SECURITY NOW!

Transcript of Episode #645

The Speculation Meltdown

Description: This week, before we focus upon the industry-wide catastrophe enabled by precisely timing the instruction execution of all contemporary high-performance processor architectures, we examine a change in Microsoft's policy regarding non-Microsoft AV systems, Firefox Quantum's performance when tracking protections are enabled, the very worrisome hard-coded backdoors in 10 of Western Digital's My Cloud drives; and, if at first (WEP) and at second (WPA) and at third (WPA2) and at fourth (WPS) you don't succeed, try, try, try, try, try yet again with WPA3, another crucial cryptographic system being developed by a closed members-only committee.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-645.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-645-lq.mp3

SHOW TEASE: It's time for Security Now!. Finally, the long-awaited what the heck is going on with Intel processors edition. Steve breaks down Meltdown and Spectre - by the way, it's not just for Intel anymore - and talks about what mitigation might involve and what the consequences of fixing this massive flaw could be. There's lots of other security news, too. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 645, recorded Tuesday, January 9th, 2018: The Speculation Meltdown.

It's time for Security Now!, the show where we cover security, now, now, with Steve Gibson. He is our security minder in chief. Hello, Steve. Good to see you.

Steve Gibson: Leo, great to be with you again for another week of exciting stuff for the true geek.

Leo: It was kind of funny last week because you talked about what were later to be dubbed Spectre and Meltdown. And you were kind of matter-of-fact about it. I mean, I heard it. In fact, I even mentioned, gee, that sounds kind of like Rowhammer. But I don't know. You must have understood its impact, but I didn't.

Steve: I did. And as you know, I don't like to go, like, overhyping things and running around with my hair on fire. So I got a little down into the weeds about the performance impact of being forced to flush the virtual memory, the so-called Translation Lookaside

Buffer caches, whenever you made a switch from the app to the OS, which is potentially huge. And I got into, like, explaining what this meant, that it was possible with the Meltdown attack. And so this was the day before then the full disclosures came out that there was so-called Meltdown and also something called Spectre.

Leo: Yeah, we didn't know that much on Tuesday because it was just a hint. We hadn't seen the full report till Wednesday.

Steve: Yes, it was from looking - it was like weird footprints in the Unix source where comments had been redacted that normally wouldn't be, and the explainer page for the changes was missing. And it was like, and then news that Microsoft had been scurrying around, I mean, there wasn't anything definitive. Now we know way more. So I titled today's podcast "The Speculation Meltdown" because, sort of taking more of the 10,000-foot view of this, believe it or not, there was a paper written in 1992 which recognized this problem.

Leo: What? What?

Steve: Twenty-five years ago.

Leo: Oh, that's not good.

Steve: And it was then referred to in more detail three years later, in '95, about like how this kind of thing could happen. And back then it was, oh, but, you know, that would be too hard. Or no one's going to bother with that. And so we, the industry, and not just Intel, but as we know now all of the major chip vendors, AMD and ARM, are all subject to these problems.

The reason is that this has been the way, the so-called "speculation" has been the way all high-performance modern processor architectures are able to continue to squeeze phenomenal performance out of the system. And so this is not just Intel, as we know. And in fact, I was distressed to see that there were some class-action suits that were being filed immediately in kneejerk reaction to this, which I regard as absolutely unfortunate. But anyway, we'll get to all that.

I formulated a general law of cross-task information leakage which reads: "In any setting where short-term performance optimizations have global effect, a sufficiently clever task can infer the recent history of other tasks by observing its own performance." And I worked on that for a while. "In any setting where short-term performance optimizations have global effect, a sufficiently clever task can infer the recent history of other tasks by observing its own performance." Which that's like the meta statement of the problem of how there has always been, and that's what's so interesting about this, there has always been this problem.

And what happened was that sort of, I mean, and this is classic security paradigm that we see over and over and over, is that something that was sitting there for a long time, somebody stubbed their toe on and said, "Ow, wait a minute, what's that doing?" And suddenly it's a big deal. So we're going to have fun at the end of the podcast talking about, again, sort of not like which patch to apply or what to do immediately, but where this problem came from. What's the origin of something which, without overhyping it, is probably not overstating it to call it an "industry-wide catastrophe."

Leo: Wow.

Steve: But there are a few other things to talk about, as well. We're going to examine an interesting change in, well, I write "Microsoft's policy," but more of a Microsoft implementation regarding non-Microsoft antiviral systems that appeared last Wednesday, on the 3rd. An interesting comparison of Firefox Quantum's browser performance when tracking protection is enabled or disabled. Some very worrisome hard-coding backdoors which were discovered in 10 of Western Digital's My Cloud drives, since patched, now public. But we want to make sure anybody with a My Cloud drive knows about this because they're incredibly bad. And if at first with WEP and at second with WPA and at third with WPA2 and at fourth with WPS you don't succeed...

Leo: Try WPA3.

Steve: Try, try, try try try yet again with WPA3, exactly. And here we have another crucial cryptographic system being developed by a closed members-only committee as all the past failures of the Wi-Fi Alliance have been. It boggles the mind that in this day that can still be the way we're doing things when again, five strikes so far. And, okay, they're getting warmer. So we'll see. So I think a great podcast for us.

Leo: A nice bunch of stuff. All right, Steve. You have a picture?

Steve: So our Picture of the Week, yes.

Leo: Oh, yes. Wow.

Steve: It's more what I was expecting when I said last week that the consequence of being forced to flush the page lookup tables, to me, was mindboggling. I'm old school, and I don't really believe any of this works. I mean, it just - it's like, no, that can't get off the ground. So the idea is, as I was explaining last week, the virtual memory mapper in Intel architectures is this three-level lookup table where the address that you're fetching isn't the actual physical address at all. It's an index into a table which is an index into a table which is an index into a table. And you need all of those in order to figure out what the value is.

And you do cache the results, and those tables are cached, and that's what's called the TLB, the Translation Lookaside Buffer, because there's this notion in general in computing that something you've done recently, you are more likely to do again than not. And the classic example is looping. If you are looping over some code, that means you go back to the top, and you do it again with different values or for different inputs or something. But so that says that, oh, look, I'm rerunning instructions that I recently ran.

So this is why the whole concept of caching makes sense. We have dynamic memory, the main motherboard, whatever it is, 4, 8, 16 gigabytes of main RAM. Due to the technology, it fell behind a long time ago in keeping up with our processors. Just no one has been able to make main memory go fast. Now, we've talked about little interesting ideas. There's some cross-wire technology that HP is kind of trying to get off the ground. Intel has it, too: XPoint. So maybe we'll see something dramatic happen. But so far main memory, compared to the way our processors have just leaped ahead in speed, it's the laggard. So it's necessary to use this notion of reusing something or hoping that you're going to be able to reuse something you recently slowly fetched from main memory again.

So all of these modern chips have on-chip caches. And you mentioned L1, L2, L3. They're in a hierarchy. The ones closest to the chip are the fastest and the smallest. And then there's a next level further back, and then a third level. And they each get larger and sometimes slower, but also it's more of a function of access patterns when you've got multiple cores. Now you've got different processor cores all competing with each other for sharing this common pool of cache, all of which is trying to provide, like just arrange to feed these hungry chips data as quickly as they need it.

So also shared are these virtual memory lookup tables. And so what I was saying last week was the idea that crossing from the userland, user code to the kernel would require the flushing of that incredibly valuable because it's so slow to get it to load it. The idea that it would force the flushing of that cache was just like, as I said, I don't really believe any of this works to start with. I've sort of reluctantly agreed that, okay, WiFi seems to be working somehow at ridiculous speeds. And when modems started going to 56K, it's like, what? No. Modems go at 300 baud, or maybe 12, not 56K. But, yeah, they somehow did.

So, okay. So this picture demonstrates what happened on one of three of Epic Games' servers when they installed the Meltdown mitigation. And this is much more in keeping with what I expected. So you can't see it, but there are three chart lines. There's green, orange, and blue. There's a little index down in the lower left, a little key showing them. And so what you see is this sine wave, which anyone who's run a big server is familiar with. That's the diurnal cycle of CPU being busy. You notice it hits a nadir. They're all kind of lowest around 4:00 a.m., and I'm not sure if this is UTC or what time zone. But then it also peaks pretty much on the opposite side of the day. And so that's just sort of generally people are sleeping, and then they're waking up and doing stuff.

Well, that sudden jump is one of the three processors, the green trace, where Meltdown mitigation was applied. And it's about two and a half times. If you look at the low point there of all the traces, just before 4:00 a.m., you could see that it crosses right around 10%. And now if you look up at the green line, it's more than 25%. It's like maybe about 27%. Whereas the other guys are at 10.

So that's, I mean, for their use case, the Meltdown mitigation jumped their processor's utilization, CPU utilization, by about 2.7, which I believe based on the severity of the need to discard this slow-to-acquire cache data every time you switch between the kernel and the user. So anyway, this is the kind of picture that is what I was expecting, rather than this, you know, we've heard 1%, or point something, or maybe 5. It's like, okay. Again, it is completely a function of what you're doing. We've talked often about...

Leo: How much kernel mode access you need compared to how much user mode?

Steve: Well, actually it's how much crossing. It's the boundary crossing.

Leo: Right. So if you go into the kernel, and you read, read, read, read, read...

Steve: And you stay there.

Leo: ...and then you exit the kernel...

Steve: Right.

Leo: ...Then there's that one swap then.

Steve: Right.

Leo: But if you go back and forth, that's when you make the biggest hit, right, because you don't want user mode to have access to kernel mode information. So you flush the cache. You go into user mode. And then you go back to kernel mode. Do they flush it twice? Do they flush it when you go back into kernel mode? No, you wouldn't need to then.

Steve: Correct. You don't need to, well, you don't need to flush it when you go from the kernel, I mean, when you go from the user to the kernel because we trust the kernel. There's no malware there. It's user code looking at the residual from being in the kernel. And so it's when you switch to the user that the kernel needs to remove its what I described last week as "footprints" because essentially, and this comes back to that statement I made, in any setting - and caching is a perfect example. It's not Speculation. We'll talk about that in a second.

But in any setting where short-term performance optimizations have global effect, a sufficiently clever task can infer the recent history of other tasks by observing its own performance. In other words, so caching is a short-term performance optimization. And if the cache is global, that is, if everybody gets the benefit of it, then clever tasks can figure out what's in the cache and what isn't by making a fetch and observing how long it takes, observing its own performance.

And so the 20,000-foot view here is that the way we've been cruising for the last several decades is we've been creating faster performance by creating processors that are increasingly clever about optimizing what they expect to have happen next. But that expectation, whether it's cache, or whether it's specular architecture that we'll be talking about, Speculation, that inherently, inherently allows these kinds of problems. And so that's why this is, as I said before, this is industry-shaking. I mean, again, this has always been sitting there, nobody really worried about it, until now everybody's worrying about it.

Leo: And we don't know who knew about it and when.

Steve: That's true.

Leo: We've learned about it now, but who knows?

Steve: That's true.

Leo: Although, you know, it feels like this is - and this is the other thing I've been speculating on. And I can't remember if it was Bruce Schneier or Brian Krebs talked about the synchronicity, the fact that two different teams simultaneously discovered this. I would guess security researchers kind of run down similar avenues.

Steve: Four teams, actually.

Leo: Four teams, wow.

Steve: Four teams. Well, if you broaden the window a little bit, within a couple months. And that was Bruce talking about this.

Leo: Bruce, yeah.

Steve: And there is a security community. There's conferences. There's papers being published. Everybody's reading everything. And so it's not like there's been no cross-pollination at all. So it may have been, for example, that last summer somebody did something, or like cache timing, that's now a thing.

Leo: Well, and Rowhammer, maybe, something like that, yeah.

Steve: Exactly. So there is something kind of in the air where individual researchers will kind of think, huh, I'm going to spend some time thinking about this. And because they're often researching in a silo within their own group or their own team or their own university or their own corporation, and they're wanting to produce results that are proprietary until they're not, until they're publicly disclosed, especially if it's something like this, where they have to whisper what they found to the parties who are affected, they're not going to be talking about it. So it is interesting, but not surprising, that suddenly this just kind of all happened. It was probably set up by, again, just sort of what was in the air in the last few years and eventually bound to happen.

Leo: Yeah, yeah.

Steve: So we'll get to that further in a minute. Now, you're going to have to pay attention to this one, Leo, because you'll be wanting to talk about this. This is really strange. Okay, I don't know how to read between the lines. This is from last Wednesday, January 3rd, 2018 on support.microsoft.com titled "Important: Windows Security Updates released January 3rd, 2018, and antivirus software."

And Microsoft wrote: "Microsoft has identified a compatibility issue with a small number of antivirus software products. The compatibility issue arises when antivirus applications make unsupported calls into Windows kernel memory." This is something you and I have been talking about now for a while. That is, the problem with instability being created by AV which is running in the kernel, and in order to get the hooks it needs, they're doing unofficially supported things.

So Microsoft writes: "The compatibility issue arises when antivirus applications make unsupported calls into Windows kernel memory. These calls may cause stop errors, also known as blue screen errors, that make the device unable to boot. To help prevent stop errors that are caused by incompatible antivirus applications, Microsoft is only offering the Windows security updates that were released on January 3, 2018, to devices that are running antivirus software that is from partners who have confirmed that their software is compatible with the January 2018 Windows operating system security update."

So listen to that. So Microsoft is only offering the Windows security updates that were released last week to devices that are running antivirus software that is from partners who've confirmed that their software is compatible. Okay. What are the mechanisms of that? How does that work? Okay, get this. Believe it or not, there's a new registry entry. And if you don't do this, you stop getting updates. That is, if you're running Windows without antivirus which has affirmatively set that registry key to say "Yes, I'm compatible, proceed with the updates," you don't get them.

Leo: Now, if you don't have antivirus at all, that key - will that matter?

Steve: Yes. Get this. Down on the second page, if you scroll down from what you're showing on the screen, under "Customers without antivirus," I had to read this, like, three times to make sure I, like, what? "In cases where customers can't install or run [or choose not to] antivirus software, Microsoft recommends manually setting the registry key as described below in order to receive the January 2018 security updates." In other words, you just don't get them. If you don't do anything, and if you've decided I don't want antivirus, then nothing sets that key.

Leo: But I think Defender counts. Defender changes the key. So you would have to be proactively disabling Defender, which is installed by default on Windows 8, 8.1, and 10.

Steve: Correct. Correct.

Leo: But the weirdest situation is there are a number of antiviruses that fixed the kernel calls, but didn't set the key. Defender does, though. So you should be using Defender. And most people are.

Steve: Yeah, and that's what you and I have been saying now for some time is I know that there are a lot of people who have been using some, whatever their AV is, for decades, and they're just sticking with it. But these kinds of problems occur. So, yeah. I just...

Leo: Yeah, because also we've been talking about is don't use a third-party antivirus because they do things like these unauthorized kernel accesses.

Steve: Right.

Leo: It's not good.

Steve: Right.

Leo: I thought that - I remember when Microsoft came out with Windows 7 they were deprecating this and said don't do this because we're going to prevent you from doing this. And I thought that they did, but I guess they didn't. They backed down.

Steve: Yes. There's antitrust issues. Nobody, like, what happened to third-party firewalls? Those went the way of the whatever. They're gone. There's still third-party AV. Now, their document does say for Windows 7 and Server 2008 R2, which is the equivalent server version of the Windows 7:

"In a default installation of Windows 7 SP1 or Windows Server 2008 R2 SP1, customers will not have an antivirus application installed by default. In these situations, Microsoft recommends installing a compatible and supported antivirus application such as Microsoft Security Essentials or a third-party AV application. The antivirus software must set a registry key as described below in order to receive" - that is, in order for the user to receive - "the January 2018 security updates." And I should also mention, it's not just this month. Down in their Q&A they ask themselves the question, how long will Microsoft require setting your registry key? And basically, yeah, from now on. That is, if this doesn't happen, if something doesn't set that key, you never get them again. So...

Leo: Ai yi yi.

Steve: I know.

Leo: Well, part of it, and I'm reading between this lines in this bulletin, is that if you do have the blue screen because you installed an incompatible program, Microsoft can't guarantee that you can fix it. It could be permanently screwing - you'd have to reinstall Windows.

Steve: Yeah.

Leo: Terrible. What a mess.

Steve: And in Microsoft's defense, we know this is a pain for Microsoft. That is, third-party AV, anything that is allowed into the kernel that starts poking around and hooking things that it's not supposed to and using undocumented calls, clearly what happened is, with this month's release, Microsoft changed something. Well, they're allowed to change their own kernel because it's theirs. But antivirus that wasn't verified against these patches - and how could it be if Microsoft just said, oh, here's a patch - I'm sure that that change collapsed some things, and Microsoft said, oh, crap. We can't push this out to users who have incompatible AV. They'll get blue screens. So they created the registry key to create a trapdoor that would prevent the update from installing itself if this key did not exist or was not set properly.

So, again, this is not Microsoft's fault, but it all does sort of create a little pressure against third-party AV. I mean, there's probably enough pressure against third-party AV already that this doesn't really, in the grand scheme of things, make much difference in terms of being more pressure. But again, as we have been saying, it's another reason just to use Security Essentials or Defender or whatever, just let it take care of you. So anyway, an interesting bit of information that I wanted to make sure our listeners knew about in case somebody had shut down their Microsoft AV in Windows 10 or didn't have Security Essentials or anything else from Microsoft installed in Windows 7 and is kind of scratching their head, hey, where's my update? Especially this week, when everybody's wanting updates like crazy important.

Leo: Well, we'll get into another reason why you may not get the update in a little bit, too.

Steve: Yeah, that's right. So AMD. Anyway, so somebody sent me a tweet from Jason Kint, probably Simon Zerafa. So thank you, Simon, if it was you. I didn't make a note of who forwarded it to me, so I'm sorry if it was somebody else not Simon, but thank you anyway. So Jason tweeted: "Seriously. Drop Chrome. Download latest version of Firefox called Quantum and turn on Tracking Protection in the settings. Tracking protection will also help mitigate ad fraud. And," Jason writes, "it's Google's Achilles heel."

So I thought, okay, what? So I did some digging, and this is a little interesting. So first of all, that chart you're showing needs to be treated with some skepticism. It's Mozilla's chart. However, if it's to be believed, it shows that Chrome average page load time per browser was 7.7 seconds. Chrome Incognito didn't change, 7.7. Firefox Quantum normal mode was a little touch faster at 7.3 seconds. But turn on Private Browsing with Firefox Quantum, and it drops by more than 50%, to less than 50%, to 3.2 seconds.

I also wanted to take a moment to note that I heard Andy in last week's MacBreak Weekly give a shout-out, surprisingly. He was jumping up and down. And I think I've heard, I think, other TWiT hosts or people on the network in general, I mean, it's more than just me noting, holy crap, Quantum is really fast. In fact, you, Leo, I think, had noted that the latest Firefox browser, Quantum, was a so-called "quantum leap" forward in performance. But I got a kick out of hearing Andy saying, wow, he's running them side by side, and his measure is where does he generally end up opening new tabs or something. He had sort of a heuristic.

Leo: But you get even more benefit if you turn on tracking protection. That's a huge, apparently, difference.

Steve: Yes. And so what Mozilla wrote was that "Most browser performance benchmarks focus on the use of a regular browsing mode. But what about private browsing? Given that private browsing use is so common" - and they linked to a DuckDuckGo Private Browsing PDF, which has some interesting stats in it, I mean, it's like for people who - there are a lot of people who've never heard of it, don't know what it is, so obviously never use it. But there are people who know about it and from time to time - typically the example was when doing a search for something embarrassing, they would turn on private browsing in order to not leave footprints behind. But anyway, apparently it is like maybe 50% of users use it with some regularity who know that it exists. And of course there's a big chunk of people who don't know.

Anyway, Mozilla says, "We wanted to see how Firefox's Private Browsing compared with Chrome's Incognito when it came to page load time, that time between a click and a page being fully loaded on the screen." Now, I want to mention that the contention is that tracking script is slowing things down. But tracking script may not matter visually. So it's one thing to say, oh, look at our instrumentation that showed how long it took for everything the browser needed to fetch to finally get fetched. But that may not represent the user's experience.

So what I want to do is I want to plant the seed in our listener's ear that, well, see how your own mileage is with this. Try it yourself. So what their results were across the top 200 news websites tested, the average page load for Firefox Quantum Private Browsing, as I mentioned, was 3.2. Everybody else was like 7.7. So like 2.4 times faster than Chrome in Incognito mode. And so the primary reason for the dramatic difference, they write, is that Mozilla's private browsing mode automatically activates tracking protection; whereas Chrome's Incognito mode does not.

So first of all, okay, that's of interest because you'd sort of think that, if you're in Private Browsing or Incognito, you'd like to have tracking protection off. Well, Chrome keeps it going, and by default Mozilla's Firefox Quantum turns it off. The takeaway here, though, is that anyone using Firefox Quantum can change that setting. It is possible to flip a switch. You go to Privacy and Security, scroll down to find the Tracking Protection section, and you can change it from Tracking Protection During Private Browsing or Always.

So what they're asserting is that you could probably, with Firefox Quantum, obtain the benefit of that speed. And again, instrumentation measurement versus how it feels may be two different things. That is, if scripts continue to load in the background that are only for tracking, but they don't slow down your browsing experience, then okay, it's sort of folded into you looking at the page and figuring out what's going on, and it may not matter. But I wanted to let our listeners know that tracking protection can be set to always and certainly lower your bandwidth consumption, apparently significantly. Maybe you'll see an increase in performance.

And Mozilla did note that, when enabling this for always, keep in mind that tracking protection may block social like buttons, commenting tools, and some cross-site video content. So you may find a bit of a compatibility problem. And probably, if you're already running with uBlock Origin enabled by default, it's probably already doing a lot of this speedup. And there you have the ability, as we've discussed, of disabling it on a per-site basis; or if something seems broken on the page, you can try turning it off and maybe things will go better.

But what we have seen is that, unfortunately, the tracking - and think about what that means in terms of speed. That means that well more than half of a page load time is now blocked by tracking protection, which says this stuff that we don't see, but which is increasingly JavaScript and busy doing stuff, not always for our benefit, is now taking up a substantial percentage of the pages of the top 200 news sites. And also that's another point. It might be that that represents top 200 news sites could be a bit of a skew from where we normally go. Just there maybe more, I mean, news sites are generally trying to keep themselves alive by generating revenue.

Leo: Any site with an ad, almost every site with an ad, the ads will be - it's kind of effectively an adblocker. So you should be aware of that, that if you're going to a site you like, you are disabling their ability to monetize you entirely, not just with ads, but with tracking, like Google Analytics. They can't even tell you're there. So I understand people's privacy concerns. But don't think that using it always is anything less than an adblocker because it's effectively an adblocker. You are telling a site that you visit, whether it's Engadget, the Verge, or TWiT, we don't want you to monetize us.

Steve: Actually, it's a little better than that because it uses a curated list. And if a site does obey the DNT, the Do Not Track, then it's not on that list.

Leo: Yeah, but if you have ads, you're going to - like we, I mean, we obey Do Not Track. But we don't personally, but because we have ads, it will block the ads on our site.

Steve: Ah, okay.

Leo: So unless they have first-party ads or somehow they're doing an ad server that doesn't, you know, we even run our own ad server so that people don't, I mean, we do everything we can to protect people's privacy, and you still block the ads on our site.

Steve: Right, right.

Leo: You're basically saying, no, I don't want to give you any money. So it's something to be aware of.

Steve: Well, and as we've discussed, sites are increasingly detecting that and saying, hey, you know, we need to be able to show you ads in order to support ourselves. Please consider allowing the ads to come through so we can make some money and stay on the air.

Leo: Right. And then maybe people will disable their adblocker. But if they have tracking protection on, they're not changing it. I think that's part of the problem I have is you're doing adblocking without knowing it.

Steve: Yeah, good point. So for quite some time, it turns out that Western Digital, and apparently D-Link also, but primarily Western Digital My Cloud drives had a hard-coded - it's hard to even believe this - a hard-coded backdoor. It got a lot of coverage in the last week. A researcher, James Bercegay, with GulfTech Research and Development, discovered and reported the flaws to WD, to Western Digital, back last summer, June of 2017. He then waited patiently for Western Digital to release firmware updates.

He took a close look at several of the My Cloud drives, 10 of them, by the way. I think there were two that weren't affected. Ten were. And he found a bunch of problems. They allowed unrestricted file upload so that, for example, a PHP file which was found on the WD's My Cloud's built-in web server allowed an attacker to upload files to the device. He was able to use the flaw to upload web shells to the device, which in turn granted him control over it. There was, as I mentioned, a hard-coded backdoor which was present in every single one of these My Cloud drives. And the reason that there's some obviously D-Link connection is that the admin username was mydlinkBRionyg, and the password was abc12345cba.

So this is sitting in all of the drives. The point of them is to be on the Internet, exposed, so that you're able to access them. And they all have a hard-coded backdoor which would have allowed anyone who discovered it, and who knows who did, until this was fixed, to have full access to your drive. And there was a cross-site request forgery bug that could be exploited to execute pranks and rogue commands. For example, you could change the device's interface language so that, when you brought the website up, it would be in a language you didn't speak and had a hard time even changing back because you couldn't find the button to do that.

So the middle of last month, mid-December, an independent and highly detailed disclosure of the vulnerabilities was publicly posted by somebody else. And so the good news is that Western Digital had fixed this in November, so the month before. So the takeaway for our listeners is you want to make sure that you are at firmware 2.30.174 or later. Or just, if you're a WD My Cloud drive user, definitely update your firmware, your drive's firmware to the latest, which will have all of those bugs, including the hard-coded backdoor, removed because you don't want your WD drive and all of its contents probably globally accessible and able to execute root, you know, bad guys.

Oh, and these are wormable. So this was a wormable set of bugs that would have allowed, had this gotten loose, it would have been part of a Mirai botnet in a few weeks because they would have added it to the code, and the drives would have been out scanning for each other and becoming compromised. So not good. Oh, and a Metasploit module now exists, making this easy to do. I looked at the documentation through Metasploit, and there's nothing particularly new or newsworthy that I didn't already talk about. Basically they're just covering the same sort of stuff.

So how not to ever learn the lesson? Announce yet another WiFi specification developed in secret. And boy, are these guys proprietary. So this is - get this. Wi-Fi Alliance(R), and then we have the "circle R" to remind us that that's a registered mark.

Leo: Hell, yeah.

Steve: Yup, Wi-Fi Alliance(R) introduces security enhancements. And their announcement, and this was yesterday at CES, new WiFi, oh, "circle R," that's our trade, we also own that, registered trademark. Security features available in 2018. Now, okay. That's good, except - so dateline Las Vegas, Nevada, January 8, 2018: "Wi-Fi Alliance(R) introduces enhancements and new features for Wi-Fi Protected Access(R)" - oops, "circle R," we own that phrase also. We don't have good security; but. boy, we've got our trademark patent attorneys. They've locked down all of these phrases that you have to give them credit for - "the essential family of Wi-Fi CERTIFIED(TM)" - that's all caps, oh, and that's a TM, trademark on that.

Leo: Oh, my god.

Steve: "...security technologies for more than a decade." Yes, and we're suffering through it for more than a decade. "Wi-Fi Alliance is launching configuration, authentication, and encryption enhancements" - because, you know, we haven't got it right yet. That's me. They didn't say that - "across its portfolio to ensure Wi-Fi CERTIFIED(TM) devices continue to implement state-of-the-art security protections." Somehow they didn't trademark that phrase. "WPA2(TM) provides reliable security used in billions of WiFi(R)" - we own that phrase - "devices every day and will continue to be developed" - anyway, it goes on like that. I won't bore our listeners.

Anyway, they've trademarked and circle R'd every possible phrase that they want to make sure they get credit for. Unfortunately, they're still struggling to get the security correct, as I said. We went through WEP, WPA, WPA2, then we had the single button configuration where it turns out you didn't have to guess all, remember all eight digits at all. The eighth one was a check digit. You could guess them in two separate sets of four and then three, totally shredding that security. So these guys, I mean, they just cannot get it right. However, there are, you know, they're going to keep trying.

And what I wanted, the useful bit of information here is that there will be a WPA3. They're being mum about what it will have. That is to say, no open spec. No ability for actual industry academics and non-paying members of the Wi-Fi Alliance(R), to see what they're doing. But we'll have a chance to take a look at it after everybody has already got it deployed; and, with any luck, maybe they will have made fewer mistakes. So they say four new capabilities for personal and enterprise WiFi networks will emerge in 2018 as part of - wait for it - Wi-Fi CERTIFIED all caps WPA3(TM).

Two of the features will deliver, they are saying, robust protections - we don't know that, of course, until somebody else sees them - even when the users choose passwords that fall short of typical complexity recommendations. So, oh, maybe they're implementing some sort of password-based key derivation function, which would be handy. Everybody else has that now. And, they say, we'll simplify the process of configuring security for devices that have limited or no display interface. Okay, sounds good. IoT stuff, light bulbs and so forth. Another feature, they write, will strengthen user privacy in open networks through individualized data encryption.

Now, that sounds great. Let's hope they get it right. Some speculation that I found suggests that maybe what they will be implementing is a Diffie-Hellman key agreement. We've talked about how that works in the past, where anyone can observe the two endpoints sending stuff back and forth to mutually arrive at a key and, even while seeing all of the back-and-forth traffic, still be unable to know what the key is.

So the presumption is that maybe there will be some sort of a public key agreement protocol like Diffie-Hellman which will then be used to encrypt the standard WPA encryption, which would allow what's known as "opportunistic encryption," meaning that, if you have WPA3 at each end, and that'll take some length of time for access points to come up to speed, for example, in coffee shops and so forth and restaurants and airports. But the idea would be that no longer would your non-password-based WiFi connections be subject to simple eavesdropping. And that's a great step forward. So again, if they didn't screw it up in implementation and definition, then yay, that would be good.

And finally, they say, a 192-bit security suite. I guess for whatever reason, probably performance, maybe, they're not wanting to jump all the way to 256 where everybody else is. But it's better than 128. So they're saying that that's aligned with the Commercial National Security Algorithm Suite from the Committee on National Security Systems. And I don't know who any of those people are. We've never talked about them before, the CNSA? Oops, it's got NSA in its name. Well, okay. Will further protect WiFi networks with, they're saying, with higher security requirements such as government, defense, and industrial. And maybe this fifth shot will be better than the previous four. Let's hope.

So anyway, they've got their trademark all over everything. WPA3 announced yesterday at CES, will happen apparently sometime this year, and sounds like it's got some good new features. So good for that. My only wish is that it were being developed the way other open systems are, where you freely invite the community of people who know a lot to take a look at it because, in the past when that community of people who know a lot have looked at every single thing this Wi-Fi Alliance has ever done, they've found serious glaring problems with it. But again, that's not the way they want to play the game.

Also, and this is my last piece of news before we get to some miscellany stuff and then talk about the Speculation Meltdown, is that the Let's Encrypt gang are celebrating their success in 2017 and have talked about their plans for 2018. In their "Looking forward to 2018" posting yesterday, or actually it was dated - oh, last month. They said: "Let's Encrypt had a great year in 2017. We more than doubled the number of active unexpired certificates we service to 46 million. We just about tripled the number of unique domains we service to 61 million. And we did it all while maintaining a stellar security and compliance track record." And we'll talk about that in a minute.

"Most," they write, "importantly, though, the web went from 46% encrypted page loads to 67%, according to statistics from Mozilla. A gain of 21 percentage points in a single year," they write, "incredible." And I agree. That's great. They said: "We're proud to have contributed to that, and we'd like to thank all the other people and organizations who also worked hard to create a more secure and privacy-respecting web."

Then they finish with a paragraph: "While we're proud of what we accomplished in 2017, we are spending most of the final quarter of the year" - that is, 2017 - "looking forward rather than back. As we wrap up our own planning process for 2018, we'd like to share some of our plans with you, including both the things we're excited about and the challenges we face."

Anyway, so in summary, where they go with this post is that they are going to be giving us a next generation of their ACME protocol, which will support some additional features. They will, during 2018, be allowing the use of wildcard certificates for the first time. So you'll be able to get a *.domain.com or whatever. Which will then allow that single cert to be used on multi-home systems within that domain or on multiple machines behind a network concentrator, whatever, rather than - and each with a different domain name, rather than requiring that each of those machines obtain its own certificate. And they're going to be adding an elliptic curve, an ECDSA elliptic curve root to their root that will then allow them to be issuing elliptic curve certificates, which have lower computational overhead, higher performance. And that's good looking forward.

All of what they wrote is true, but it is also true that an inevitable consequence of the automation of DV, that is, domain validation certificates, which is what they're issuing, is fraud. And while it's true that a greater percentage of the web is encrypted, and that's good, it is also unfortunately true that a phenomenal number - and we've discussed this in the past, a phenomenal number - of Let's Encrypt-issued certificates are fraudulent. I mean, they are being used for fraudulent purposes.

And again, there's nothing to have prevented that from happening with domain validation certs before Let's Encrypt. But the automation of this process has really taken the lid off. And the numbers of fraudulent certs being used essentially to spoof security and get the closed padlock and better treatment in the browser URL to seduce people into believing they are where they're not is phenomenal. So, while, yes, it's good that more of our connections are encrypted, it was inevitable, I think, that we were going to see a significant jump in the abuse of this kind of automated cert issuance, and indeed we have.

And this is a note I got middle of December, last month on the 19th, that caught my eye. I mentioned it tangentially last week, I think, or the week before. The subject was "Thanks for giving me $728!!!!" with four exclamation points.

Leo: Oh, yeah, you mentioned this, from the drip, the Bitcoin Faucet.

Steve: Yes, exactly. Someone named David L. in Utah wrote to both of us. He said: "Steve and Leo, I've been a Security Now! listener since the single digits" - and wow, okay, that's the beginning. So he says: "But I did not buy a copy of SpinRite until my computer wouldn't boot because of some error." Then he wrote: "Whatever it was, SpinRite fixed it. Thanks, Steve." And then the reason that I had brought this up before was he said: "Also thanks for $728. After your first episode about Bitcoin in 2009, I decided to download it and fiddled around for a week. My computer was too weak to mine, but I did go to a website called the Bitcoin Faucet, which gave me five bitcoin cents for free. I finally found my wallet in my backup drive" - which maybe thanks to SpinRite is still working - "and I just sold them and netted $728."

Leo: Nice.

Steve: But he says: "By the way, it took my old computer two weeks to process the 157GB block chain, which used to only be 250MB when I got my five cents."

Leo: Yeah, exactly.

Steve: "Anyway, I must say it pays to listen to Security Now!."

Leo: Yes, it does. Very nice.

Steve: So very cool.

Leo: Happy ending.

Steve: So I have a couple of closing-the-loop bits, feedback from our listeners. And then we will take our last break and talk about the Speculation Meltdown. Mike Gatzke, I guess - sorry, Mike, if I mangled your last name. He wrote, or tweeted. He said: "I've started using Firefox with the Tree Style Tab add-on. Is there a way for Firefox to remember the tabs I have open?"

So to Mike and anybody else, absolutely. I use an add-on called Session Manager which is fabulous. It was written by the Mozilla developers, so they've nailed it. Because, you know, they have to know how Firefox works. And it is far more than just session management. That is, you're able to take sets of tabs and save them. You can create named sessions. You can export sessions, like email them to yourself and get them somewhere else. You can do all kinds of things with it. So anyway, I highly recommend it. I've been using it for years. Session Manager is the add-on.

And then we will talk - actually now we should talk about it, probably, because I'm going to talk more about theory than practice when we talk about the Speculation nightmare. But this started with, well, of course a lot of people have problems with what began happening with their computers. The first I saw of it was Richard J. Wilcox, who tweeted to me and to Simon Zerafa, he said: "I had the same experience as reported in this article." And he quotes a SecurityAffairs.co posting. He said: "The Microsoft update bricked" - not quite the right term in this case - "my AMD Athlon 64 X2 6000+ computer."

And so this was an article, one of many, of AMD users - and you referred to this at the top of the show, Leo - who were having problems with the AMD Athlon. And also the Semprons were doing the same thing. Woody Leonard got in on this dialogue in Twitter. And he tweeted, he says: "I'm getting reports of a lot of AMD Win7 machines blue screening, plus a handful of others - Intel, Win10, various mixtures. No common theme as yet except Win7 AMD Athlons really took a hit." And of course we know since then that Microsoft has officially stopped updating AMD and blamed AMD for this. I don't know if you saw that, Leo.

Leo: Oh, I didn't, no.

Steve: Yeah, Microsoft has an official statement saying that the documentation which AMD provided to Microsoft has some errors, which is the reason that this bricking was occurring. And again, I've misused the term. What actually happens is that the reboot after the update will blue screen. Then Windows will recognize it had a problem. It will try it again. And after, I think it's two, might be three, then it will do a self-rollback to the version that worked. But apparently that's a continuous loop.

There was actually one funny posting that I got a kick out of. It was posted at the Microsoft forum. Someone wrote: "I have older AMD Athlon 64 X2 6000+, Asus motherboard. After installation of" - and then he quoted the Knowledge Base number which is the one that we're all getting, the one that is causing problems - "the system doesn't boot. It only shows the Windows logo, without animation, and nothing more. After several failed boots, it does a rollback. Then it shows error" - and he shows the hex error 0x800f0845.

"Unfortunately," he writes, "it seems it's not easy to disable the automatic updates without group policy edit tweaks, so it tries installing and rolling back the update over and over." He says: "Sfc/scannow shows no problem. In-place upgrade also doesn't seem to help." He says: "I can try full reinstall, but I doubt it will change anything. It seems like the update is binary incompatible with my old CPU." Then here's what I got a kick out of. He says: "I understand that making the machine unbootable is the best protection from remote exploitation. But I would rather have the OS working."

Leo: It is the best, yeah.

Steve: Exactly. If you can't get that machine to boot, nobody's going to get in there. So anyway, it's been a mess. I am sure Microsoft and AMD are working together to figure out what it was that Microsoft didn't understand or that AMD posted incorrectly or provided incorrectly in order to fix this. And then I don't know what Microsoft will do. I guess if you roll back maybe, I don't know, I'm sure Microsoft must have some path through un-blue screening these systems that keep trying to reload the same thing over and over. Maybe it will check to see if the patch has been changed. If Microsoft changes it, then it'll redownload one that no longer has the problem. But, yikes, a bit of a nightmare for those people.

Johnny Brian said: "I have an older Mac Pro as my main computer. The latest OS it can run is 10.11.6. It's not clear if Apple will patch this OS for Meltdown." And then he links to a support.apple.com posting. "Should I be concerned? Replacement is not an option. Should I disconnect it from the Internet?" And, you know, that's a tough call.

I'm of the opinion that, first of all, the exploitation now may not occur if Meltdown has been mitigated. It's really only a problem, well, the performance overhead will probably not affect most typical users. It's largely very active multitasking, multiprocess systems where there's a lot of these ring transitions going on. I guess I would take a wait-and-see, both with what Apple chooses to do and whether we see some chip support mitigations. There is the possibility of some chip support mitigation, although it may not, and early snapshots suggest they will not, address Meltdown, but will address the other Speculation problems.

So the only thing I could say, Johnny, is to wait and see. Certainly many people will be in your situation, and it's hard to imagine that Apple won't offer some sort of mitigation. So I don't know.

Leo: I'm sure Apple will mitigate all of its devices. And then there's always hope with the processor manufacturer; right? Because Intel and AMD and Qualcomm and everybody say they're going to do their own mitigations.

Steve: It's complicated. And we will be talking about that next. And it's not looking, I mean, okay, so anyway, it's complicated. And so there's no easy way to summarize it, so we'll talk about that in a second.

Leo: Got it.

Steve: However, you'll remember our Picture of the Week last week, Leo, that we got such a kick out of, which was the fan blowing on the switching matrix with the duct...

Leo: But we couldn't read the fine print.

Steve: We couldn't read the fine print. Well, leave it to one of our listeners, who found from TheDailyWTF.com the so-called "Sophisticated Cooling Apparatus." By going back to the original picture and blowing it up, the fine print has been made legible. Because there's an asterisk in the warning that all UKTVs will be taken off the air. And the fine print says: "*In a matter of minutes, probably." So that answers the mystery. It won't take long for this thing to overheat and cause a meltdown.

Ian Beckett, who is a frequent correspondent, says: "I'm wondering how vulnerable ARM controllers in SSDs are to Spectre, Meltdown, and the like?" So I would say not at all. An ARM controller in an SSD isn't going to be running code that you download from the Internet. I mean, that's really the danger is that somehow malicious code will be running in a process in a multiprocess system. Mostly cloud systems, I think, are in danger. But ARM controllers are not. And also...

Leo: They're probably not running user mode stuff at all, are they? I mean...

Steve: Correct. They're probably not doing that. Also, for example, the Raspberry Pi is ARM based; but it, too, is not vulnerable because it's using an ARM license and ARM architecture that lacks Speculation.

Leo: Same with the Apple Watch.

Steve: Exactly. And I would imagine the Raspberry Pi chose that because they didn't need that level of crazy performance, and it's a much cheaper license and a much smaller die size, thus much less expensive to fabricate an ARM that uses a much less expensive chip. So we're fine for embedded applications which are certainly going to be using inexpensive controllers and not wide open the way cloud-based Azure and AWS and the like, Google computing platforms are.

Andres Vidal said: "I just read that MongoDB noticed a 10 to 15% impact on HVM hypervisors from patches applied to their AWS infrastructure. That seems really high!" And as I've noted, that doesn't surprise me at all. What I think we're going to see is an initial reaction that has a stronger impact. And actually I mentioned this, I guess on Saturday on The New Screen Savers when I was on with you, Leo, at the beginning of the show, is that I think there will be an initial sort of panic reaction mitigation and that, as we get then chip-level solution, although it's not clear that we're going to get that - well, again, it's complicated.

So I don't want to say something that is wrong. I want to cover it correctly in a minute. But again, that level of impact does not surprise me because, if it's necessary to flush the cache, and the application, like a big database application which may be crossing back and forth and doing a lot of low-level or disk-level I/O, that's going to have - that's the type of application that is going to see a larger end performance hit.

Oh, and I just wanted to put this in here. Trevor Welch asks why Security Now! is not on Spotify, but other TWiT shows are. And I'm thinking maybe he just didn't see it. Or it would surprise me if it wasn't.

Leo: I'll check. Yeah, I mean, it should be.

Steve: Yeah. Okay. So anyone who went back and reread our series on how processors work would have sort of encountered some of this because we've talked about the phenomenal lengths to which modern processors now go in order to squeeze every, I mean, truly phenomenal, squeeze every last possible bit of performance out of their design in order to give us the kind of performance that we're used to getting now. Essentially, every possible stop has been explored and removed. This notion of Speculation, we've already pretty much talked about the Meltdown problem, that the Meltdown problem is a way that was found of leveraging the caching of virtual memory pages which are inherently shared globally because there really hadn't been a reason not to.

Now, I should mention that there is a feature in Intel's recent processors, PCID. And I referred to this last week. PCID is a page context identifier which allows, I mean, it basically solves the Meltdown problem. If you have, and if your OS is using the PCID, it is able to tag the pages with the identifier for the process, which provides the isolation which we need on a process versus OS basis without requiring the pages to be flushed. So the PCID support was added, I want to say 4.16 version of the Linux kernel. It is there. It has been ignored because nobody saw a particular need for it.

Well, that changed last week. And the good news is, when an operating system supports this PCID, and when your chipset supports it, that's another part of the key is that I don't remember now which family PCID appeared in, but it's been around for maybe a decade, but not before. So this is something that Intel added. Maybe it was prescient, or it's just sort of more thorough. To be able to tag cached pages with the process ID is a good thing. So the kernel needs to support it. It exists in Linux. It has not been used until now. It will now be used and will prevent this dramatic performance slowdown that we've seen.

So that's why this is a little bit of a nuanced response. That huge drop in performance that we showed on this week's Picture of the Week was a system that implemented the cache flushing mitigation for Meltdown, not PCID. It probably wasn't available. We don't know what OS they're running on. So that's something that I'm sure Windows will be getting soon, if it doesn't already have it. Linux has it. It hasn't had it enabled. So I'm sure it will. And it does require that the chipset you're running on support this PCID capability, and really old ones don't. But it has existed in Intel chips for some time. So that's the Meltdown, and this ability to use subtle timing in order to sense whether something is in the shared cache or not.

Now, mentioning timing is useful because all that has to be done is that access to high-resolution timing would solve this problem, that is, limiting access to high-resolution timing. And I mentioned this also last week when we were first talking about this. It's not really clear that user mode, userland code, that is, non-OS kernel stuff, in a production environment has to have access to the high-resolution timer.

There is a - it's called the RDTSC instruction on the Intel, Read Time-Stamp Counter. And, for example, I use it in SQRL as just one more source of entropy, just another piece of data which is unpredictable, and no external attacker, no one remote can have any idea how many cycles at whatever, 3.some gigahertz, my particular instance or any of the SQRL client users' instance of their processor has executed since it was booted. It'll never be the same twice. It'll never be the same even if you tried to make it the same, just due to all the other things going on, things like where in its rotation the hard disk was or where the hard disk's read head was as it was seeking. I mean, like there's so much uncertainty in the system that having clock cycle level granularity is exquisitely fine.

Well, that's useful as one of many sources of entropy. But it's that level of precision timing, at sub-billionth of a second that the clock is running at, it's having access to that that allows these attacks to occur, that allows software to discern with that exquisite granularity whether some data came from the local immediate access cache or even has to be pulled out of main memory, which is much slower. And so if the instruction stalls in trying to fetch something, and then immediately after executing that instruction it reads the Time-Stamp Counter again, it's able to discern that it stalled. It's able to sense that time passed while it was not executing while the processor went out to get the data that was needed by the immediately preceding instruction.

So there's an argument to be made that a solution to this would be to fuzz the resolution of the timestamp. That is, it's not clear that there's any useful real-world need for it. Maybe the OS needs it, but it's not even clear to me that it needs it at that level of granularity. It's there because it's in the chip. It's been there for a long time, since one of the early Pentiums, I remember, that appeared. SpinRite has used it. When I was doing the sector re-interleaving in SpinRite 1, the only reliable source of time that was high granularity was this timestamp that has been there from the dawn of Intel. So it's still there. And there are many other so-called "performance counters" which Intel makes available in their silicon to programs that want to execute it.

Now, what's interesting is the Read Time-Stamp Counter can be marked as privileged so that software cannot access it. On the other hand, unfortunately, there's no way to take it away from software now that presumes its presence. However, in a VM environment where you have virtualized hardware, the timestamp could be and is typically virtualized, which would allow it to be fuzzed. It would allow it to be made less accurate. And all you had to do was just add some uncertainty to it so that software cannot actually determine with the required level of certainty how long an instruction took to execute. And then this whole class of recently discovered problems disappears because the only way they're able to operate is to sense with absolute precision how long something took.

And again, it's not that the instruction didn't execute. It's just it took a tiny, tiny bit longer in order for the system to fetch data out of main memory, in the case of Meltdown. So that's one potentially global fix to all of these problems. But this PCID is arguably the proper way to solve the problem of one process being able to sense the contents of a cache from another process. Again, five years ago, 10 years ago, it was like, eh, that wasn't a problem. Today it's a showstopper which everybody is reacting to.

So I think that the short-term sort of kneejerk reaction of flushing the cache whenever you go back out of the OS to the user, or maybe even cross-user, is going to be reduced once we get access and once operating systems are actively supporting this PCID flag on the caches, which solves that. But that's only one of the several problems.

The other whole class of problem is Speculation. As I started to say before, Speculation is the notion of executing when you've got execution capability that is not in use. So, for example, we've got multiple cores and hyperthreading technology on a state-of-the-art chip. Whether it's Intel, AMD, ARM doesn't matter. Say that you have - and I'll just use some simple algebraic equations to give an example. You have A plus B equals C where imagine that these are registers, and D plus E equals F. In those two equations, the outcome C of the A plus B equals C is not used by the next computation, D plus E equals F.

And the way processor engineers discuss this, they say there's no dependency of the outcome of the first by the second. So if those two simple math operations occur next to each other on a modern processor, it will notice that. It's smart enough, it's been engineered by state-of-the-art processor designers to see that it - and imagine that B, you know, A plus B equals C, imagine that B is off chip at the moment. That is, it's in main memory. And so it's going to take a while for it to fetch the value of B in order to compute A plus B equals C. It looks ahead, and it goes, oh, but look, D plus E equals F, and I've got all of that data in my cache. And I don't need to wait to find out what C is because nobody cares in this next math operation.

So the processor will do what's known as "out-of-order execution." It will leap ahead. It'll start asking for B to be hauled in from main memory. Meanwhile, it'll go ahead. And as long as there is no dependency, it will move forward, saying, well, nobody seems to be asking for C. So it'll just keep on executing as far as it can and as far as the architecture will allow it to get ahead, moving ahead. So out-of-order execution is one form of this. The other is branch prediction. If all code could only run in a straight line, it would be very limited in what it could do. Not completely limited. There are many instances where, for example, high performance graphics processing does a whole bunch of stuff all at once that doesn't involve lots of branching. So you can get work done without branching, but ultimately you need to make some decisions.

So here again we have a situation where somebody, a sufficiently motivated engineer could design, and they have designed, a system where the code is coming along, and a conditional branch is encountered, meaning that the decision whether to take the left fork or the right fork in the code is dependent upon the value of something which is not yet known, like that C. Imagine that it says, well, if C equals zero, then go left. And if C is not equal to zero, go right. But we're still waiting to get that value of B in from main memory to add A to it to find out what C is before we know whether C is going to be zero or not.

So what we could do is what's called a "processor stall." We could just say, shoot, we don't know C because we haven't got B yet. But again, in this interest of squeezing with any cost of effort the absolute maximum technology out possible, the engineers say, well, instead of having to decide which path to take, let's take them both. And it's like, what? Yes. If we've got, again, excess computing resource, let's just go down both paths, the idea being...

Leo: Oh, so it's not like branch prediction. They get both.

Steve: Correct.

Leo: In branch prediction you pick one, and then there's a penalty if you pick the wrong one.

Steve: Yes. Now, there is branch prediction, too. I mean, so there's that also. But there's also, instead of taking - and so that's like taking the road more traveled rather than road less traveled. But you can also just say let's go down both. And again...

Leo: Amazing.

Steve: ...the complexity, yes, it's just, like, unbelievable how complicated the processors are today because they have to keep track of everything they're doing down both paths and make a checkpoint of, like, okay, we're going to go down both paths, but we have to make sure we don't screw anything up because we have to be able to unwind any changes which end up being made by the path we don't end up taking, once we finally get the result of the earlier math operation to tell us which branch we should have taken.

And so it's like mindboggling. But this is what state-of-the-art processors are doing. And this is all collectively called "Speculation." You mentioned branch prediction, the idea being - and I referred to this, I guess when we were talking about this on Saturday, Leo, the idea being that when you are - in general caching works because what has been known is that code generally runs in patterns, and data which is in use tends to get used again. Well, the instructions that are being executed also run in patterns.

And so branch prediction is another form of caching. It notes that, gee, the code took this branch last time and the time before and the time before that. And so, if it can't, if it doesn't have enough resources to go down both paths because other aspects are using the available resources, then it will settle for guessing the path likely taken based on what's happened recently.

And so it'll go down the path that has been taken more recently, again still making a checkpoint in case it has guessed wrong because that is eventually going to happen; and it will go as far as it can based on how deep the Speculation resources are in the chip. And when the result of C is finally known, it checks, and it says, oh, yeah, we're going down there again. Then it commits, or in processor design terms it retires, it's called "retiring an instruction," meaning finally saying, okay, fine, we're done with you. It will retire and commit all of those things it was kind of holding its breath about, waiting for the answer from the earlier question.

So what we have is we have, in all of these things, we have caching. We have short-term memory of which branch, which fork in the road is likely to be taken. And we have essentially a processor which is maintaining a huge amount of state, a huge amount of information of recent activity. That is, it doesn't know what happened last week. It only knows about what happened milliseconds ago as code is zipping around and zooming around in there. But that's enough to give it a huge performance advantage if it's able to guess right, or if it's able to speculate down multiple paths and hold its breath, hold all of these possible outcomes, sort of like the multiverse, where it's like, oh, if a decision is being made, well, that causes the universe to split down both paths, one where it happened and one where it didn't.

Well, this is happening inside of our chips now. And where decisions are being made later which essentially affect the future, it's able to hold all outcomes and then commit to one that finally occurred, essentially leapfrog, jump forward, as long as it has sufficient resources. But the secret here is the problem because doing this leaves footprints. Doing this alters the state of the processor. And today those things, those alterations have global effect. That is, there is no flushing of all of this state when you change processes or change tasks because nobody has really seen the need for it.

Now, there are instructions that have been around for a long time in the Intel instruction set which have not been much used. There's one called LFENCE which explicitly allows speculation to be blocked. And there are various reasons why you want to sort of like force the chip into a known state because, well, for multitasking and fine-tuned optimizations, there are places where you need to know what's going on, and so you'll sort of want to rein in this speculation. So this LFENCE instruction can be used at critical branch points to block speculation and prevent these kinds of speculation-based exploits.

And in all of the reading that I did over the last few days, I ran across somebody who - and I think it was from Intel, so I was a little skeptical because I remembered that - I sensed that they had a little bit of a bias in this. But the sense was that in the Linux kernel there were only a few places where there was really this problem where this LFENCE instruction could be inserted. There's been mention of compiler changes which would cause compilers to start inserting these.

Now, you want to do them sparingly because it kills your performance. That is, we are relying on this level of speculation in order to have the performance our systems have today. And so if we start blocking speculative execution at every branch, suddenly something we're relying upon is gone, and we'll feel the performance hit. And we don't need to block it everywhere. We just need to determine where it matters. So that's why my feeling is we're at the beginning today, like this week we're at the beginning of a somewhat lengthy process of dealing with the consequences of this revelation, which is that this speculation can be - and, I mean, the proof of concepts have demonstrated that it is possible for code running in a userland to read about 2K bytes per second from the kernel.

So it's not super high-bandwidth, but it has been demonstrated. And so that's why everybody for the last few months has been running around with their hair on fire, quietly trying to get mitigations in place because the second this became known the hackers were going to try to take advantage of it.

Intel has said that there will be forthcoming updates to the firmware in the chips. They are going to be introducing some additional instructions to give the low-level coders, the kernel developers, more control over speculation in order to mitigate these. Yesterday, the Intel CEO said that patches will come to 90% plus, that is, more than 90% of Intel's chips in the next week, and the rest of those by the end of January.

So again, typically, end users are applying OS patches and software patches, but not chipset patches. That needs to go through the chipset supplier. So I expect that we'll see Apple patching their stuff at the chip level. And I imagine people who have laptops and desktops with chipsets which are still being maintained from suppliers who are being responsible and still keeping the BIOS and the chip firmware up to date, I mean, who knows how long it's going to take to get through the pipeline. Weeks, months, probably.

But ultimately we should see a revamping of the firmware to create new instructions to provide additional granularity to speculation where, as we grow as an industry to understand the consequences of this, this begins to filter out, and we basically roll back some of the freewheeling freedom that we have enjoyed for the last couple decades, not worrying about and not really recognizing the consequence of chips maintaining so much state in order to give us the kind of performance we've gotten accustomed to, and now suddenly recognizing, and it going public, that there is a way to sense that state which was put in place by other tasks running in the same system, and that creating an information leak. That's what it comes down to.

So basically this is the big coverage of this now. And I'm sure that in coming weeks and months we'll be discussing the consequences, the actual consequences in terms of performance and mitigations, and where is Android, where is iOS, where is Mac, where is Windows, where is Linux, and where are all the various chipsets scattered through all of our machines. It is somewhat annoying that we can no longer rely upon the chipsets we have, and that they're going to have to be updated.

Note that the mitigations - and this is what I was trying to explain on Saturday, more than last Tuesday, because this wasn't that clear. What'll happen is the OS kernels will use a blunt mitigation which will cost us in performance, but protect us, much like with Meltdown. But then when something like the PCID, the Process Context ID, is available in the hardware, then the OS will be able to back out of that blunt kneejerk reaction to get back most of the performance by leveraging a feature of the processor that wasn't being leveraged.

Well, that works for the PCID. What is going to happen with the firmware updates in our chips is new instructions which operating systems will then become aware of, which will allow us to stay secure while recovering performance that we have lost. So what this means is people who don't or are unable to update their processor's instruction sets with these additional instructions that allow fine-grained mitigation, they will see a performance hit and never be able to get the performance back. Essentially it's like we were running on borrowed performance at the cost of security without knowing it.

Leo: They were saying that, for instance, Haswell and older chips are not going to be able to.

Steve: I think it's true.

Leo: But that doesn't even seem that old.

Steve: That's not that old, yes.

Leo: It's a couple of years old; right?

Steve: Yes, that is, exactly. So, okay. Microsoft has a power shell script which I don't know what the story is there. I made it the bit.ly link for this week. And then the zip file from Microsoft disappeared. Then it came back. Then it's gone again. I tried to use it last night when I found it, and it didn't work. So they've been having all kinds of trouble.

There is a very small and clean utility on GitHub. It's called SpecuCheck, S-P-E-C-U-C-H-E-C-K. I've been watching it. I've been following its development because I just haven't had time, I haven't had time to build it myself. But I have absolutely no reason to mistrust this person. If you do SpecuCheck/releases, Leo, add a slash, a forward slash "releases" to that URL. You will find releases, as of the time of this podcast, 1.0.4, where he has the EXE. So anybody who is skeptical is welcome to compile their own executable from the source - it's there both in ZIP and TAR format - and the executable for people who can't compile their own. It's 80K. That's what I recommend people use. It looks very clean. He's improved the documentation on his source.

What this does is probe the chip you're running for all of these specular execution mitigations and the OS that you're on, that is, it is a Windows EXE. So for Windows users, this is what I think you want to use is SpecuCheck. It will allow you to verify both before and after a firmware update, a microcode update to your processor, whether or not it supports these features, and whether or not the OS is basically in blunt mode, which is costing you performance, or recognizes that your processor has the required features, if it does, and whether the processor is aware of that and is using them.

So this is the best we have now. Again, I'm sure this is not the last we'll be talking about this. I mean, this is, as I said, this is truly a major event for the industry that impacts security and costs us performance because it is because these features give us performance that we've had the performance we have. But it's the nature of the way we've achieved that performance that inherently - which has been global for the chip, which inherently costs us security. We are going to immediately get the security back at a short-term cost to performance that we then may be able to recover in the future. Certainly with any chips moving forward.

Leo: Whenever I run it, it just crashes out. It opens a window, and it goes away. So I'm not...

Steve: Oh, no, yes. You need to, I'm sorry, unfortunately this is not a Windows app. So you need to open a...

Leo: Oh, a command line, okay.

Steve: ...command line and then manually run it from the command line, and it'll dump out a bunch of texts. I would love to write one of my little Windows apps for this, but I'm going to keep on SQRL and assume that others are going to write a nice Windows tool, because it's simple to do, in order to show people what's going on with their chip and their system.

Leo: So I just ran it. And, yes, it says mitigations for CVE-5754, which you can't really read this, which is rogue data cache load. Yes for shadowing, but no for user pages marked global. This is with all the Microsoft patches on here. And a lot of no's for 5715, which is branch target injection. In fact, the only...

Steve: Yup, in fact that's what we're going to be getting some firmware updates, some microcode updates for our processors.

Leo: Well, actually no's aren't necessarily bad.

Steve: Exactly. As long as there's coverage elsewhere.

Leo: Yeah. Kernel shadowing [crosstalk].

Steve: And again, unfortunately this is a bit on the techie side. I'm hoping that somebody will come up with something that makes it clear.

Leo: Yeah, yeah. But, yeah, at least you can do this.

Steve: And if not, I'll steal a day and write one that makes it clear.

Leo: Yeah, branch protection mitigations enabled, no. Disabled due to system policy registry, no. So that's that key you were talking about.

Steve: Yup.

Leo: Disabled to lack of microcode update, yes. So I guess I'm waiting for a microcode update.

Steve: Well, we all are, Leo. We're all waiting. That's going to be, you know, again, the problem is Intel having it available doesn't mean that it is applicable, unfortunately, because it's got to come through our platform vendor, Lenovo, for example, or whatever motherboard we're using.

Leo: Yeah, it's not clear what you want to see on these results.

Steve: No. It's unfortunate that it's not more clear. It's all we've got right now. But again, I think a week or two from now it'll be made more clear. And if it's not, I'll steal a day and write one that is clear.

Leo: Yeah. Wow. Okay. That's good. I want one for the Mac now. I guess I could look at his code. Looks like it's C code, so it's probably not...

Steve: Oh, it is, it is. It's just pulling information from a couple of the model-specific registers in order to figure out what's going on. And again, anybody who doesn't trust it can write their own. And again, I'm sure we'll see several tools out in no time to make this clear.

Leo: I'm getting exactly the same results as his screenshot, interestingly enough. So I don't - I hope he explains it.

Steve: Yup.

Leo: Yeah, all right, okay. Interesting. I wonder if this would work - does it require the Windows kernel, or is it just calling - I think it's, yeah, I think it's probably Windows only. Okay, good to know. Good tool. As usual, good information. And what it really does is it's kind of like Apple's battery problem. It kind of requires everybody with anything Haswell and older to upgrade.

Steve: Yes.

Leo: Geez.

Steve: Yes.

Leo: That's terrible.

Steve: Or suffer a probably noticeable performance hit. End users may not get hit. That's the good news. But cloud platforms, hopefully they're on more modern hardware, and/or they have the clout to get the firmware and the microcode update that they need. So anyway, we'll be riding this one for some time to come, I'm sure.

Leo: Geez. So Haswell's from 2013. So it's fourth-generation. We're up to eighth. So, yeah, if you bought a computer in the last, you know, more than four years old, you're kind of out of luck.

Steve: Yeah.

Leo: Holy cow.

Steve: You can be secure, but you cannot be fast.

Leo: Right. Wow. Well, there you go. You've heard it here. Thank you, Steve, as always. Steve Gibson, you can find him on the Twitter, and I know there's a lot of traffic on the Twitter about all of this, @SGgrc. You can also DM him. He accepts DMs from anybody, unlimited length there, if you have something to say or a question. You could also go to his website, GRC.com. Leave questions at GRC.com/feedback. While you're there, pick up SpinRite. You must. You have to. If you have a hard drive, you need SpinRite, world's best hard drive recovery and maintenance utility even for SSDs.

You can also check out all the work he's doing on SQRL, or the passwords. I use your passwords all the time. That 64-character password generator is really great. On and on and on. All that stuff is free, including your copy of Security Now!. He offers 64Kb MP3 audio plus very nice transcripts written by Elaine Farris at GRC.com. We have audio and video at our website, TWiT.tv/sn. And you can of course subscribe. You'll get exactly the same audio if you subscribe on any podcast program, and just make sure you do because you want to get every episode as soon as it's available, which usually is later on, late in the day Pacific time, Tuesday.

We record, if you want to watch it live, Tuesdays at 1:30 Pacific, 4:30 Eastern, 21:30 UTC. And if you do watch live, by all means join the chatroom. They are awesome, and it's a great way to get kind of some additional information, some back channel, and some socialization during the show: irc.twit.tv. If you want to be in the studio, you can do that, as well. But please don't surprise us. We don't like surprises. Just email tickets@twit.tv, even if it's just an hour ahead of time, just to let us know you're on your way. We'll make sure we have some space for you: tickets@twit.tv.

Thank you, Steve Gibson. Wow. Wow, wow, wow. Great show once again. And we will see you next week on Security Now!.

Steve: Thanks, Leo.

☒ Jun to top

☒ Jun to