System, the Content Interpreter, and the Sensor Detection Engine.

"The Content Configuration System is part of the Falcon platform in the cloud, while the Content Interpreter and Sensor Detection Engine are components of the Falcon sensor." In other words, running in the kernel. So we've got a content interpreter running in the kernel. What could possibly go wrong? Well, we found out. They said: "The Content Configuration System is used to create Template Instances, which are validated and deployed to the sensor through a mechanism called Channel Files. The sensor stores and updates its content configuration data through Channel Files, which are written to disk on the host.

"The Content Interpreter on the sensor reads the Channel Files and interprets the Rapid Response Content, enabling the Sensor Detection Engine to observe, detect, or prevent malicious activity, depending on the customer's policy configuration. The Content Interpreter is designed to gracefully handle exceptions from potentially problematic content." Let me read that sentence again because that's what failed. "The content interpreter is designed to gracefully handle exceptions from potentially problematic content." Except in this instance, as we know, it did not.

And they finish with: "Newly released Template Types are stress tested across many aspects, such as resource utilization, system performance impact, and event volume. For each Template Type, a specific Template Instance is used to stress test the Template Type by matching against any possible value of the associated data fields to identify adverse system interactions." In other words, that's the nice way of saying "crashes." And "Template Instances are created and configured through the use of the Content Configuration System, which includes the Content Validator that performs validation checks on the content before it is published."

Okay, now, I've read this a total of I think maybe five times, and I now finally feel like I understand it. So, you know, don't be put off if that just all like, what did he just say? I get it.

Then they lay out a timeline of events which I'll make a bit easier to understand by interpreting what they wrote. So recall that there was mention of named pipes being involved with this trouble. And I explained that named pipes were a very common means for different independent processes to communicate with each other within Windows. So way back on February 28th of this year, CrowdStrike sensor v7.11 was made generally available to customers. It introduced a new Inter Process Communication (IPC) Template Type which was designed to detect novel attack techniques that abused Named Pipes. This release followed all Sensor Content testing procedures outlined above in that Sensor Content section. So again, sensor content is - this was an update essentially to the driver and all of its AI and heuristic stuff. And that happened on February 28th.

A week later, on March 5th, a stress test of the IPC, this newly created IPC Template Type was executed, they said, in our staging environment, which consists of a variety of operating system workloads. The IPC Template Type passed the stress test and was thereby validated for use. Later that same day, following the successful stress testing, the first of the IPC Template Instances were released to production as part of a content configuration update. So like what happened on this fateful Friday, that happened back on March 1st for the first time for this new IPC Template Type. They said, after that, three additional IPC Template Instances were deployed between April 8th and April 24th. These Template Instances performed as expected in production.

Then, on that fateful day of Friday, July 19th, two additional IPC Template Instances were deployed, much as multiples of those had in February, March, and April. One of these two, one of the two deployed on July 19th was malformed and should never have been released. But due to a bug in the Content Validator, and also in the interpreter, that malformed IPC Template Instance erroneously passed validation, despite containing problematic content data.

They said, based on the testing performed before the initial deployment of the new IPC Template Type back on March 5th, and in the trust in the checks performed by the Content Validator, and the several previous successful IPC Template Instance deployments, on Friday the 19th these two instances, one of them malformed, were released and deployed into production. When received by the sensor and loaded into the Content Interpreter, which is in the kernel, problematic content in Channel File 291 resulted in an out-of-bounds memory read triggering an exception. This unexpected exception could not be gracefully handled, resulting in a Windows operating system crash, the infamous Blue Screen of Death, which then followed its attempt to recover.

And Leo, we're going to talk about how they prevent it from happening again, but let's take a break so I can catch my breath and sip a little caffeine.

**Leo:** Bessie in our YouTube chat says: "How can all those PCs fail, but no PCs at CrowdStrike itself failed? Don't they use CrowdStrike at CrowdStrike?" I'm curious,

and I sure you'll cover this, how quickly they figured out that this update was causing a problem. Surely - maybe CrowdStrike was closed for the day. I don't know. It happened in Australia first; right?

**Steve:** Well, it did happen in the wee hours of the morning in the United States.

**Leo:** It's wild. It's just wild. Anyway, we will get to that. The post-mortem. Steve, I'm trying to decide, I'm bringing, you know, I'm packing up stuff, as you probably noticed, stuff's starting to disappear from the studio. I'm trying to decide, should I take this needlepoint that says "Look Humble"? Can't decide. I'll have to think about it.

**Steve:** Where did it come from? Does it have a special meaning for you?

**Leo:** It wasn't my Grandma, it was probably a listener sent it to me. I just always thought it was very funny. I've always had it. I am definitely taking the Nixie clock. You cannot have a studio without a Nixie clock.

**Steve:** I guess, no, I think you should keep the Nixie clock. It's definitely a...

**Leo:** Everything that blinks I'm taking.

**Steve:** Good.

**Leo:** I am not taking this darn clock, which has been the bane of my existence since the Brick House. People get mad when that clock is not visible on the show, that digital clock. But I have other clocks. I'm not going to bring that one. We'll see.

**Steve:** Well, I guess the question is also how much room do you have.

**Leo:** Well, that's the point is, you know, pretty much everything you see behind me I'm going to leave here. We've got a company that does...

**Steve:** Leave behind.

**Leo:** Yeah. There's a company that's a liquidator.

**Steve:** Ah.

**Leo:** They come in, and they sell what they can, they donate what they can, recycle what they can, and toss the rest. And I guess that's all we can do. This is a crazy amount of stuff we've accumulated over the years.

**Steve:** As one does.

**Leo:** As one does. We are opening the studio on the 8th to any Club TWiT members who want to come and get something. Come on down. We're blowing it out to the bare walls. You know I'm giving away, and I hate to do it, but I have that giant demonstration slide rule. I don't think you can see it. You can see the bottom of it. It's one of those yellow...

**Steve:** Oh, I do see the bottom of it, with the plastic slider.

**Leo:** But where am I going to put that? Hanging off the roof, I don't know.

**Steve:** Yeah.

**Leo:** I only have, you know, a tiny little attic studio. So a lot of stuff getting left behind, I'm sad to say.

**Steve:** Well, I've got some friends who - some of my high school buddies are actively lightening their load.

**Leo:** Yeah.

**Steve:** Like one guy deliberately converted his huge CD collection over to audio files and threw away all the CDs. I mean, it hurt to do it, right, because those are, like...

**Leo:** Lifetime. That's a lifetime of collecting.

**Steve:** Yeah. But it's like, you know, I want to travel more. I don't want to...

**Leo:** And, like, am I going to dump this on my kids? I don't want to do that. So, yeah, they call it - there's a book about it called "Swedish Death Cleaning," where you prepare for your death and do your heirs a favor. You get rid of the stuff that you don't think they'd be interested in. It's hard, though.

**Steve:** Yeah, I don't know what those PDP-8s that I've got are going to...

**Leo:** I'm taking my IMSI, you know, the two Raspberry Pi devices, and the PiDP-8. I'm absolutely taking those. They're too cool. Blinking lights have got to stay.

**Steve:** Well, and I have been remiss in not telling our listeners, it's just there's so much that's been happening on the podcast, but the guy that did the fantastic PiDP-8...

**Leo:** PiDP-8, yeah.

**Steve:** ...has done a PDP-10. And it is astonishing. I mean, it is. I'll have to make time to - I don't know when. The podcast has just been so crazy lately. But the emulated PDP-10, he gathered all the software from MIT...

**Leo:** It was Oscar Vermeulen; right? Was it...

**Steve:** Oscar, yes, Oscar has done a PDP-10, also with a little Raspberry Pi running behind it, a gorgeous injection-molded PDP-10 console recreation. So there's the 8.

**Leo:** Yes, this is the one we have.

**Steve:** And the 10. Is there a link to it there?

**Leo:** Let's see if he has - his vintage computer collection, I think this is...

**Steve:** Yeah, he gave it its own website.

**Leo:** Oh, okay. I'll find that.

**Steve:** I'm surprised he's not linked to it. But you might put in PDP-10 recreation or something.

**Leo:** Yeah, see what I can find.

**Steve:** To find it because, oh, my goodness.

**Leo:** Here we go. There we go. Obsolescence Guaranteed.

**Steve:** Look at that.

**Leo:** Oh. Okay. Oscar, I want it.

**Steve:** It is astonishing.

**Leo:** Oh, it's beautiful. It's very Star Trek. That is...

**Steve:** It is gorgeous. That's an injection molded, you know, full working system. All of the software is there. You're able to connect a normal PC to it so you can work with a console and keyboard.

**Leo:** Oh, it's got Adventure. Uh-oh. I might have to buy this. So it's running a Pi, Raspberry Pi, but that's the same performance as a PDP-10? I guess so.

**Steve:** Oh, it blows the PDP-10 away. He had to slow it down in order in order to make it...

**Leo:** Oh. This is so cool.

**Steve:** I mean, the original software running. And so there they were, comparing the operation of their console to a real one to the - in fact, I was thinking about this because Paul Allen was selling some of their original machines; right?

**Leo:** Wow. Oscar Vermeulen. Obsolescence.wixsite.com is Obsolescence Guaranteed. And you can build the kit. You can buy it or build the kit.

**Steve:** Yup, it is embarrassingly inexpensive, again.

**Leo:** Nice.

**Steve:** And, oh, boy. But, I mean, he and his buddy, they were out demonstrating it to the Boston Computer Museum. And he said, "Steve, could you make time for us to show you?" So he came and set this up, plugged its HDMI output into our screen in our family room and gave Lorrie and me a full demonstration of the operation of this.

**Leo:** Poor Lorrie. Did Lorrie know what she was getting into when she married you? And later we're going to have somebody demonstrate a replica PDP-10. Won't that be fun? I love it. Well, Oscar, well done. I'm glad he did it again. Yeah, so I got the email, too. And I guess I'll have to build - because I have the PiDP-8 is already on the set in the attic. And I can't upgrade.

**Steve:** Oh, this 10, oh, just look at that console. It is just gorgeous.

**Leo:** Oh, it's beautiful.

**Steve:** It was funny, too, because her 28-year-old son Robert happened to also be there. And he made - he was, like, watching this. And he'd never seen a console before, right, with like lights and switches. And he said, "Are those bits?"

**Leo:** Good question. Good question.

**Steve:** It was like, it would have never occurred to me to ask that question. But it's like, yes, those are bits. Those are what bits are, are those individual lights turning on and off, and the switches are bits.

**Leo:** Wow. Wow. Very, very cool.

**Steve:** And what's cool is that, whereas the PDP-8 is a pain to program because you've only got a three-bit op code, so you've got seven instructions, the PDP-10, oh, it's a 36-bit system, and it's a gorgeous instruction set. I mean, really just a joy. So, I mean, so this is a complete recreation. You can be using it with its editors and its compilers and the works.

**Leo:** Wow. Oh, that's what you want; right, Steve? There, by the way, are Oscar and Otto, showing off the entire line, Oscar's on the left, at the Vintage Computer Festival. Very cool. Very cool. So they've got a 1, 8, 11, and a 10.

**Steve:** Yes. Oh, and it says down there, and the 1 are at - I guess 1 and 10 at that time were in prototype. 10 is finished, and they're now working on a PDP-1. And I have to tell you, he credits this podcast as changing this from a hobby to a business because there was so much interest shown in the PiDP-8, and then in the 11, that they turned it into a business. And so anyway, I'm glad that this came up because I've been - I felt badly that I have not found time because, I mean, our podcasts have been running more than two hours recently.

**Leo:** I know, I know. But I'm so glad because I wanted to get this in, too. So I'm glad we could mention it, as I pack up my PiDP and bring it home, my PiDP-8.

**Steve:** And gentlemen listening, if you saw the actual size, it's not a huge thing. So it does store in the closet. You were talking about how patient Lorrie is.

**Leo:** Don't put it on the dining room table, whatever you do. I know you're tempted, gentlemen. But don't. Yeah, it's the white one right here. This is it.

**Steve:** Yup, exactly.

**Leo:** So it's what, it's about a couple of feet wide maybe.

**Steve:** It is a scale, yes, it's a scale size replica of the console of the original PDP-10.

**Leo:** Yeah. And I have that one.

**Steve:** But loaded with all of the original software. They even have one guy who specializes in recreating recovering data from unreadable nine-track magnetic tapes. And so they were getting mag tapes. You can see one right there behind my head. That is a

nine-track magnetic tape. That actually came from SAIL, from Stanford's Artificial Intelligence Lab.

**Leo:** Oh, that's cool.

**Steve:** And that's got my code on it.

**Leo:** Oh, that's really cool. Wow.

**Steve:** So they've literally - they went back and recreated the original files and in some cases hand-editing typos out in order to get everything to recompile again, in order to - because there was no, like, preservation project until now. So they've really - they did a beautiful job.

**Leo:** Mm-hmm. Mm-hmm.

**Steve:** And, you know, congratulations.

**Leo:** Yeah.

**Steve:** Okay. So following on what happened...

**Leo:** Oh, back to the bad news.

**Steve:** Naturally, yeah, naturally CrowdStrike wants to explain how this will never happen again.

**Leo:** Yeah. Let's hear it.

**Steve:** Yeah. So under the subhead of Software Resiliency and Testing, they've got bullet points. And I have to say that this first batch sounds like the result of a brainstorming session rather than an action plan. They have Improve Rapid Response Content testing. That was the problematic download; right? By using testing types such as local developer testing; content update and rollback testing; stress testing, fuzzing and fault injection; stability testing; and content interface testing. And of course many people would respond to that, why weren't you doing all that before? Unfortunately, that's the generic response, right, to anything that they say that they're now going to do is, well, why weren't you doing that before?

Anyway, so they also have: Add additional validation checks to the Content Validator for Rapid Response Content. A new check is in progress to guard against this type of problematic content from being deployed in the future. Good. Enhance existing error handling in the Content Interpreter, right, because it was ultimately the interpreter that crashed the entire system when it was interpreting some bad content. So that should not have been able to happen.

And then under the subhead of Rapid Response Content Deployment, they've got four items: Implement a staggered deployment strategy for Rapid Response Content in which updates are gradually deployed to larger portions of the sensor base, starting with a canary deployment; improve monitoring for both sensor and system performance, collecting feedback during Rapid Response Content deployment to guide a phased rollout; provide customers with greater control over the delivery of Rapid Response Content updates by allowing granular selection of when and where these updates are deployed; provide content update details via release notes which customers can subscribe to.

And I have to say, kind of reading between the lines again, you know, programmers have egos; right? We write code, and we think it's right. And then it's tested, and the testing agrees that it's right. And it's difficult without evidence of it being wrong to, like, to go overboard. They did have systems in place to catch this stuff. It turns out in retrospect something got past that system, or those systems. Now they know that what they had was not good enough, and they're making it better.

So, you know, I get it that could they have done more? Yes. But can't you always do more? Yes. And then one could argue in that case, if it's possible, if it's in any way possible for something to go wrong, then shouldn't you prevent that? Well, they thought they had. They thought that the content interpreter was bulletproof, that it was an interpreter. It would find any problems and refuse to interpret them if they were going to cause a problem. But there was a bug in that. So this happened.

Okay. So the bottom line to all of this, I think, is that CrowdStrike now promises to do what it should have been doing all along, like the staggered deployment. Again, I mean, that's indefensible; right? Why were you guys not incrementally releasing this? Well, it's because they really and truly believed that nothing could cause this to happen. They really thought that. They were wrong. But they, you know, it wasn't negligence. I mean, in the same sense that, you know, most programmers don't release buggy code; right? They fix the bugs. Microsoft is an exception. They've got a list of 10,000 known bugs when they release Windows. But they're small, and they figure they won't actually hurt anybody, and they're not showstoppers. They actually use that term. So it's like, okay, fine, it works.

So is this another of those small earthquake tremors I've been recently talking about? You know, I guess it would depend upon whom you ask. The source of the problem was centralized, but the remediation of the problem was widely distributed. Across 8.5 million machines, several hundred thousand individual technicians got to work figuring out what had happened to their own networks and workstations, and each was repairing the machines over which they had responsibility. Because initially, you know, CrowdStrike, as we saw, they ended up coming up with a cool cloud-based solution. But initially that didn't exist. Presumably it will now be deployed in some sort of a permanent fashion.

And as we know in the aftermath, some users of Windows appear to have been more seriously damaged than others. In some cases machines that were rebooted repaired themselves by retrieving the updated and repaired Template File. Whereas in other situations, such as, wow, Delta Airlines, the effects from having Windows system crashing lasted days.

I have no direct experience with CrowdStrike; but not a single one of our listeners from whom we have heard, even after enduring this pain, sounded like they would prefer to operate without CrowdStrike-level protection and monitoring in the future. And I think that's a rational position. No one was happy that this occurred, and it really is foreseeable that CrowdStrike has learned a valuable lesson about using belts, suspenders, Velcro, and probably some epoxy. They may have grown a bit too comfortable over time, but I'll bet that's been shaken out of them today. And I have no

doubt that it will be, I mean, like that they really raise the bar on having this happen to them again.

Another little bit of feedback. Since it's relevant to the CrowdStrike discussion, I wanted to share what another listener of ours, Vernon Young, shared with me. He wrote: "Dear Steve. I am the IT Director for a high school and manage 700 computers and 50 virtual servers."

**Leo:** Oh. Oh.

**Steve:** Get this, Leo. "A few weeks before the Kaspersky ban was announced, I placed a $12,000 renewal order with Kaspersky, $12,000 which will now be lost..."

**Leo:** Forever.

**Steve:** "...since the software won't work after September," he wrote. He said: "After the ban was announced, I started looking for alternatives. I decided on Thursday, July 18th to go with CrowdStrike."

**Leo:** Oh, my god, the day before. Oh, my god.

**Steve:** He said: "The day before the world collapsed." He said: "Thankfully, I didn't feel like walking to the copier to scan the purchase order to send to the sales rep before I left for the day. Needless to say, I changed my mind Friday morning."

**Leo:** Bullet dodged.

**Steve:** Now, I cannot imagine being Vernon, our listener, and needing to corral a high school campus full of mischievous and precocious high-schoolers...

**Leo:** People like you.

**Steve:** ...who imagine, you know, as high-schoolers will, that they're more clever than the rest of the world and whose juvenile brains' sense of right and wrong hasn't yet had the chance to fully develop. But think about the world Vernon is facing. He invests $12,000 to renew the school's Kaspersky AV system license, only to have that lost. Then decides that CrowdStrike looks like the best alternative, only to have it collapse the world. For what it's worth, I stand by the way I ended that CrowdStrike discussion. I would go with them today.

**Leo:** Yes. Yeah.

**Steve:** All of the feedback we've received suggests that they are topnotch, that they're very clearly raising the bar to prevent another mistake like this from ever slipping past. There's just no way that they haven't really learned a lesson from this debacle, and I

think that's all anyone can ask at this point. And the fact that our listeners are telling us they are the best there is...

**Leo:** That there's no other choices, no good choice, yeah.

**Steve:** Right. Microsoft is number two, and they don't hold a candle to...

**Leo:** Right.

**Steve:** I mean, you know, certainly all of the systems that we talked about succumbing to ransomware are at least running Microsoft Defender, and that's not helping them.

Okay. So who is to blame? Our wonderful hacker friend Marcus Hutchins posted a wonderfully comprehensive 18-minute YouTube video which thoroughly examines, explores, and explains the history of the still-raging three-way battle among Microsoft, third-party AV vendors, and malware creators. That video, it's on YouTube, it's this week's GRC shortcut of the week, so your browser can be redirected to it if you go to grc.sc/985, today's episode number, grc.sc as in shortcut, grc.sc/985. When you go there, be prepared for nearly 18 minutes of high-speed nonstop perfectly articulated techie detail because that's what you're going to get.

I'll summarize what Marcus said. Since the beginning of Windows, the appearance of viruses and other malware, and the emergence of a market for third-party antivirus vendors, there's been an uncomfortable relationship between Microsoft and third-party AV. To truly get the job done correctly, third-party antivirus has always needed deeper access into Windows than Microsoft has been willing or comfortable to give. And the CrowdStrike incident shows what can happen when a third party makes a mistake in the Windows kernel. But it is not true, Marcus says, that third-party antivirus vendors can do the same thing as Microsoft can without being in the kernel. This is why Microsoft themselves do not use the APIs they have made available to other antivirus vendors. Those APIs do not get the job done.

And the EU did not say that Microsoft had to make the kernel available to other third parties. The EU merely said that Microsoft needed to create a level playing field where the same features would be available to third parties as they were using themselves. Since Microsoft was unwilling to use only their own watered-down antivirus APIs and needed access to their own OS kernel, that same EU-mandated access has remained available to third-party vendors, as well.

Any comprehensive retelling of the saga of Windows kernel access must recognize that this area of Windows has been constantly evolving. Marcus notes that Windows Vista was a real mess, that many changes have been made along the way since then, and that the latest Windows 10 1703 has made some changes that might offer some hope of a more stable world in the future. The problem, of course, is that third parties still need to be offering their solutions on older Windows platforms which are still running just fine, refuse to die, and may not be upgradeable to later versions.

So Marcus holds Microsoft responsible. That's his position. And I commend our listeners to go to grc.sc/985 to get all the details. Anyone who has a techie bent will certainly enjoy him explaining pretty much what I just have in his own words. And Leo, we're at an hour.

**Leo:** Yes.

**Steve:** Let's take another break. And then we're going to look at what happened during Entrust's recent webinar, where they explain how they're going to hold onto their customers.

**Leo:** Some of these never-ending stories are really quite amusing, I must say. I must say. See, you know, you talk about, well, are there options to Bitwarden? There are. And a lot of them are seeing this as an opportunity; right? This is the time. By the way, did my clock disappear? I think it did. Stuff's leaving the building.

**Steve:** Yeah, the Nixie clock just is gone. Wow.

**Leo:** Yeah. I had an interesting giant sword that seems to have disappeared, as well. So if you see somebody going down the street with a sword about yea-long, you might call the authorities.

**Steve:** Looks like the ukulele is still there, though.

**Leo:** The uke, no one's taken the uke. I don't know why.

**Steve:** Okay.

**Leo:** All right. Continue on, my friend.

**Steve:** So Entrust held a 10:00 a.m. webinar last week which included the description of their solution with the partnership we mentioned last week with SSL.com. It was largely what I presumed from what they had said earlier, which was that behind the scenes the Certificate Authority SSL.com would be creating and signing the certificates that Entrust would be purchasing from SSL.com and effectively reselling. There were, however, two additional details that were interesting.

Before any certificate authority can issue domain validation certificates, the applicant's control over the domain name in question must be demonstrated. So, for example, if I want to get a certificate for GRC.com, I need to somehow prove to the certificate authority that GRC.com is under my control. I can do that by putting a file that they give me on the root of the server, the web server that answers at GRC.com, showing that, yeah, that's my server, because they asked me to put this file there and I did. I can put a text record into the DNS for GRC.com, again, proving that I'm the guy who's in charge of GRC.com's DNS. And there's a weak method, the weakest is to have email sent from the GRC.com domain; but when all else fails, you can do that.

So anyway, you need to somehow prove you own the domain, you have control over it. So it turns out SSL.com is unwilling to take Entrust's word for that. So the additional wrinkle that will exist for any Entrust customers who wish to purchase web server certificates from Entrust after this coming October 31st is that they will need to prove their domain ownership, not to Entrust, as they have in the past, but to SSL.com. Not surprising; but still, oops, not quite what Entrust was hoping for.

The second wrinkle is that Entrust does not want SSL.com's name to appear in the web browser when a user inspects the security of their connection to see who issued a site's certificate. No, it's got to be Entrust. So although SSL.com will be creating each entire certificate on behalf of Entrust, they've agreed to have SSL.com embed an Entrust intermediate certificate into the certificate chain, since web browsers only show the signer of the web server's final certificate in the chain. By placing Entrust in the middle, SSL.com will be signing the Entrust intermediary, and Entrust's intermediary will be signing the server's domain certificate. In this way, it will be Entrust's name that will be seen in the web browser by anyone who is checking.

So, you know. The webinar was full of a lot of, you know, all of this how they're going to get back in the good graces of the CA/Browser Forum, and all the steps they're taking, and blah blah blah. We'll see how that goes with the passage of time. For now, that's what they're doing in order to, in every way they can, hold onto the customers that they've got who've been purchasing certificates.

I should mention that the webinar also explained that all of the existing mechanism of using Entrust is in place. Everything is Entrust-centric with, whoops, the exception of needing to prove domain ownership to somebody else. No way around that one.

And Leo, this actually came as a result of our talking about the GRC cookie forensics stuff last week. Something is going on with Firefox that is not clear and is not good. After last week's discussion of third-party cookies, and you playing with GRC's Cookie Forensics pages, several people commented that Firefox did not appear to be doing the right thing when it came to blocking third-party cookies in what it calls "Strict" mode. Strict mode is what I want; but, sure enough, Strict mode behavior does not appear to be what I'm getting.

Under Firefox's "Enhanced Tracking Protection" we have three settings, three overall settings: Standard, Strict, and Custom. Standard is described as "Balanced for protection and performance. Pages will load normally." In other words, third-party cookies, we love you. Anybody who wants one can munch on one. Strict is described as "Stronger protection, but may cause some sites or content to break." And then it details this further by claiming, it says: "Firefox blocks the following: social media trackers, cross-site cookies in all windows, tracking content in all windows, cryptominers, and fingerprinters."

Well, that all sounds great. The problem is, it does not appear to be working at all under Firefox. This issue arose, initially came to my attention in our old-school newsgroups, where I hang out with a great group of people. So it grabbed a lot of attention. And many others have confirmed, as have I, Firefox's Strict mode is apparently not doing what it says, what we want and expect. It says cross-site cookies in all windows. That's not working. Chrome and Bing work perfectly.

In order to get Firefox to actually block third-party cookies, cross-site cookies, it's necessary to switch to Custom mode, tell it that you want to block cookies, then under which types of cookies to block you cannot choose "Cross-site tracking cookies." I mean, you can, but it doesn't work. You need to turn the strength up higher. So "Cross-site tracking cookies, and isolate other cross-site cookies"? Nope, that doesn't work either. Neither does setting "Cookies from unvisited websites." Nope. Still doesn't work. It's necessary to choose the Custom mode, and then the cookie-blocking selection of "All cross-site cookies," with then it says in parens "(may cause websites to break)."

Once that's done, GRC's Cookie Forensics page shows that NO third-party session or persistent cookies are being returned from Firefox, just as happens with Chrome and Bing when you tell them to block third-party cookies. They actually do. Firefox actually does not. Back when I first wrote this, when third-party cookies were disabled, some of the broken browsers, they had really weird behavior. It's why I'm testing eight different

ways of setting cookies in a browser because they used to all be jumbled up. And some worked; some didn't. Some were broken; some weren't. In some cases, when you told it to disable third-party cookies, it would stop accepting settings for new cookies. But if you still had any old, you'd call them "stale" cookies, then those would still be getting sent back. All of that behavior's been fixed. But it's broken under Firefox.

Looking at the wording, which specifically refers to cross-site tracking cookies, it appears that Firefox may be making some sort of value judgment about which third-party cross-site cookies are being used for tracking, and which are not. That seems like a bad idea. I don't want any third-party cookies. Chrome and Bing and Safari, well, Safari's had that all shut down for years. Chrome and Bing will now do it if you tell them to. So, what, do they imagine that they can and have somehow maintained a comprehensive list of tracking domains and won't allow third-party cookies to be set by any of those? The only thing that comes to mind is like some sort of heuristic thing, and all of that seems dumb. Just turn them off, like everybody else does.

You know, there may be more to what's going on here, though. One person in GRC's newsgroup said that they set up a new virtual machine, installed Firefox, and it is working correctly. If that's true, and it's not been confirmed, that would suggest that what we may have is another of those situations we have encountered in the past where less secure behavior is allowed to endure in the interest of not breaking anything in an existing installation; whereas anything new is run under the new and improved security settings. But if so, that's intolerable, too, because it appears to be completely transparent, that is, no sign of that is shown in the user interface. And if that's really what's going on, Firefox's UI is not telling the truth. Which, anyway, that's a problem.

I wanted to bring all this up because, you know, it should be on everyone's radar in case others like me were trusting and believing Firefox's meaning of the term "Strict." I would imagine that some of our listeners will be interested enough to dig into this and see whether they can determine what's going on. As everyone knows, you know, I now have an effective incoming channel for effortless sending and receiving of email with our listening community. So I'm really glad for that. And, you know, I'm getting lots of good feedback about that from our listeners.

Okay, Leo, get a load of this one. PC Magazine brings us the story of a security training firm who inadvertently hired a remote software engineer, only to later discover that he was an imposter based in North Korea. They wrote: "A U.S. security training company discovered it mistakenly hired a North Korean hacker to be a software engineer after the employee's newly issued computer became infected with malware." The incident occurred at KnowBe4, and apparently they didn't - K-N-O-W-B-E and then numeral 4 - which develops security awareness programs to teach employees about phishing attacks and cyber threats.

So, yeah, you know, they're certainly a security forward, security aware company. The company recently hired a remote software engineer who cleared the interview and background check process. But last week, KnowBe4 uncovered something odd after sending the employee a company-issued Mac. KnowBe4 wrote in a post last Tuesday: 'The moment it was received, it immediately started to load malware.'

"The company detected the malware thanks to Mac's onboard security software. An investigation, with the help of the FBI and Google's security arm Mandiant, then concluded that the hired software engineer was actually a North Korean posing as a domestic IT worker. Fortunately, the company remotely contained the Mac before the hacker could use the computer to compromise KnowBe4's internal systems." Right? So it was going to VPN into their network and get up to some serious mischief.

"When the malware was first detected, the company's IT team initially reached out to the employee, who claimed 'that he was following steps on his router guide to troubleshoot a speed issue.' But in reality, KnowBe4 caught the hired worker manipulating session files and executing unauthorized software, including using a Raspberry Pi to load the malware. In response, KnowBe4's security team tried to call the hired software engineer, but he 'stated he was unavailable for a call and later became unresponsive.'" Yeah, I'll bet. Oh, I should also say a stock photo of a Caucasian male was modified by AI to appear to have Asian descent. And that's the photo that this employee submitted as part of his hiring process.

"KnowBe4 says it shipped the work computer" - and get this - "to an address that is basically an 'IT mule laptop farm,' which the North Korean then accessed via VPN."

**Leo:** Oh, interesting.

**Steve:** "Although KnowBe4 managed to thwart the breach, the incident underscores how North Korean hackers are exploiting remote IT jobs to infiltrate U.S. companies. In May, the U.S. warned that one group of North Koreans had been using identities from over 60, six zero, real U.S. citizens to help them snag remote jobs. The remote jobs can help North Korea generate revenue for their illegal programs and provide a way for the country's hackers to steal confidential information and pave the way for other attacks. In the case of KnowBe4, the fake software engineer resorted to using an AI-edited photo of a stock image to help them clear the company's interview process."

So this should bring a chill to anyone who might ever hire someone sight-unseen based upon information that's available online - as, you know, opposed to the old-fashioned way of actually taking a face-to-face meeting to interview the person and discuss how and whether there might be a good fit. One of the things we know is going on more and more is domestic firms, we've talked about it recently, are dropping their in-house teams of talent in favor of off-shoring their needs as a means of increasing their so-called agility and reducing their fixed costs. This is one of those things that accountants think is a great idea, and I suppose there may be some places where this could work. But remote software development? I'd sure be wary about that one.

The new tidbit that really caught my attention, though, was the idea of something that was described as "an IT mule laptop farm." Whoa. So this is at a benign location, where the fake worker says they're located. Being able to receive a physical company laptop at that location further solidifies the online legend that this phony worker has erected for themselves. So this laptop is received by confederates who set it up in the IT mule farm and install VPN software, or perhaps attach the laptop to a remote KVM-over-IP system to keep the laptop completely clean. Either way, this allows the fake worker to appear to be using the laptop from the expected location, when instead they're half a world away in a room filled with North Korean hackers all working diligently to attack the West. I wish this was just a "B" grade sci-fi movie, but it's all for real, and it's happening as we speak. Wow. The world we're in today.

Okay. Some feedback from our listeners. Robert said: "Hello, Steve. I'll try to not take too much of your time, but I'd like to mention one thing that irked me about the entire 'Google trying to eradicate third-party cookies is a good thing' business." He said: "TL;DR: Google tries to get rid of third-party cookies to gain a monopoly in the ad market, not to protect users." He says that. I don't see it that way, but okay.

He said: "First and foremost, eradicating third-party cookies is a good thing, as a vehicle to stop tracking of website visitors. The only reason why Google would be able to actually force website owners to move from their cookie-based ad strategies to something else

(FLoC, Topics, labels, whatever they call it) is that they have a near-monopoly in the browser market." Of course I 100% agree with that. The only way the world would ever be able to drop third-party cookies would be if it was forced to do so. And at this time in history, only Google is in the position to have the market power to force such a change.

Anyway, he goes on: "It's important to keep in mind that Google is still a company that makes most of their money selling ads." Right. Okay, agreed. "Every move they had made so far smelled like they wanted to upgrade their browser monopoly into an ad tech monopoly." Okay, I would argue they already have that. He said: "My suspicion is that it wasn't necessarily the ad companies directly that threatened Google about its plan to eradicate third-party cookies, but rather some pending monopoly concern about the ad market. But maybe I'm just too optimistic about that. So, well, just a thought I felt was a bit underrepresented. Thanks again for the work. Robert."

Okay. So the problem I have with Robert's analysis is that I cannot see how Google is giving itself any special privileges through the adoption of their Privacy Sandbox. While it is absolutely true that they were dramatically changing the rules, everything that they are doing was a 100% open process, with open design and open discussion and open source. And they themselves were also being forced, you know, forcing themselves to play by those same rules that they were asking everyone else to play by.

You know, very much like Microsoft. We were just talking about them versus AV. Microsoft is unwilling to accept the same limitations that they're asking the AV vendors to accept by using the API that they provide. Google, not doing that. They're going to use the same Privacy Sandbox that they're saying everyone else should. So there was no advantage that they had over any other advertiser just because it was their Chrome web browser. And had they been successful in bringing about this change, the other browsers would have eventually adopted the same open Privacy Sandbox technologies. But as we know, that hasn't happened.

I did not have time to address this fully last week due to the CrowdStrike event. So anyway, I'm glad for Robert's note. The EU bureaucrats' apparent capitulation to the slimy tracking and secretive user-profiling underworld, which in turn forced Google's browser to retain its historically abused third-party cookie support, represents a massive privacy loss to the world. This was the wrong outcome, and I sincerely hope that it's only a setback that will not stand for long.

Lisa in Worcester, Massachusetts wrote: "Steve, another intriguing podcast. Many thanks. I find it interesting the influence Google has and doesn't have. It seems more powerful over one company like Entrust than a whole market like third-party cookies. Is it influence, or is it calculated cost benefit analysis that helps Google/Alphabet decide where to flex its muscles? Thoughts from Worcester, Massachusetts. Lisa."

Okay. As an observer of human politics I often observe the simple exercise of power. In U.S. politics we see this all the time. Both major political parties scream at each other crying foul and unfair, but they each do what they do simply because they want to, and they can when they have the power to do so. And I suspect the same is true with Google. Google has more power than Entrust, but less power than the European Union. So Google was able to do what it wished with Entrust, whereas the EU had the power to force Google to do what it wished.

And who knows what's really going on inside Google? I very much wanted to see, as we know, the end of third-party cookie abuse. But we don't really know that Google, or at least that all of Google did. Others are suspicious of Google's motives, and maybe they're right to be. The EU's pushback against Google's Privacy Sandbox might not be such a bad thing for Google. I would imagine that an entity the size of Google has plenty of its own

internal politics, and that not everyone may have identical motivations. So I'm sure that some of Google was pleased and relieved by this turn of events.

But I mostly wanted to share Robert's and Lisa's notes as a segue to observing that this entire issue is more of a symptom than a cause, and that there's an underlying problem. The cause of the actual problem is that, unfortunately, the online collection and sale of personal information has become a large, thriving, highly profitable, and powerful industry all unto itself, and that it may now be too big to stop. This is what keeps the EFF awake at night. We know enough from our previous examination of the EU's extended decision process here to have seen that their decision was directly influenced by commercial interests that wanted the status quo to remain unchanged. And those interests were powerful enough to have their way. The question then becomes, how will this ever change?

The only thing more powerful than a minority of strong commercial interests is a majority of even stronger voting citizens. But people cannot dislike what they're unaware of, and the personal data collection industry has always been careful to remain in the shadows since they know how vulnerable they would be if we, the larger public, were ever to learn a lot more about what was really going on. We've often observed on this podcast that conduct that goes unseen is allowed to occur in darkness, and we know that users sitting in front of browsers have nearly zero visibility into what's taking place right before their eyes on the other side of the screen. Those who perform this spying and data collection claim that no one really cares. But the only reason people don't appear to care is that they don't really know what's going on.

When iOS began requiring apps to ask for explicit permission to track people outside of their own apps, the response was overwhelmingly negative. People who were asked said no. Similarly, it likely never occurs to the typical consumer that their own ISP who provides them with Internet bandwidth and who knows their real-world identity because they're being paid every month, is in the perfect position to catch and aggregate our unencrypted DNS queries, and the IP connections our account makes to remote sites. This data represents a profit center, so it is routinely collected and sold. It's allowed to happen only because it goes undetected and unseen.

Nearly three years ago, in October of 2021, the U.S. Federal Trade Commission, our FTC, published a news release with the headline: "FTC Staff Report Finds Many Internet Service Providers Collect Troves of Personal Data; Users Have Few Options to Restrict Use." And the subhead reads: "Report finds many ISPs use web browsing data and group consumers using sensitive characteristics such as race and sexual orientation." Why would they be doing this if it wasn't of some commercial use to them?

It seems obvious that if any consumer ever gave their permission, it was not truthfully made clear to them what was going to transpire. I certainly never gave my cable provider permission to do that. But I have no doubt that the consumer agreement I originally signed but, you know, never read, or any of the updated amendments which may have been sent to me about it, I'm sure it contained the sort of language we've talked about before, where information about our online use may be shared with business partners and so forth.

Anyway, sometimes enterprise can be a little too free. This is why the protection of consumers from this sort of pervasive privacy violation for profit is the role of government. Unfortunately, government is just people, too, and people can be purchased. In the U.S. at least, lobbyists for commercial interests hold a great deal of sway over how the government spends its time and our tax dollars. The U.S. has a couple of senators who see and understand the problem. But, you know, they're investing a great deal of their time in doing what they can. But most legislators appear to feel they have bigger fish to fry.

I think what all this means is that it's up to those of us who care to do what we can. I'm disappointed that Google appears to have lost this round, but I understand that it probably had no choice. I'm sure we'll be talking about this again, once we see what Google says that they'll be coming up with, you know, as a compromise of some sort.

Okay. Lee Mossner shared a Mastodon posting by Brian Krebs about, speaking of the devil, the collection and reselling of automotive data being done by automakers without their drivers' clear knowledge or permission. Brian posted - and this, you know, I mentioned a couple of the senators who are doing what they can? "Senator Ron Wyden has released details," Brian Krebs posted, "about an investigation into automakers' disclosure of driving data, such as sudden braking and acceleration, to data brokers for subsequent resale to insurance companies. General Motors also confirmed to Wyden's office that it shared consumers' location data with two other companies, which GM refused to identify.

"The senators' letter to the FTC included new details about GM, Honda, and Hyundai's sharing of drivers' data with data brokers, including details about the payments the data broker Verisk made to automakers. Based on information Wyden obtained from automakers, the senators revealed Hyundai shared data from 1.7 million cars with Verisk, which paid Hyundai a little over a million dollars, $1.043 million. Honda shared data from 97,000 cars with Verisk. And automakers used deceptive design tactics, known as 'dark patterns,' to manipulate consumers into signing up for programs in which driver data was shared with data brokers, for subsequent resale to insurance companies." So yes. Technically we're giving permission, but not intending to, or not understanding what it is that will be done as a result.

And of course I have no answer to this other than for us to be aware of what's going on and take whatever measures make sense. I presume that it's no longer possible to purchase any modern vehicle that isn't connected to the Internet and dutifully feeding back everything that goes on within its perimeter to some hidden agency in the cloud. So for the time being that's part of what it means to be an owner and operator.

And finally, Alex Neihaus, one of our earliest supporters, or maybe he was THE earliest supporter, he was with Astaro, and they were advertising the Astaro Security Gateway in the early days. He sent an interesting note asking an interesting question. He said: "Hi, Steve. I tried the DNS Benchmark today running on Windows 11 ARM64 in a VM hosted on a MacBook Pro using Apple Silicon. See the image below." He said: "It appeared to run flawlessly and at full speed." And he says in parens - and you'll like this, Leo. "Windows 11 ARM" - this is Alex saying - "runs faster, in my humble opinion, on Apple Silicon than on any real PC I've tried. It's astonishing," he says.

So he says: "I'm wondering if you have an opinion about accuracy of the app's results in this scenario, emulation of x86 instructions in a VM." He said: "I think I remember you saying the DNS Benchmark is highly timing dependent for accuracy. I wonder if sheer brute computing capability, as provided by an Apple Silicon processor, can overcome the costs of double emulation. Really enjoying Security Now! these days. Thanks. Alex." And Alex attached a screenshot of GRC's DNS Benchmark running on a Windows 11 desktop hosted on a MacBook Pro.

I included this because with the rise of ARM and Windows for ARM finally becoming real after so many false starts, I've been thinking about the fact that I write x86 code, and that all of my utilities are written in Intel x86 assembly language. I've always felt that this meant that my code had unnecessary performance overkill, since it uses so very little of the processor's available resources to run. But this becomes something of an advantage when an ARM-based machine is used to emulate Intel's x86 instruction set. My utilities are objectively small because there are so many fewer instructions in them. And that means significantly less emulation overhead.

So I think that my approach is going to stand the test of time because there's no way any version of Windows, whether hosted on Intel or ARM, will not be able to run x86 instructions one way or another. And the fewer of those there are, the faster the code will go.

And to answer Alex's question, yes to the DNS Benchmark's proper operation under ARM. The only thing the benchmark requires for accuracy is reliable timing snapshots from the system's processor clock counter, and that would be one of the first things the designers of the x86 emulation and its VM would have provided. So accurate, like what is this instant right now, that's what we need. And I'm sure that would be available within a VM emulating an Intel x86 environment. And Leo?

**Leo:** Yes, sir.

**Steve:** Our last break, and we're going to talk about something of potentially significant interest to our listeners. And there is a way to find out, as we will see, if your systems are vulnerable.

**Leo:** Okay.

**Steve:** So we've got some takeaway user action, too. Okay. So Platform Key Disclosure. Today's topic will reveal the details behind a widespread - and by that I mean industry-wide, shockingly industry-wide - security failure that was discovered in the supply chain of hundreds of today's PCs from some of our largest manufacturers. The upshot of this is that these machines are unable to boot securely, despite the fact that that's what they say they're doing. And while that'll be our primary focus, the larger point I hope to drive home is that this is additional evidence to substantiate my belief that Microsoft's Recall is an inherently doomed proposition, at least if we require absolute privacy for the accumulated device history that Recall proposes to aggregate.

The reason for this is that despite all of Microsoft's assertions about how tightly and deeply they'll be protecting the accumulated history of their users, doing so with sufficient security is simply not possible due to the pervasive lack of security that pervades the entire PC ecosystem. It's like Leo asked earlier in this podcast, you know, there's really no security anywhere, is there. It's like, well, you know, there's barriers. But I've always referred to security as "porous," and this is why. Much lip service is given to how securable everything is, yet it keeps getting broken over and over and over.

Okay. So get a load of what's happened now. The security firm Binarly (B-I-N-A-R-L-Y) has coined the term "PKfail" for their discovery of serious problems with the Platform Keys - this is abbreviated PK - being used across our industry to provide the root of trust for our system's Secure Boot technology. Here's what they explain. They said: "Today we disclose PKfail, a firmware supply-chain issue affecting hundreds" - and I should say just shy of 850 hundreds - "of device models in the UEFI ecosystem. The root cause of this issue lies with the Secure Boot 'master key,' called the Platform Key in UEFI terminology.

"The Platform Key used in affected devices is completely untrusted because it's generated by Independent BIOS vendors and widely shared among different hardware vendors. This key is used to manage the Secure Boot databases that determine what is trusted and what instead should not be executed, effectively maintaining the chain of trust from the firmware to the operating system. Given its importance, the creation and the management of this master key should be done by the device vendors following best practices for cryptographic key management, for example, by using Hardware Security

Modules. Specifically, it must never be disclosed or known publicly." Right? Now, the hardware that supports the firmware is a hardware security module. The hardware's designed to keep secrets. But if what you store in there is not secret, then it doesn't matter if you keep it because it's already known.

So they said: "However, the Binarly Research Team found that these keys are generated and embedded in a BIOS vendor's reference implementation as sample keys under the expectation that any upstream entity in the supply chain" - I guess I would call that a downstream entity, but anyway, you know, any subsequent entity in the supply chain - "such as OEMs or device vendors would replace them. When this does not happen, devices are shipped with the original 'sample' untrusted keys in place. Binarly researchers identified the private part of one of these Platform Keys in a recent data dump following a leak. This key is currently being used by hundreds of devices in the market, putting every one of them at immediate risk. A peculiarity of PKfail is that it represents yet another example of cross-silicon issue, as it affects both x86 and ARM devices."

They wrote: "We've developed proofs of concept to demonstrate that attackers with access to a device vulnerable to PKfail can easily bypass Secure Boot by signing their malicious code and thus enabling them to deliver any sort of UEFI rootkit, like the recently discovered BlackLotus. Modern computing relies on establishing and maintaining trust, starting with trusted foundations and extending through operating systems and applications in a chain-like manner. This allows end users to confidently rely on the integrity of the underlying hardware, firmware, and software. In modern systems, trust is typically rooted in hardware-based implementations such as Intel Boot Guard or AMD's Platform Security Processor.

"The root trust is then propagated to the operating system via Secure Boot, which ensures that only digitally signed and verified bootloaders and OS kernels are executed by the boot manager. Secure Boot technology uses public-key cryptography and relies on four keys and databases for authentication. The four keys are, one, the Platform Key (PK)." And they say: "The root-of-trust key embedded in the system firmware establishes trust between the platform owner and platform firmware. Two, the Key Exchange Key (KEK). This key establishes trust between the operating system and the platform firmware. Third is the Signature Database (db), this database containing trusted signatures and certificates for third-party UEFI components and boot loaders, which are thus granted execution. And then fourth, the dbx, which is the Forbidden Signature Database, a database containing signatures and certificates used to sign known malicious software, which are thus denied execution."

In other words, these are specific signatures and certificates that would otherwise be valid because they somehow got themselves signed by keys that are valid, but these are specifically known to be not valid. And I should mention that it was Security Now! Podcast 500. We're at 985. We're approaching the famous 999 boundary and going to a thousand. So this was exactly half of that ago. Episode 500 was the one where the entire podcast talked about this trusted platform technology and the UEFI with platform keys and all that, if anyone wants to go back and get more information.

Anyway, they said: "Each database is stored in its corresponding" - each of those four things - "corresponding nonvolatile RAM variable (PK, KEK, db, and dbx). These variables are authenticated, meaning that when Secure Boot is enabled, updates are only allowed if the update data is signed by a higher level key, the highest level being PK, the Platform Key. The Platform Key can be used to add or remove keys from the Key Exchange Key database, while the Key Exchange database key can be used to update the Signature Database and the Forbidden Signature Database.

"Being at the root of the trust hierarchy, that master Platform Key (PK) plays a critical role in the security of Secure Boot. Access to the private part of the Platform Key allows an attacker to easily bypass Secure Boot. The attacker can update the Key Exchange Key database with a malicious KEK which can be subsequently used to tamper with the Signature Database and the Forbidden Signature Database. Since these databases are used during the verification process, an attacker exploiting PKfail can cause untrusted code to be run during the boot process, even when Secure Boot is enabled.

"The Binarly Research Team discovered that hundreds of products use a sample test Platform Key that was generated by American Megatrends International (AMI). This key was likely included in their reference implementation with the expectation that it would be replaced with another safely generated key. That never happened. Since these test keys are shared with commercial partners and vendors, they must be treated as completely untrusted.

"Several facts give us confidence in this assessment," they wrote. Okay. So here they are, three of them. "One, by scanning an internal dataset of firmware images, we confirm that devices from unrelated vendors contain the same Platform Key, meaning that these keys must have been generated at the root of the firmware supply chain.

"Number two." Get this. "These test keys have strong indications of being untrusted. The certificate issuer contains the clear strings 'DO NOT TRUST' and 'DO NOT SHIP' in all capital letters." Like the common name, the CN, the Common Name in the certificate is "DO NOT TRUST," and the issuer is "DO NOT SHIP." It couldn't be made any more clear. Yet they are trusted, and they did ship.

"Number three," they said. "More importantly, Binarly Research discovered the private component of one Platform Key in a data leak, where an alleged OEM employee published the source code containing the private Platform Key to a public GitHub repository. The private key was stored in an encrypted file, which was 'protected' [they have in quotes] by a weak four-character password and thus easily cracked with any password-cracking tool. Thus the untrustworthiness of this key is clear.

"Shortly after discovering PKfail, it became apparent that this was not a novel vulnerability. In fact, it's quite the opposite. A quick search on the Internet returned numerous posts from users finding keys marked as 'DO NOT TRUST' in their systems, worried about the security implications of that. But even more concerning, we discovered that the same vulnerability was known as far back as 2016, and it was even assigned CVE-2016-5247. Why are so many devices still vulnerable to this issue almost 10 years after its public disclosure?

"The harsh truth is that the complex nature of the firmware supply chain - where multiple companies contribute to the production of a single firmware image, and the security of each component relies on others' security measures - demands inspection capabilities that are far from the current industry's standards or simply unavailable from a technological point of view.

"In 2019, the Linux Vendor Firmware Service project (LVFS) introduced a check based on YARA rules to detect non-production keys. This rule matches on the strings 'DO NOT TRUST' or 'DO NOT SHIP' with the intent of identifying and reporting firmware vulnerable to PKfail. This rule works well when the Platform Key is stored in an uncompressed form, but fails when the key is compressed and stored in a raw section or within the data section of UEFI modules, as is often the case.

"To address this and other software supply-chain security vulnerabilities, Binarly Transparency Platform analyzes firmware images and autonomously unpacks all nested

components, creating a detailed blueprint of the input, allowing for the detection of PKfail and other known and unknown security vulnerabilities.

"To understand the actual scope of PKfail and its historical patterns, we scanned an internal dataset of UEFI images using our Binarly Transparency Platform. This dataset is representative of the UEFI ecosystem as it contains tens of thousands of firmware images released in the last decade by every major device vendor, including Lenovo, Dell, HPE, HP, Supermicro, Intel, MSI, and Gigabyte.

"The macro results of this scan are quite alarming. More than 10% of firmware images in our dataset use an untrusted Platform Key" - meaning a key that was specifically branded, labeled "DO NOT TRUST," "DO NOT SHIP," and has been cracked and broken and is known - "and are thus vulnerable to PKfail." More than one in 10. "When reducing the dataset to only more recent firmware released in only the past four years, the percentage drops to 8%, though remaining at concerning levels.

"The first firmware vulnerable to PKfail was released back in May of 2012, while the latest was released last month, in June of 2024. Overall," they write, "this makes this supply-chain issue one of the longest lasting of its kind, spanning more than 12 years. The list of affected devices, which at this moment contains nearly 850 devices, can be found in our BRLY-2024-005 advisory.

"A closer look at the scan results revealed that our platform extracted and identified 22 unique untrusted keys. The table below reports the five most frequently used keys, along with a breakdown of affected products and vendors." Thank you, Leo, for putting that on the screen.

So what we see in this table, we have the certificate serial number, five different certificates. The certificate subject, that is the CN, says "DO NOT TRUST AMI Test PK," Platform Key. All five of them clearly labeled "DO NOT TRUST AMI Test Platform Key." The issuer is "DO NOT TRUST AMI Test Platform Key."

**Leo:** Who issued these keys? Some guy named DO NOT TRUST.

**Steve:** So where are they in use? Acer, Dell, Fujitsu, Gigabyte, Intel - Intel themselves - Lenovo, and Supermicro. First seen in April of 2018; most recently seen last month, in June, in firmware released on a new machine from some one of these guys in June of 2024. And Leo, what this means is Secure Boot is subverted on that platform. Malware can install itself, even with Secure Boot enabled.

**Leo:** Do you have to have access to the machine?

**Steve:** Physical access definitely allows it to happen. But we've seen many instances where, once malware got into the system, they were able to modify the UEFI boot just using their software access to the machine.

**Leo:** And then the machine can't detect it because...

**Steve:** Exactly. And then you've got a bootkit installed in your firmware permanently, where even reinstalling the OS, reformatting the drive, taking the drive out, blah blah blah, nothing gets rid of it.

Okay. So they explain and finish: "From the certificate subject and issuer strings" - which all say DO NOT TRUST - "we conclude that these keys were generated by AMI." Because it says AMI Test PK. "This conclusion is further supported by how these test keys ended up in devices sold by unrelated vendors, as shown in the last column of the table." Okay. So who? Acer, Dell, Fujitsu, Gigabyte, Intel, Lenovo, Supermicro. A repeat on the second certificate, looks like same people in the third certificate, same people in the fourth certificate, and same people in the fifth. So Acer, Dell, Fujitsu, Gigabyte, HP, Lenovo. Oh, Samsung appeared in the fifth most recently. Oh, earlier. First appeared in 2012, in May of 2012; and last seen in March of 2021. So for that fifth one.

They said: "By looking at the actual product names and models, we found another concerning issue: the very same key is used to protect a highly heterogeneous set of products. For example, the key with serial starting with '55:FB:EF' was found both in gaming laptops and in server motherboards." So probably, what, an Acer gaming machine and a Supermicro server, since Acer and Supermicro were both found to be using that first key. "Moreover, as we can see in the Last Seen and First Seen columns, these keys survive in the ecosystem for years, up to almost 10 years in the case of the key with a serial number beginning '1B:ED:93.'

"When looking at the historical trends of PKfail, several worrisome observations can be made. First, even after CVE-2016-5247 was made public, the release rate of images vulnerable to PKfail retained its previous increasing trend, suggesting that the firmware industry was not responsive to that 2016 vulnerability discovery. This behavior is consistent with the findings from our retrospective analysis of LogoFAIL patches, where we found that the industry needed several months after the disclosure before security patches were ready and propagated to end-users.

"The reaction to CVE-2016-5247 can finally be seen in the period from 2017 to 2020, where the number of vulnerable images steadily decreased. This trend, however, changed again after 2020 and has persisted until current day, with a constant increase of vulnerable devices," which means people forgot about this problem and then started replicating the bad keys once again across their own devices.

"Another observation related to the private Platform Key leaked on GitHub is that this leak did not result in any visible impact on the number of vulnerable devices. This is once again unsurprising when put into the historical context of the firmware industry just not caring.

"To be clear, this leak went almost unnoticed. In fact, they write, "we were the first to report that it contained the private part of a Platform Key. Second, the slow reaction to this security incident. By mining the data provided by the GitHub Archive and available on the Wayback Machine, we confirm that the repository remained publicly available for at least four months before getting noticed and removed by its original author, while it took five months to delete all the forks of the offending repository. Quite concerningly, the leaked key is still in use today in many devices and has been used for quite some time. The first occurrence of this key in our dataset dates back to April of 2018."

Okay. So this means that, while much as been made of Secure Boot, because Secure Boot is utterly reliant upon the secrecy of the private key at its root, around one out of every 10 machines in use today, and even shipped as recently as last month, is only pretending to have Secure Boot because its private key is one of the handful that AMI originally provided - and clearly marked, they believed well enough - as DO NOT TRUST and DO NOT SHIP. Yet shipped they were, and trusted they are still being.

One very cool bit of tech that Binarly shared is the way any Linux or Windows user can check their own PCs for the presence of these insecure keys. They wrote: "Devices affected by PKfail will have the strings 'DO NOT TRUST' or 'DO NOT SHIP' in the subject

and issuer fields of the Platform Key certificate. On Linux, PKfail can be easily detected by displaying the content of the PK variable." And they show the command: efi-readvar -v PK. And that causes Linux to dump out Variable PK, length 862. And then basically you're looking at the PK variable certificate, where you can clearly see subject is CN=DO NOT TRUST - AMI Test PK, and issuer is CN=DO NOT TRUST - AMI Test PK.

Then they write: "On Windows, running the following command in a privileged PowerShell console will return True on affected devices." And this command is a little long for me to read out on the podcast. It's at the bottom of the show notes, page 23. Basically, it is a powershell command Get-SecureBootUEFI and then space PK, then .bytes. That returns a bunch of ASCII, which you then run a match on, matching on DO NOT TRUST or DO NOT SHIP.

**Leo:** Now, will this work on all machines? Because didn't you say some of them were obfuscated?

**Steve:** Yeah. No, no. This will work because it's...

**Leo:** It's already loaded.

**Steve:** ...looking at the certificate.

**Leo:** Yeah. Okay.

**Steve:** Well, it's looking at the certificate of the Platform Key. So anybody who wants to know whether they're one of the one in 10 whose machine has this, is able to run it. And I'm sure, I'd love to hear some anonymous feedback from our listeners who, either under Linux or Windows, run these commands and do or don't find they've got this insecure, well-known key.

The Binarly Research Team recommends that affected users update their firmware when device vendors release an updated version with fixes for this PKfail event. Expert users, they said, can re-key the Platform Key (PK) with a trusted key. The other Secure Boot databases - the KEK, the db, and the dbx - must also be assumed to be compromised, thus expert users should check and replace them with new databases of trusted signatures and certificates. And I'm sure that will be part of the pack that the vendors release.

So just so everyone is clear here, there is no obvious remote vulnerability. Just because you have, like, a bad Secure Boot doesn't in any way make your system actively vulnerable. The primary danger is from local boot tampering with a machine that was believed to be secured against exactly such manipulation. And as I said when Leo asked, we have seen plenty of instances where remotely injected instances of malware were then able to establish persistent bootkit rootkits by messing with the user's motherboard firmware from that machine. That would, you know, and this compromise means that could work again.

So this is not the end of the world by any means. And since this time, unlike previous times, Binarly's research, which is breathtaking in its scope, has generated far more interest than the issue did back in 2016. And since there really does appear to be a great deal more interest in security in general today than eight years ago, I would expect that

the manufacturers of all still supported systems will arrange to respond to the egg that they now all have on their faces because this is all their fault for not ever generating their own Platform Key and just using the one that came with the firmware which said "DO NOT TRUST" and "DO NOT SHIP." There is no excuse for being so negligent as to ship systems with those keys in place.

And again, I'm not going to tell anybody that they should not use Recall once it becomes available. That's not my place. I understand 100% that the reward from its use may well justify the very slim chance that the data it gathers might be put to some use that its owner would find objectionable. After all, that's certainly happening no matter where we go today on the web, or where or even how we drive our cars in the real world. You know, why should this be any different?

But that said, what does appear to be imperative, you know, Microsoft's repeated and well-meaning assertions about their ability to secure this information notwithstanding is for every individual to be given the well-informed choice about whether or not they want this for themselves, this Recall storage, and for that choice to be honored without exception or excuse. Here's another example today that, you know, Microsoft says, oh, yeah, we've got Secure Boot. Turn it on, you know, and let Windows Hello logon. Well, that was violated two weeks ago, but I didn't have time to talk about that because of CrowdStrike crashed the world. So, yeah. Good luck.

**Leo:** I have to say, you know, in order to install Linux, I'm in the habit of turning off Secure Boot anyway. Nowadays you can keep it on in many Linux distros and so forth. But it's not, I mean...

**Steve:** No. It's not. Yes, I agree.

**Leo:** Not a huge loss, I guess.

**Steve:** And as somebody who would love people to be able to boot DOS on their systems, Secure Boot...

**Leo:** Same thing; right? Just gets in the way, yeah.

**Steve:** ...is a thorn in my side.

**Leo:** Yeah.

**Steve:** Yeah.

**Leo:** I understand it was created at the time when we were really worried about BIOS-resident malware, which you can't ever get rid of. So I understand that. Rootkits and that kind of thing. But I don't know. I guess I've lived dangerously.

**Steve:** Yeah, Leo, just use CrowdStrike. What could go wrong?

**Leo:** Chromebooks do the same thing. They have a Secure Boot. Macs now do that, as well. They verify their boot code. It does make sense to do that. I think it really does, yeah. And most Linux systems now support UEFI and Secure Boot.

**Steve:** Yeah. I would say that the vulnerability is the targeted, you know, state-level actor.

**Leo:** Yeah, exactly.

**Steve:** You know, the kind of vulnerability that Stuxnet was wanting to take advantage of, where you've got somebody who is able to get brief, you know, the kind of Mission Impossible thing where they switch out someone's laptop and then take it in the backroom and install the rootkit and then switch them back before they know. Now they've got a rootkit that they didn't have before.

**Leo:** Yeah. Didn't have to take it in the backroom anymore. It saves a lot of time.

**Steve:** We don't think you're worried about, you know, Ethan Hawke coming down from a guy wire...

**Leo:** On a cable.

**Steve:** On a cable, making the switch from hanging from the side of a cliff. Then I think you're probably okay.

**Leo:** All right. This has been another Mission Impossible to figure out what the hell is going on and to keep you safe in the face of extraordinary threats. That's this guy.

**Steve:** That's this ever-changing world.

**Leo:** An ever-changing world. Mr. Steve Gibson, he's at GRC.com. There are many things, many things at GRC.com. I'll mention a few. SpinRite, the world's best mass storage performance, maintenance, and recovery utility. Performance is important. We talked on Ask the Tech Guy about a guy had an SSD, was worried that, you know, what do you need to do. And he had this very elaborate thing of copying everything off the SSD and then formatting it and copying everything back on. And I thought, I don't think that's really what you should be doing. But maybe check out SpinRite. It'll help your SSD if you're having performance problems.

**Steve:** Bye.