The InSpectre

Description: This week we discuss more trouble with Intel's AMT, what Skype's use of Signal really means, the UK's data protection legislation giving researchers a bit of relief, the continuing winding down of HTTP, "progress" on the development of Meltdown attacks, Google successfully tackling the hardest to fix Spectre concern with a Return Trampoline, some closing-the-loop feedback with our terrific listeners, and the evolving landscape of Meltdown and Spectre - including Steve's just completed "InSpectre" test and explanation utility.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-646.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-646-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. This is a hot episode, I think perhaps will be one of the most listened-to Security Now! episodes of the year, maybe of the decade, I don't know, because we're going to give you, not only the latest on Spectre and Meltdown, but a program you can use on Windows machines that will help you know whether you're safe, and what performance hit you're going to get. Security Now! is next.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 646, recorded Tuesday, January 16th, 2018: The InSpectre.

It's time for Security Now!, the show where we come to you via the Internet to protect you from the Internet. I don't know how that works. But that's Steve Gibson. He's figured it all out. It's like Inception. Steve is the man in charge at GRC.com and our security hero. And you're going to really be a hero this week, Steve.

Steve Gibson: Well, yes. After last week's podcast I went back to work on SQRL that evening, and I worked on it Wednesday. But Wednesday night I was just thinking, you know, this problem with Meltdown and Spectre is not going to be short-lived. It's not going to go away. It's going to be an ongoing issue. At that time the only solution we had, there was a Linux script, and there was the PowerShell thing. And I had said on the show, "No, I'm not going to do this; I'm not going to do this; somebody else is going to do this." And Thursday morning I thought, I've got to do this.

So Thursday, Friday, Saturday, and Sunday - and yesterday, actually - I first had to get fully read in on what was going on because my solution, of course, differs from any of the other ones in several respects. One is on the back end it deals with the hardware, so

it's going down and probing the processor itself, not just relying on some newly instantiated calls from Windows. And then on the front end, very much as I did with the DNS Benchmark, it interprets the complex interaction of all of these different factors into English and writes a description about the status of your system for the user, for their own particular instance.

The press has just begun to pick up on it. There have been a couple articles written. I shot a note out, it was late last night when I finally got this thing done, and had about 20,000 downloads. And it's now the first three links in Google's search for the term "InSpectre" spelled with the "re" backwards, of course. So I-N-S-P-E-C-T-R-E is just released - paint is still dripping off the edges of it - freeware which really brings, I think, some clarity to a confusing situation because there's whether or not Windows is doing what, what your processor is doing, whether you need firmware updated or not.

So basically it explains everything and then gives you sort of some bullet points of what you might do. And whereas Microsoft's script, of course, isn't rough on them, I hold their feet to the fire a little bit because Windows 7 is still being updated for security problems, and Microsoft has shown no indication, at least yet, that they're going to bring the mitigations that they've brought to the Fall Creators Update to 7. And in fact, even the, what is it, 1703 build, the pre-Fall Creators Update, it doesn't give you safety. So there's some concern that Microsoft may be tempted to leverage this to push people where they want us all to go.

Anyway, so the title of today's show is "The InSpectre," so we'll talk about that a little bit more. But also sort of where the industry has gone in the last week. And I listened to you talking about this at the top of MacBreak Weekly. I now, well, there's some interesting facts about firmware and microcode for our processors because that ends up being the thing that's left hanging out, which is going to affect many people. So anyway, lots to talk about. We're going to continue. We've got of course news for the week. We're going to talk about some new trouble with Intel's AMT. Once again it's in the news, and not in a good way.

What does Skype's use of Signal, which is now in pre-release form, but Skype essentially has announced, Microsoft's Skype will be using the Signal protocol for point-to-point encryption of conversations. What does that actually mean? The U.K.'s data protection legislation has given, surprisingly, researchers a bit of relief. Also some interesting moves in the continuing winding down of HTTP, that is, without the "S" on the end, insecure HTTP, it's really becoming something that no one can practically use, certainly moving forward. Also there was news of, and I have this in quotes, "progress" on the development of Meltdown attacks. We knew this was inevitable. People would start working on these vulnerabilities to turn them into practical attacks. And there's some early results from that.

Also, Google has successfully tackled one of the hardest to fix Spectre concerns with something known as a "return trampoline." And then we've got some closing-the-loop feedback with our listeners, and then we're going to sort of do a catch-up on where the world stands with these two critical vulnerabilities.

Leo: I'm willing to be this is going to be the most listened-to Security Now! in some time. And, I mean, not only because people will be interested in InSpectre, but also because there's still so much we don't know about what's going on and how it's being fixed. And one of the things I said on MacBreak Weekly is I wish there was something like InSpectre on the Macintosh. We really don't know what Apple has

done. We can only take their word for it. There's no way to validate what's going on. We don't really understand it. And iOS, same thing.

Steve: So my utility does run on Linux under WINE, and on the Mac under WINE.

Leo: But that's not going to help because...

Steve: Well, but it does because it's able to probe the hardware.

Leo: Well, if the hardware is mitigated, right. But we won't know if the OS is mitigated.

Steve: No, no. And in fact, probably, what would be interesting would be to develop actual use tests where, rather than just - so, for example - I'm getting ahead of myself.

Leo: We'll talk about it in a sec. All right, Steve. I am going to, while you're talking right now, I have InSpectre running on my Lenovo, so if you want a screenshot from that I'll show you.

Steve: You can get a sense for what it does just by scrolling down, and you can see...

Leo: I love this.

Steve: Yes.

Leo: Love this. But I don't want to - I'm not going to steal your thunder. I'll let you talk about it. But this is really - this is what needed to be written. I'm really glad you did this, yeah.

Steve: Yes. Okay. So security news first. The press went a little crazy over news of another Intel AMT configuration flaw. And so I wanted to sort of bring some sanity to it. I mean, it's not good, but it's sort of a bunch of problems coming together. It turns out that a lot of corporate laptops which support remote management have AMT, which is, as we've often talked, is this Intel management technology that allows corporate management of machines through the network, which upsets some people. It's problematical and so forth.

Well, it turns out that it has its own BIOS. And because it's sort of obscure, and it's not the normal password that the user types in, and it's not normally accessible, even though it has a password, it's almost never changed - you know, you can't make this stuff up - from "admin." Which is the default password for Intel's AMT BIOS access.

So here's the so-called "attack" which, again, it's not good, but it requires physical

access. During a reboot of the laptop, and I've read all the various scenarios where the owner is distracted, or there's a two-man team who asks one person a question, and it's like, okay, fine. So movie plots. But the idea is that, when the laptop is rebooting, Ctrl-P can be pressed at the right moment, or maybe just "P," I don't think you even need Ctrl, just "P" in order to intercept the normal boot sequence and get to the AMT BIOS extension.

It is password protected. But in all the tests that people have run, nobody ever bothers to change that password because, again, it's sort of like many - there may be corporate IT who don't even know you can get in that way. So you type "admin" in order to get to the AMT BIOS, where you can change or set the password, enable network access, and change a setting for the user's opt-in to "None," meaning that there needs to be no further physical access to opt into AMT through the keyboard. It's just all in the background. And at that point the hacker hits F10 or saves the BIOS, whatever they do to save settings, and from this point on that laptop is now vulnerable to local network access. You can't do it remotely. It's got to be on the same network segment. But that's a low bar to get under because WiFi qualifies and so forth.

So anyway, that's the crux of this is that, unfortunately, this very powerful hands-free kind of remote access, or at least network access, local network access by default has a password which anyone who has brief physical access to the laptop can change or use in order to enable a set of features that then allow hands-off access. Oh, and I forgot that I sort of buried the lede here. This bypasses, this use of this bypasses the user's password, BIOS password, the Trusted Platform Module protection, and BitLocker as part of this because you want corporate access to the machine in a way that makes sense to it, that it's able to see.

So all, I mean, all of the interactive protections go away once this is done. So it's just something to be aware of. I don't think it's the end of the world. One would argue, you know, how is it possible that there could be a default password on something that allows you to do this, which is "admin," but that's what it is. And in Intel's defense, in their best practices documentation they say, of course, change the password. But nobody does that.

Leo: Nobody even knows it exists.

Steve: Exactly. You can't change the password for something you don't know exists, and nobody hits "P" when they're booting their laptop. So the recommendations for end-users would be obviously don't let bad guys distract you while the other member of the bad guy team does this to you. If you have a laptop that's a corporate laptop, that probably has these features, you might want to say hey to your IT service desk. Hey, you know, what do I need to do about this? If you're an individual running your own device, that is, responsible for it yourself, then by all means change the AMT password away from admin to something strong, even if you don't plan using AMT.

And that's the problem is these are systems where AMT is present, but has not been deployed, typically. So it's just sitting there available for the first person to come along and grab it to do so. So you should grab it. If there's an option to disable it, disable it, if you don't need it. But by all means give it a strong password so that you're not, you know, there's no chance that someone could come along and set that up without you knowing it and then get in behind your back.

And anyway, so that's the story. It's not what it sounded like from the headlines, of

course, that this is the end of the world, remote access, like some new major attack. But with any luck the major attacks on AMT are behind us now, and now we're just dealing with things that require physical access.

I thought it was interesting that Skype, Microsoft and Skype have announced a partnership with Signal. We're very bullish about Signal. We did a podcast on it. We've talked about it a number of times. I took a deep dive and completely read the spec. And our listeners will remember that, as I'm beginning to read the spec, I'm thinking, oh, my lord, has this thing been overdesigned, because I'm always about keeping it simple. But this thing is just nuts.

But as I continued to read and understood what much bigger than I expected set of problems it was solving, I realized why Moxie and company had done what they had done. And I just, you know, they really, really thought this thing through well. They solved very difficult-to-solve problems in a way that, like, problems for which there's no perfect solution, but they did a beautiful set of compromises with no compromise in security. So I thought, okay, this is interesting that Skype is going to get this.

What we need to remember, however, because there's a lot of hand waving about how Signal is an open source protocol, an open protocol, which we know makes a lot of sense. It allows someone like me to read the protocol and say, okay, I don't see any problems here. And in fact I see a lot of good, which can't happen - what is that, the BitTorrent protocol, still undocumented. No transparency.

Leo: BitTorrent Sync, yeah.

Steve: Yes. Everyone wants to know what I think. I don't think anything because, you know, except their name and reputation. But there's no documentation. Their press guy was hounding me for a while to talk about it, and I said, I would love to, but I need something to talk about. We do technology here, not just regurgitate press releases that somebody else wrote. So I just keep saying, okay, it'd be nice to say something. I could about Signal because it's fully laid out.

But it's also important to recognize that Skype is not open source, and that given a fully proper implementation of Signal with Skype, what we get is the strength that Signal offers for preventing interception, and some of the extra features that Signal brings, which are those clever bits about, like, how do you handle somebody wanting to send you a message when you're not there, and they need a token, so your client has posted a bunch of available ones on a common server. I mean, Signal did a lot of things right.

But we don't know, for example, that Skype itself isn't able to respond to a court order. It couldn't probably through Signal if it's a faithful implementation of the Signal protocol. But remember that the Signal protocol only protects the data in transit, not at either end after it emerges from Signal or before it goes in for that encryption. So because it's all packaged into a Skype client, there could be any kind of other means for secreting the communications out.

So it's certainly nice, but I sort of wanted to temper any concept that, oh, this means that Skype itself is absolutely secure, and we can send state secrets back and forth. No. It's sort of good that it is, but in my opinion in no way does this even make it more safe because, while it's nice to have Signal connecting the endpoints, as we know, security tends to attack the weakest link. And the weakest link in the chain is absolutely no longer the end-to-end encryption, but it's everything else. And there's plenty of everything else

to attack in a scenario like this, malware on either endpoint or undocumented features that allow Skype to respond to court orders - which, if not yet part of the law, is frighteningly threatening to become part of the law.

So anyway, it's nice, but I'm not sure that it really actually changes the security profile. I would be far more inclined if I was really concerned about security to use a fully open source solution, that is, where even the client itself is open source. And of course, if you really want to go all the way, you use source code that has been fully vetted, and you build it in a clean room environment so you know something about the client and where it came from. So that's certainly not what Skype is. So anyway, it's not clear what a lot we're getting from that, but at least end-to-end encryption will be strong, thanks to Signal.

And in a surprising move in the U.K., which has been probably the loudest saber-rattler about their privacy laws and generating lots of, if nothing else at this point, worrisome rhetoric, there was a bit of nice news that I hope portends a trend. How many times on the podcast have we noted that it was by allowing security researchers to poke at things that significant problems were found, even in situations where the company whose software was poked at was absolutely unhappy that that happened. Yet we, at this point still, we are not stepping on security researchers imagining that the world is better if nobody challenges the assumption of security.

So what happened in the U.K. is that their rather onerous-looking data protection legislation was amended to a little bit protect security researchers. There was a section in it where the language was really broad. It said that it was a crime to deanonymize data, period. That is, if data had been anonymized, you couldn't deanonymize it. And a bunch of researchers rightly said, whoa, whoa, whoa, wait a minute, you're saying that it's not possible for us to then verify the anonymization of data? Because the only way we can verify it, the only way we can test it and poke at it is to try. And in trying, we would become criminals. So that clause got backed out to read, quote: "intentionally or recklessly" re-identifying individuals from anonymized or pseudonymized, there we go, pseudonymized data...

Leo: Don't know if that's right, either, but...

Steve: Pseudonymized, yeah, I think "pseudonymous" is the term.

Leo: Okay, pseudonymous, all right.

Steve: So pseudonymized, yeah. There's no good way to - it's just like a crazy word. But anyway, so I'm just - I'm so hoping that we don't end up in a world where researchers fear for becoming criminals. And if it was going to happen, it would be in the U.K. So this represents at least some movement in an encouraging direction, which I'm glad for.

Okay. Winding down of HTTP. I think it was Simon Zerafa, our friend of the podcast, who sent this to me. And I was curious because this looked like a big deal. So I dug into it, and it is. So we have to understand first this notion of a secure context. There's a formal definition for the term "secure context" in the world of browser design.

And so formally defined, a secure context is a window or a worker, which is a term for an execution thread, a JavaScript worker, for which there is reasonable confidence that the

content has been delivered securely, meaning over HTTPS with TLS, and for which the potential for communication with other contexts that are not secure is limited. So you obtain this data, this window or worker thread, script, securely; and it's walled off so that it is protected. Many web APIs and features are currently only accessible in a secure context. The primary goal of secure context is to prevent man-in-the-middle attackers from accessing powerful APIs that could further compromise the victim of an attack.

So, okay. That all makes sense. We talk about this sort of thing all the time. So in fleshing this out in the Mozilla definition of this, they say: "Why should some features be restricted? Some APIs on the web are very powerful." And as we know, getting more powerful all the time. We're really seeing the browser turning into an application delivery platform. Where we use to rely on plugins, those plugin things are now moving themselves into the browser, which in general is a good idea because it means that the developers and users can just count on a unified experience independent of which specific browser they use, and that on the other end services are able to presume that those features are present. Back in the day you would get, oh, you have to load the Flash plugin in order to proceed. And we know that's advice that you want to be very wary of for quite some time.

So the problem is that, as these capabilities increase in strength, attackers can leverage them to invade a user's privacy, obtain low-level access to a user's computer, even get access to user credentials. Although it's not yet widespread, Chrome supports the Bluetooth API. So a browser will be able to access by radio Bluetooth devices within its reach. So, yeah, we really don't want bad guys to be able to put ads on sites that then run code to do an inventory of all the Bluetooth devices that you have around you. So these things are both powerful, but with them comes a real security risk.

So Microsoft's blog posting was sort of putting a toe in the water titled "Secure Contexts Everywhere." And that's sort of a play on the HTTPS Everywhere idea. Well, and essentially that's what it means, the idea being that what's being proposed is that, thanks to this world where we have Let's Encrypt and we have reduced the objections to establishing a secure connection between endpoints, where you could argue there is really no strong reason now for a site not being HTTPS because it costs nothing. And once you set it up, it just manages itself. And given that attacks are so problematical, and nonencrypted connections are hard to have secure - you can't have a session context cookie over HTTP. It can be intercepted, and your context can be stolen a la Firesheep years ago.

So what Mozilla is proposing is a more formal statement that anything new moving forward will require a secure context, that is, as new features are introduced, unless there's a clear strong reason where it cannot be instantiated into the specification into the World Wide Web Consortium HTML5 spec, if there isn't a very clear reason not for it to require a secure context, it will. Which says that, moving forward, any significant new features are going to be able to presume security. That presumption gives them more flexibility and more power.

Now, it also doesn't necessarily hurt somebody who, for whatever reason, adamantly refuses to, like, their server to be doing HTTPS. But it does mean that with that refusal comes a significant reduction in features. So that is to say, if somebody were to set up a website that was not HTTPS for whatever reason, they could certainly serve web pages, but there's all kinds of other things that they very well may want to do which browsers moving forward - and, I mean, which browsers today, there are features which browsers today will already not do unless over a secure context. And moving forward, all new features, if this proposal is adopted, will require security.

So probably what this has the effect of doing, unless you're running just a very barebones, deliver text, low multimedia, no extra features, you really can't log someone in because they have to have a session cookie in order to maintain their login state. So I think what this says is another nail in the coffin of HTTP, which thanks to the fact that we now have the ability to easily deliver and administer very low cost, essentially zero cost domain validation certs, and have them dynamically maintained, it makes sense that this ends up happening.

And we're already seeing that servers are being delivered with this capability built in. That'll be the next, probably the final straw is that all of the major servers on the 'Net will offer that feature. If you do not want a certificate stronger than domain validation, just set this switch, and your server will be set up with a certificate, and you don't have to worry about it. At that point, it's hard to imagine, you'd actually have to work, probably, in order to get a non-TLS connection. And it's certainly foreseeable that probably five, 10 years in the future HTTP will just be gone. It will no longer be - it'll be like, oh, yeah, remember when those crazy web browsers just sent everything in ASCII text over the wire, and everyone could look at it.

Leo: Crazy.

Steve: Crazy. Two tweets came to mind that sort of fall into the rest of what we'll be talking about here. A researcher, Raphael Carvalho, tweeted an expletive, the F-word, comma, "I can barely believe that I was able to read non-cached data from other process efficiently. Removed iteration and issued flush on secret." And then he gives somebody thanks for helping him technically, and also Alex Ionescu, I guess that's how you pronounce his name, he was the researcher who did the Linux script early last week and has been very involved in the Spectre and Meltdown stuff. And then he says thanks to these guys for all the tips. "Not releasing it, or somebody could definitely set the world on fire with this." And then he says "#meltdown."

And then Alex tweeted: "I can finally efficiently" - and he says "(fast)" - "and reliably (no errors) read paged pool/non-L1 data." And then he had, instead of "Mimikatz," he twisted the terms around. He called it "MeltiKatz" or "MimiDown" in order to take the first and second halves of "melt" and "down." He says: "I'll sit on this a few weeks before setting the world on fire and watching it burn. Or probably somebody will do it first." So these are indicative of what we would now expect to see start happening. That is, this is just too exciting for both researchers and those with malicious intent to go and pursue. I mean, there's so much data about how this is done, the Spectre and Meltdown fronts.

So far this is all just Meltdown, what these guys are talking about. Spectre is probably going to take some more work. And what's interesting, though, is that Spectre, the second variant of Spectre, is the one which is the most difficult to mitigate because it requires firmware updates. It requires that some features not in the earlier versions of our chips be present in order to mitigate this. Or it requires that code be recompiled.

Anyway, that leads me into the next and final subject before we move into the next section. And that is, Google has come up with a very efficient system for dealing with the hardest to fix, that is, the one that in unmodified code requires microcode for existing code. They call it "Retpoline," a contraction of, or I guess a mangling of, "Return Trampoline." The return as in the return from an end of a subroutine, and trampoline is the process, as the name sounds like, of bouncing your code through something else. So it's called Retpoline, as in Return Trampoline, and we're probably going to be hearing about that in the future. They've released the technology and the technique formally to

the public domain and explicitly, at the end of their posting, explaining exactly how to do this, releasing all claim to this technology for the benefit of all.

This was designed by one of the Google engineers who was worried about mitigating the Spectre attack for Google's own processes and code and servers. This is the one of the two that triggers the speculative execution through branch target injection. It's the CVE ending in 5715. Google's gotten a lot of attention and press about this, and deservedly so. I read through it. It is very clever. It does require that code be recompiled.

So this is the kind of thing I heard you guys talking about on MacBreak Weekly before, Leo, where Apple and Microsoft are recompiling the browsers in order to deal with these problems. And I think what we're going to see, Microsoft also just announced an update to their Visual C compiler to add a /Q Spectre switch to, in that case, that's the other one of these two problems. It's the one that's easier to fix, where you use an instruction that already exists, an LFENCE instruction, in order to essentially put up a speculation fence at critical parts of the code. The compiler...

Leo: See, this confuses me because, okay, they're going to fix the compiler.

Steve: It is confusing.

Leo: But if I'm writing malware, I'm sure as hell not going to set that flag; right?

Steve: Right.

Leo: So it's only - I don't understand what this fixes. It fixes applications I compile myself or write myself.

Steve: From being abused.

Leo: Ah. So I can't abuse that userland application. But if I wrote some malicious software I could still do that; right?

Steve: Right. And so that's what I mean. I mean, so this...

Leo: Kind of not much of a mitigation, in other words.

Steve: No. I mean, it...

Leo: Who compiles their own software?

Steve: Well, and that's just it. So the idea being that, as Windows gets recompiled, then its code will be prevented from being abused.

Leo: Well, but does it mean a rogue program won't work anymore on Windows?

Steve: Well, no. It means that a rogue program would not be able to steal data from those protected Windows programs.

Leo: Ah.

Steve: Would not be able to get into them.

Leo: So these are userland - okay. So that's the thing. What you want to steal is kernel processes; right? Or what you get is kernel processes. Or no?

Steve: Or like users' passwords and confidential data.

Leo: Because they're sitting in cache; right?

Steve: Right, because they...

Leo: From a kernel process. Or no?

Steve: Or maybe even more. How about like cryptographic keys?

Leo: Yeah. But where are those keys being - how are they getting in the cache? From userland apps or from kernel apps?

Steve: Well, yeah. Well, it depends on where the decryption is happening, but it's happening everywhere, essentially.

Leo: So I misunderstood, then, the problem with Spectre. I thought it was I could write a malicious userland app that would spy into cache that had been loaded during the speculative execution.

Steve: That is true.

Leo: During a context switch to kernel apps.

Steve: Correct.

Leo: The data that I would be seeing would be coming only from cache loaded by the kernel apps? Or would it be coming from cache loaded by userland apps?

Steve: It's actually able to get data out of main memory. And that's what has everyone so freaked out.

Leo: Out of RAM, got it.

Steve: Yes.

Leo: So there could be other userland apps running, as often happens in a computer, side by side. And their data would be in RAM, and this malicious userland app could get it.

Steve: Yes.

Leo: But is it only able to get it during a context switch to kernel?

Steve: Well, the way to think of it, it's only able to get it at a low bandwidth, like 2K bytes of data [crosstalk].

Leo: This like Rowhammer. This is that timing attack.

Steve: Yeah. Although, well, although it's very different. But the idea is that this branch target thing, what that allows someone to do is to reach over into the OS and obtain a specific byte of data. And you get it bit at a time, but you're able to essentially probe the RAM of the OS and know what you're probing. And that's why everyone is so freaked out. It is a...

Leo: So if I have a decrypted LastPass Vault in RAM, you could run a bad app that would be able to, not only get it, but it wouldn't be just a dump, it would be something - it would say, "I want to see that LastPass stuff."

Steve: Yes. Now...

Leo: That's risky, of course.

Steve: Oh, and it's why everyone's hair is on fire.

Leo: So if we recompile LastPass with this Retpoline, then it's not vulnerable to that?

Steve: Correct.

Leo: Got it. Now I get it.

Steve: But LastPass would be protecting itself from attackers.

Leo: So every developer will want to recompile with these flags turned on.

Steve: Exactly.

Leo: And they'll be protected. Is there a performance hit from Retpoline?

Steve: No.

Leo: Nice.

Steve: That's the other thing. It is, I mean, they've benchmarked it. I mean, it's barely detectable effect. But so, like, no - I don't want to say none because it's like...

Leo: But 1% or, yeah.

Steve: It's like an extra cycle of the processor.

Leo: Yeah. Oh, good.

Steve: it is negligible overhead.

Leo: And then, of course, if Microsoft recompiles Windows using it, that goes even further because now you can't take advantage of operating system processes.

Steve: Right, right.

Leo: Okay. But again, this isn't every possible vulnerability. But it is a large class of vulnerabilities.

Steve: Yeah.

Leo: Is it all the Spectre vulnerabilities?

Steve: No.

Leo: There's other ones.

Steve: There are two.

Leo: This is the branching vulnerability.

Steve: Yes. This is the one that, if it weren't for this, would require the microcode update.

Leo: Got it.

Steve: Which is good because it means, if we all did this, then the vulnerability, which is still theoretical, but the vulnerability that would otherwise force us to get a microcode update would be dramatically mitigated.

Leo: Got it. So this is the big one that the microcode update fixes.

Steve: Right, right. And Google found a way of just doing a little bit of trickery at the subroutine return, which is where kind of this vulnerability occurs, so that an attacker cannot control prediction of that event. And if the attacker cannot control prediction of the return, then the code ends up being safe.

Leo: Very nice.

Steve: So, nice piece of work. And, let's see, I said...

Leo: Sorry. I didn't mean to derail you. But I just - I wanted this clarification.

Steve: No, no, no. I'm sure that was useful for our listeners. So they wrote: "Variant" - and this is, you know, the 5715 - "triggers the speculative execution by utilizing branch target injection. It relies on the presence of a precisely defined instruction sequence in the privileged code, as well as the fact that memory accesses may cause allocation into the microprocessor's data cache even for speculatively executed instructions that never actually commit" and are retired, as we were talking about the last couple weeks. "As a result, an unprivileged attacker could use this flaw to cross the system call and the

guest/host boundaries to read privileged memory by conducting targeted cache side-channel attacks." I mean, this is all propellerhead stuff. I mean, it's way out there.

But they say: "Source can be recompiled with a Spectre-aware compiler" - and, by the way, they've already done this. There is a GCC has been made Spectre Aware so anybody who is using GCC can immediately do this. And Google provided this to the GCC guys. "Source can be recompiled with a Spectre-aware compiler to produce modified subroutine return code whose prediction cannot be externally manipulated." And that's the key.

And so just by tweaking the instruction sequence at the end of subroutines, which any compiler can be taught how to do, this problem goes away for that particular piece of code, meaning that that code is able to keep its secrets. So what we need, and I imagine this will happen over time with multi-gig downloads, is for Microsoft to roll up their sleeves and fix Windows in a way that then doesn't require, that at least dramatically reduces the importance of microcode being updated. Although we're going to talk about microcode updates here in a second.

I got a nice piece of email dated January 15th from a Torkel Hasle who's in - boy, how to pronounce this Norwegian - Sandefjord.

Leo: Sandefjord.

Steve: Sandefjord, thank you.

Leo: You know, I find it's easier to pronounce words from Scandinavian languages if you just go like this: Sandefjord, oopie-doopie, it makes it easier.

Steve: So in this case SpinRite saved the dentist his daughter works for $60,000, which is a pretty good bargain. Actually, the dentist saved, yeah, not even less 89 because the dentist didn't have to buy a copy. He wrote: "My daughter works for a group of dentists and called me" - so he must be like the local guru - "called me when the old PC that runs the machine did not boot. The machine operated a huge OPG" - which is a panoramic radiograph, so that sounds like something that does some crazy panorama of your face, you know, dentist stuff - "costing around $60,000 and was installed around 10 years ago. However, no support for newer versions of the OS. This expensive machine was dead, period." Well, they got 10 years out of it.

But anyway, he says: "I offered to run SpinRite for a few hours." And of course everybody knows how this story goes. "It worked for a long time with defective blocks. But after finishing, the machine booted and did a file check. All was good, and the $60,000 panoramic radiograph was live again. Since the machine was so valuable, and they had no backup, I bought a new hard disk" - and he says, parens, (SSD) - "cloned the old disk which SpinRite had made readable with Clonezilla" - which is Linux freeware, he says. Then he wrote: "Blew dust away, and booted with the new disk. All set. Hopefully the OPG will last another three to five years and postpone another $60,000 investment. Regards, Torkel Hasle," and you've got to help me here again, Leo.

Leo: Sandefjord.

Steve: Sandefjord, Norway.

Leo: I don't know if that's right. I shouldn't do that.

Steve: Well, no. But, you know, thank you. Better than mine.

Leo: No, I don't think so.

Steve: Well, okay. Russ Johnson asked: "Why not just code modern OSes to run the kernel on one core and keep user-mode applications on the others?" Or, he asks, "Do cores share caches?" And the answer is, yes, they do. There's typically L1, L2, and L3; and all of that cache is shared. Sometimes they're shared between threads on a core at the L1 level. Then sometimes multiple cores will each have their own L1 cache, but then they'll share an L2 that then goes down to the next level. And then everybody shares a much bigger L3. There's, like, there's a lot of juggling about architecture. And that's sort of why there are about 1,900 different versions of an Intel chip, because it's like, it's very confusing. It's like, well, do you want more of this or more of that? And it's like, okay, guys.

So anyway, but the answer is that you could technically split these things except that, then, think about it, one processor would only be - if you had two processors, one for the kernel and one for the user, typically you're bouncing back and forth between user and kernel. So you'd really be having one processor idle all the time, which would not be very efficient. And I also heard you talking during MacBreak Weekly, and we'll talk about this a little bit more, this whole issue of the performance hit from the Meltdown mitigations and what that means. And of course now I understand this because one of the things that InSpectre does, which Microsoft doesn't do, is it tells you if your system will see a significant slowdown from its current configuration, not just is it secure or not, but is it secure at a cost. And is that cost necessary?

So anyway, Simon Zerafa said - oh, this, I got a kick out of this. Remember we talked about an Internet-based storage device, Leo, which actually used ping commands in order to, like, you would send your database out on the Internet in pings, and they would get echoed back. And then you'd send them out again, essentially sort of a very dynamic file system. Well, not to be the last word on that, it turns out that there's a DNS file system where data is stored in DNS caches.

Leo: Oh, of course. Why not?

Steve: Exactly. Just anywhere you can squirrel away some data on the Internet, do it. So, yes. Anyway, so Ben Cox, whose Twitter handle is @Benjojo12, he says: "Introducing DNSFS, true cloud storage. Store your files in other people's DNS caches."

Leo: Wow.

Steve: Yeah.

Leo: What a great idea.

Steve: Because if it can be done, it's going to be done.

Leo: Sure.

Steve: Oh, and I got this so many times, I just happened to - in fact, I didn't even note who sent this to me because so many people tweeted this. "Attempted to add Session Manager per your recommendation to Firefox. It appears not yet to be compatible with Quantum, or I've got the wrong add-on. Not sure if you were aware." So first of all, I was delighted that, for whatever reason, my recommendation of Session Manager had so much uptake among our listeners. And last night I was at my Win7 machine that won't run an old version of Firefox, I'm running Quantum there, and it turns out it was Tab Session Manager that I was using.

Leo: Oh, yeah, I remember that, yeah.

Steve: So there's Tree-Style Tabs, and that still runs under Quantum. But it's Tab Session Manager. So many of our listeners were scratching their heads. I'm sorry. I wasn't clear. I'm using Session Manager on my old Firefox. On new Firefox, on Quantum, it's Tab Session Manager. So, happy to clarify.

And Philip O'Connell said: "Hey, Steve. I remember in a Security Now! episode you mentioned something you used to monitor web pages when you wanted to be notified of any updates to the monitored page. Do you recall the name of whatever it was you used? Thanks." So there's two things I use. The thing I really tend to use most is an add-on to Firefox called Check4Change, and that's a numeral four.

The way it works is, after you add it to Firefox, you go to a page that you want to keep an eye on. You mark a block of text, that is, the text that would be indicative of the change you're looking for, like on a movie page, something that you're sure would change when the page was updated to reflect that they're now showing you a different set of movies available at that theater. Then you right-click on the tab, and there will be on the tab's context, Check4Change is there, and then you are able to select how often you want it to reload the page. And then it just takes over. It just sits in the background, and every X minutes it will reload the page in the background and see whether the region you marked has changed. So it's nice because it's not in the way. It's there if you infrequently need that. But sometimes you do, sometimes if you really do want to check to see when something changes. And so it's out of the way most of the time, but really handy when that's what you want.

And then in Windows, that is, outside of any browser, it's just a little piece of freeware called WebMon, W-E-B-M-O-N. And it does what you think. It's free. It's simple. You give it a URL. And it's got other features, too. You can give it like a range of offsets from the page and stuff that it will then monitor. And so it just does it independently. It's a non-browser-based query to a remote site that waits for a page to change and lets you know when it does. So those are my two web browser update goodies. And lastly...

Leo: One more feedback from the peanut gallery, the chatroom. Sandefjord Sam says this is how you pronounce Sandefjord. And of course we have no audio at all.

RECORDING: Sandefjord, Norway.

Leo: Oh. Sandefjord, Norway.

RECORDING: Sandefjord, Norway.

Leo: Sandefjord.

Steve: Very good job, Leo.

Leo: I was close.

Steve: Yeah.

Leo: Sandefjord. That's Emma pronouncing - EmmaSaying.com. So we're going to trust Emma.

Steve: Perfect. I do. She sounds very authentic to me.

Leo: She does.

Steve: So several people commented that they were stunned by how much turning tracking protection on in Firefox to "Always" improved their experience. We talked about it last week. It made you a little uncomfortable, understandably, because it looked like it was just going to do blanket advertising blocking with no finesse. And as we know, the 'Net is becoming increasingly dependent on advertising. But after a number of tweets from people saying wow, Grant Taylor sent me a note saying "Firefox's tracking protection always on even helped with uBlock Origin enabled," meaning it does more.

But here's the point, the reason I'm bringing it up, and it can be disabled per site. So we get the best of both. If you have sites that you want to support, that complain if you have adblocking, tracking protection on, you can turn it off, and it's sticky. So thanks, Grant, for bringing that up. And of course so that makes it useful for people who do want to have maximum speed. And the feedback has been strong that it really makes a difference. Yet as we do, we want to be able to support sites that say, hey, please, we need you to look at our ads. And so we're able to do so.

Leo: Nice. Very good. Here's, by the way, just for completion, here's how you

pronounce Alex Ionescu's name.

RECORDING: PronounceNames.com.

Leo: Yeah, thank you.

RECORDING: Ionescu.

Leo: Ionescu.

Steve: Ionesco.

Leo: Ionescu. There's Ionesco, which is Eugene Ionesco, who was a famous playwright - was that his first name? - who wrote "Rhinoceros." But then there's Ionescu. But I think they're probably variants of the same Romanian name. So Ionescu, Alex Ionescu. I just want to make sure we get his name right.

Steve: I'm happy to do that. This is an audio podcast. We're not doing sign language, so it does matter.

Leo: All right. Steve Gibson, GRC.com, has a new tool for all of us.

Steve: So you like it.

Leo: I love it. So I have run - it started with - and I did this I think on Windows Weekly last week, the PowerShell script that Microsoft put out, the Speculation Execution Validator. I've run other tools. But only yours tells us about vulnerabilities specifically to Meltdown, to Spectre, and what the performance hit is. And you have a very good explainer in there. Go ahead. You want me to show it? Or do you want to talk about it first?

Steve: Yeah, go ahead and show it, yeah.

Leo: So I ran it on the fully mitigated Lenovo, and that's what the screen looks like. And I'm happy to say vulnerable to Meltdown, no; vulnerable to Spectre, no; and performance, good. And then this is what I really think you're doing everybody a service because you're kind of explaining it in the app. And this is something I've not seen anywhere. So if you are getting a performance hit, you can disable protection. You have to reboot. But the ability to take it out and get full performance if you need it, and then put it back, is huge, I think.

Steve: Yeah, yeah. In fact, the show notes start out with a tweet three hours ago from a Brian Yates, who said: "Thank you, sir. My work laptop went to a crawl after latest Win update. Seen your tweet about the freeware. It showed my performance slower. Yeah, no joke. So disabled Meltdown fix, restarted, back to normal."

Leo: Now, you might want to reenable it; right?

Steve: So, yeah.

Leo: Is there a way to reenable it? Does that button change to Enable?

Steve: Oh, yeah, yes.

Leo: Oh, nice, nice.

Steve: It toggles back and forth.

Leo: Now, I should point out, so I also, as you said, it does run on a Macintosh with WINE. But I don't know if it's seeing the operating system. I've done all the patches here, as I'm just showing you the update. And I know this is small. It's kind of hard to read. But it says, "Vulnerable to Meltdown, yes; vulnerable to Spectre, yes; performance, good." I'm not sure that's accurate.

Steve: Okay. So the question...

Leo: This is a fully patched Macintosh.

Steve: The question would be - and you're right, it's going to have a problem because it's going to be seeing WINE. But you know how I break everything down. One of the things it says is, and I don't remember exactly what the language is I used. There is, you know how I show individual paragraphs to explain what it has found, there is something about the performance, whether the hardware supports high performance mitigation of Meltdown.

Leo: That part is green. It says it does support that.

Steve: Then that shows your firmware is patched on that Macintosh.

Leo: Good.

Steve: That's something I -

Leo: That's why I'm saying good. But it says the hardware has not been updated with new features required to protect against Spectre.

Steve: Correct.

Leo: And it's the 64-bit of Windows, version of Windows. Obviously that's wrong because I'm not running Windows, I'm running WINE. So that would be - it says it's not aware of Spectre or Meltdown. But of course WINE probably isn't. But that doesn't say anything about whether macOS is; right?

Steve: Correct.

Leo: So I'm going to ignore that paragraph. But it does say "System hardware has not been updated with new features allowing the operating system to protect against Spectre."

Steve: I'm trying to remember...

Leo: Same with Meltdown.

Steve: Yeah.

Leo: But that's the operating system; right? So it's really the hardware mitigations that I'm curious about. The registry is configured properly, but that's ironic. I don't even know if there is a registry.

Steve: Correct. And so the Intel processor's CPU ID instruction contains this wealth of information.

Leo: Yeah. That's where you're getting this from?

Steve: Yes. And so I'm pulling it from the chip itself, underneath the level of the OS. And I wanted to do that because all of the other, I mean, I'm sure you saw the one, in fact I think I saw you running it over the weekend, the Ashampoo thing.

Leo: Yeah, that was dopey.

Steve: Well, you know that it makes a network connection. I didn't realize, but it's connecting back to the mothership for some reason.

Leo: That might be benign, but I - yeah.

Steve: It's like, okay. Well, no. And I don't think they're bad people. It's just that, okay, why do it? And also it pretends to be scanning for, like, a minute or so.

Leo: Yeah. I knew that was bogus.

Steve: It was like, what the hell.

Leo: There's nothing to scan. You just query the CPU, and it tells you.

Steve: Yeah. But anyway...

Leo: So I should say, on this machine I'm running, this is an older machine. This is a late 2014 5K iMac, which means it's running probably a Haswell or earlier chipset. So I am going to run this on my iMac Pro at home, and I'll send you a screenshot.

Steve: I would love to know because I'm trying to remember, there was some data that I thought I could trust from Windows, but I am pulling some from the CPU ID. But I'm not...

Leo: And that should still work under WINE; right?

Steve: Yeah. But I'm not pulling both. I don't remember now if it was Meltdown or Spectre. One or the other, I think it was Spectre - man. No. Anyway, I'll have to look because this thing is...

Leo: I'm just saying it's just great for a PC. But if you're running it on Linux or macOS, Steve's probably not getting all the information that would be useful in this case.

Steve: Correct.

Leo: It's being hidden.

Steve: Now, here's one of the really interesting things, speaking of Linux, is the way microcode is updated. When we talk about updating our microcode or needing like a microcode patch or a microcode fix, notice that it always comes in the form of a BIOS update.

Leo: It's firmware, yeah.

Steve: Well, yes. But it's not written into the processor. And that's the key is that it is dynamically loaded by the BIOS every time you boot. So if you look at the second link here at the end of the show notes, Leo, this is Intel has published the Linux processor microcode data file. Linux has the ability to load updated microcode for, I mean, to me it looks like every processor Intel has ever made.

Leo: Yeah. It's a modular kernel. So I'm sure it's part - it's a kernel mod, probably.

Steve: Well, it turns out if it's - I think it was under, god, it was in the - if you place this file in a certain directory in Linux...

Leo: Oh, how interesting.

Steve: ...it will dynamically...

Leo: Oh, that's awesome.

Steve: Yes.

Leo: I need this. Because I'll tell you what, none of my Linux machines are mitigated as far as I can tell. There's been many patches and updates to Arch Linux and to Ubuntu and to Debian. But none of them have received firmware updates. So this is going to be a boon.

Steve: Yes.

Leo: If you figure out what processor you have.

Steve: Yes.

Leo: That's nice. You just put this file in the right spot, and you're good to go.

Steve: Yes.

Leo: Love that.

Steve: And so this solves - so my point is that, first of all, Linux has always had this

feature, that you're able to fix microcode problems by just putting a microcode data file in Linux, and it patches the processor.

Leo: And then you put it in the ETSI firmware directory.

Steve: That's it, yes.

Leo: Wow, that's so cool. That's so cool.

Steve: That's it.

Leo: All right.

Steve: VMware has something. If you click the link above that one, you'll see that VMware has a CPU microcode update driver. It's not, like, formally supported. They produced it. And it works for Intel and AMD processors. The problem is that VMware loads on top of the operating system, after Windows has decided whether or not it has updated microcode. So for Windows users, patching the microcode after Windows is running doesn't help us. But this does say that, if Microsoft cared to...

Leo: Oh, they could do it, sure.

Steve: Yes. They could do the same thing Linux...

Leo: All they'd have to do is check and say, oh, let's load this in.

Steve: Yes. They could do the same thing that Linux has long been doing of loading a microcode update on the fly. And so when we are flashing our firmware, when we're flashing our BIOS, when we're updating our microcode, as you and I have done recently on Lenovo and various machines, what we're really doing is that microcode is being loaded into the flash ROM. And then, when the system boots, the BIOS patches the microcode.

Leo: Wow.

Steve: It's literally an on-the-fly patch because, think about it, microcode has to be the fastest thing on the planet. I mean, it's executing at that sub-gigahertz, at the 3GHz rate. It's like a processor in the processor, microcode deciding what to do. So there's no way it could even be flash. It's got to be RAM. It's got to be able to run at full-on lightspeed speed. So the chip has default microcode; but the BIOS, there is a provision for patching the microcode, for updating microcode at power-up, and that's the responsibility of the BIOS. So that's why...

Leo: And that's for Meltdown specifically?

Steve: No. That's for...

Leo: That's for Spectre.

Steve: That's for Spectre.

Leo: Okay.

Steve: Okay. So here's the deal with Meltdown. Meltdown can be mitigated with recent features. And I have this in my notes. There was one feature added - shoot, I posted it in the newsgroup and didn't copy it into my show notes. There are two instructions or features. There's something called PCID, and then there's Invalidate PCID, INVPCID. Intel put this into the chipset, like in 2010, that first instruction, PCID. And what this does is it tags the cache with a - PCID stands for Process Context Identifier. So it adds a tag to all of the cache entries indicating which process caused this to be loaded into the cache. And if this was used by operating systems, Meltdown would be prevented. But no OS bothered. And it turns out the engineers at Intel didn't really think it through very well. It turned out to actually do it was really burdensome.

So a few years later they added a second instruction, Invalidate PCID, INVPCID. If you have both of them, then it is easy for an operating system to solve the Meltdown problem with no overhead, no flushing. What this does is those two instructions allow the cache to be tagged in a way that prevents any cross-process bleed, at no expense, like no performance overhead.

Leo: So that's why Intel said Haswell or earlier you're going to have a bigger hit.

Steve: Yes.

Leo: So post-Haswell. So if you have generation 5, 6, 7, or 8 Intel processors, it has that instruction, probably.

Steve: Both instructions.

Leo: Both instructions probably, yes, yes.

Steve: Yeah. Because they did sort of a half measure a few years before.

Leo: Did that mean they saw this coming?

Steve: Yeah. I mean, they - yes. And as we know, it was, as I heard you mention during MacBreak Weekly correctly, and we talked about it on Security Now! last week, there's a 1995 paper cautioning about these sorts of problems.

Leo: Terrible.

Steve: The idea, I mean, sharing the cache is so efficient that they just wanted to. And so they gave us a mechanism for preventing cache probing attacks. But because nobody was probing anyone's cache, nobody built it in. And it was like, yeah, okay, fine. Well, so that's why in a matter of, like, a few weeks, Microsoft was able to say, oops, we'd better do that. So if those two instructions exist, then the cache does not need to be flushed. It can simply be tagged using this Process Context ID in a way that prevents Meltdown completely. And so no performance hit. But that's only available on recent chips.

And there isn't - I don't hold Microsoft responsible for not doing this except that - get this, Leo. If you don't have Fall Creators Update, even on Windows 10, they're not giving you this. The 1703 build that was the pre-Fall Creators Update, it's not taking advantage of these instructions. And, I mean, Leo, it's the same code. There is - I cannot forgive Microsoft for saying, oh, if you're not using Fall Creators Update version of Windows 10, we're going to slow you down in order to give you Meltdown protection.

Leo: Well, there's a class action lawsuit waiting to happen. Holy cow. I mean, let's - maybe, to be fair, I'm thinking maybe there was kind of something in the Fall Creators Update that made this easier to do.

Steve: It's the same code base. You know it is.

Leo: Yeah.

Steve: I mean, look at the dialog boxes. They still look like XP if you dig down far enough.

Leo: I know, I know. And I'm sure they will roll out more patches. I mean, this is kind of the - I'm wondering, InSpectre itself, you're never going to be done with this because there's going to be more and more all the time to mitigate; right? Or no. If I get green across the board, I'm good.

Steve: Yes.

Leo: On this vulnerability, anyway, or these two CVEs.

Steve: I just posted, while you were doing that second sponsor slot, I posted a test release to the newsgroup. If anyone's curious, well, I won't tell you where it is. That would be a disaster.

Leo: No, but I will tell you where the current InSpectre is; right? GRC.com, and then click the Security, or actually the Freeware tab, Security tab within the Freeware tab, and then you'll see it right at the top, InSpectre.

Steve: Right. And Google has now got it as - it's the first three links.

Leo: I saw that, yup.

Steve: If you just look for "InSpectre."

Leo: Google "InSpectre," I-N-S-P-E-C-T-R-E. And, yeah, I saw Woody Leonard gave you a plug in Computer World.

Steve: Very nice.

Leo: gHacks has it. I'm sure it will be mentioned all over. It is easily the simplest and smallest and yet most feature-rich InSpectre checker I've tried. Highly recommended.

Steve: So what we just realized last night, and I found out about it this morning, is that it was causing false positives on a bunch of AV. And somebody who had been testing them all week, or the last five days as I've been iterating this, said, "Steve" - I don't remember what his AV was. But he said: "No version of InSpectre until the final one was triggering AV." Well, the final one is where I added the Enable/Disable buttons. And so I thought, oh, it's annoyed with the registry key. It sees my use of that registry key as being malicious and raising a red flag. So I just posted a version that should bypass AV. I scrambled the registry key, and I decrypt it on the fly so that no scam will see it. And I imagine we'll go back then to being quiet as we were before on AV. So the gang in the newsgroup are testing it right now.

I want to - there's one bit of wording I want to clarify where two of the paragraphs seem to be conflicting with each other. But then I think I'm done. I was worried that, if it was possible for us, that is, like this VMware solution to patch microcode, that I was going to have to do that because this is the big remaining problem is that, without the microcode patch, there is really no way to mitigate one of these two Spectre problems without recompilation. Now, maybe that'll just happen organically throughout the entire industry. All the compilers will just add - which really seems a shame - add this code. Or maybe they'll add it by default until all the processors do get fixed, and then we can stop horsing around with this Retpoline solution that Google came up with.

But the real problem is, when you think about it, the reason our IoT devices have had such problems is that light bulbs don't update their own firmware. And up until this point the security problems have been caused by the OS and thus fixed by OS updates and patches. This is arguably the first time that we've seen a severe security vulnerability caused by or leveraging a feature of the processor. And so suddenly our PCs are sort of like our IoT, inasmuch as we're realizing that there isn't an ecosystem in place for responsibly keeping processor firmware up to date in the same way we're keeping our

router firmware up to date. We just really haven't had to until now.

And so I was very glad that my Lenovo Carbon laptop could get updated. I have a Dell laptop that's the E7440. And Dell, being a mainline supplier, they were right on the ball. They had across-the-board firmware for their laptops on the 8th, I think it was dated. So they responded immediately to this. But how many of us are using computers we built, or computers with an unknown heritage? I mean, they're just never going to get firmware updates.

Leo: And not to mention a bunch of smartphones, IoT devices.

Steve: Oh, yes, yes.

Leo: This is, I would say, the vast majority of Intel and AMD and ARM processors out there will never be patched. By the way, does your program work for AMD?

Steve: Yes. It is AMD-aware.

Leo: Okay.

Steve: It has whole separate blocks of text for AMD, explaining that you never were vulnerable. It disables the left-hand Meltdown pushbutton because there's no way to enable or disable Meltdown on AMD because you're never vulnerable. No, I mean, it's - you're going to like it, Leo, the more you see it. It just really explains what's going on with your system.

Leo: That's nice. That's really nice. We should mention a number of people are reporting that some antiviruses flag this as malware when you download it. Obviously it's not malware. That's, I don't know, there's some false positive going on.

Steve: Well, yeah. In fact, that's what I was just talking about is that I already have a test version that will bypass that malware.

Leo: Oh, I'm sorry, yes, of course, okay.

Steve: Yeah. And so that's what that is. You're right. It is not malware. It's signed by me. It's a hundred - I was annoyed. It's 125K.

Leo: Why so big, Steve?

Steve: Because, if you're going to have Windows 10, you need large high-color icons.

Leo: Oh. The icon.

Steve: The icon is 93K. 93K for GD icons.

Leo: Well, it's a nice icon. Who did that, the little ghost icon? I like it.

Steve: I did.

Leo: It's very nice.

Steve: I took the Spectre icon and then made it unhappy and pissed off-looking and took the stick out of its hand. Yeah, but 93K of my 125 is icons.

Leo: So really this is just a 22K [crosstalk].

Steve: Yeah, it's the proper size for something like this, yeah.

Leo: Well, thank you, Steve, for putting this out. And of course, as always, letting us know. Somebody's saying I bet you spent as much time on the ghost graphic as the program. I don't think that's true, but...

Steve: No, no. I've got good tools. So relative to disabling, we should talk about that for a second.

Leo: Oh, yes, good.

Steve: Okay. At this moment we don't know - I should mention there's been a lot of dialogue in the GRC newsgroups about turn it on, turn it off, what should we do? We don't know of a single instance of exploit of these problems. I salute the industry for no longer being lazy about this and going, eh, it's not a problem, wait till somebody attacks us. No. We've learned our lesson. Everybody has jumped on this immediately. Clearly, Meltdown is the most worrisome and the easiest to exploit problem; and, with the addition on older processors, flushing the cache at a cost of performance mitigates that vulnerability on Haswell and later. And InSpectre will tell you whether you have a newer or later processor.

On the newer ones, it's just you need OS support, so in order not to have the cheap, but easy, yet expensive performance hit. And this is where, I mean, I'm annoyed that Microsoft hasn't done it in the pre-Fall Creators Update. They haven't done it for Windows 8.1 and 7, either, and they could. So I'll be really interested to see whether, like with next month's Quality Rollup, we get this fixed because Windows 7 and 8.1 and even Windows 10 before the Fall Creators Update are - they have Meltdown mitigation, but they've got the performance expensive mitigation, even on a new processor.

So you need both a newer version, actually the newest version of Windows, Windows 10 Fall Creators Update, and a very recent processor for Meltdown to be - for you to be protected against Meltdown. And that's with no performance hit. If you have an older processor or an older Windows, any older Windows, then you're protected; but XP isn't because they're not going that far back. But Windows 7 and on you're protected with a performance hit.

So at this moment today, Meltdown is the easiest to exploit; yet it is, as far as we know, unexploited. Yet to varying degrees, people are experiencing a performance hit on, if they don't have Fall Creators Update, with recent enough hardware. So the question is should they? I don't know. I don't think I'm going to put that mitigation in place for GRC's servers because nobody runs ever, ever runs suspicious code on GRC's servers. It just doesn't happen. I'm not your typical, oh, let's download this piece of software. I mean, those servers are never touched. So on the other hand, they're also very lightly loaded.

So it may be that I will end up getting around to do that. But for the typical user, certainly for the podcast listener, people listening to Security Now!, you will know the instant we find Meltdown being used in the wild. I do expect it. I think that there are so many systems which are not going to be fixed, that may not have their OSes updated, that will be victims. And Meltdown is so simple to, I mean, we're going to be covering Meltdown exploitation just as I did with those couple tweets from the researchers who were saying, oh, my goodness, this wasn't hard. I mean, so my guess is it's going to get exploited. And at some point we'll all want to have our Meltdown shields up.

But if you're not running the latest Windows 10, if you don't have recent enough hardware, if you actually feel the slowdown, as that one person who tweeted really did, then I would argue there's a case to be made for turning it off until there's some reason to turn it on. Again, personal decision. Individuals can make the decision. InSpectre, my little free app, makes it easy to flip it back and forth. And in fact you can turn it off and then reboot, do some things and see how your system feels, flip it back on, reboot, do the same thing. See if you notice a difference. If you really don't notice a difference, yeah, then I would say, by all means, run with Meltdown protection enabled. If it does really affect what you're doing, then today I would say it's safe to turn it off. There's a known vulnerability, but we know of no exploitation of it yet.

Leo: Okay. There you have it. And there's no reason not to get InSpectre right now. It's free at GRC.com, or Google "InSpectre." And while you're there, get SpinRite. I heard a yabba-dabba-doo during the show.

Steve: We had one, yes.

Leo: I like that. We want to hear some more yabba-dabba-doos as people pick up the world's best hard drive recovery and maintenance utility. Steve, it's awesome what Steve does. And this utility is just a simple little bit of all the many things he does for free, which you'll find at GRC.com. SpinRite's the one that funds it all.

He also for free offers this show, audio and transcripts at GRC.com. Thank you for paying for those. We have audio and video at our site, TWiT.tv/sn. You can also subscribe to the audio or video feeds. Any podcast program should know about

Security Now! by now. And I think it's a good idea to subscribe. That way you get it the minute it's out. And you can listen at your leisure. Just search for "Security Now!" in your podcast app.

We do this show every - if you want to watch live, you can. Get the latest, freshest version of Security Now!, well, that's about 1:30 p.m. Pacific, 4:30 Eastern, 21:30 UTC every Tuesday right after MacBreak Weekly. And you can watch it at TWiT.tv/live. And you could also join us in the chatroom. It's always - this is a good show to be in the chatroom. Lots of explication, exclamations, and fun and games at irc.twit.tv. They're loving Steve. They've loving Steve for InSpectre. Good job, Steve. Thank you very much. Well, that's it, I guess, for Security Now!.

Steve: Yup. We'll be back next week with some more news and updates and so on.

Leo: I'll let you get back to your newsgroups and SQRL.

Steve: Till Monday.

Leo: Right. See you, Steve.

Steve: Yes, sir. Thanks.