SECURITY NOW!

Transcript of Episode #672

All Up in Their Business

Description: This week we look at even MORE new Spectre-related attacks, highlights from last Tuesday's monthly patch event, advances in GPS spoofing technology, GitHub's welcome help with security dependencies, Chrome's new (or forthcoming) "Site Isolation" feature, when hackers DO look behind the routers they commandeer, and the consequences of deliberate BGP routing misbehavior. Plus, reading between the lines of last Friday's DOJ indictment of the U.S. 2016 election hacking by 12 Russian operatives, the U.S. appears to really have been "all up in their business."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-672.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-672-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson's here to talk about, yes, even more Spectre-related attacks, highlights from last Tuesday's monthly patch event, GPS spoofing - yeah, it's as bad as it sounds. And he was blown away by the amount of detail revealed in last Friday's Department of Justice indictment against Russian spies, basically. He talks about that, too. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 672, recorded Tuesday, July 17th, 2018: All Up in Their Business.

It's time for Security Now!. Yay. The highlight of the week for many of you, I know. Steve Gibson is here, and we're ready to talk about a world at war.

Steve Gibson: Well, so this was actually initially titled "More Spectre Madness" because, believe it or not, there is more Spectre madness. Two researchers affiliated with MIT won $100,000 reward just recently by Intel. Remember that Intel put up a bounty, I think it was a quarter million dollar bounty, through the end of the year for any additional discoveries. Well, we had some. And so that was all sort of set up to make that be the big deal.

But then I started the 29-page DoJ indictment which we got last Friday against the 12 Russian agents who, it is alleged, hacked into the Democratic side of the 2016 U.S. election. And what stunned me, I mean, and it's almost chilling, is the degree of detail. And of course we're all about technology on the podcast, we and all of our listeners. And so it just changed the subject of the podcast. Now this Security Now! #672 for July 17 is titled "All Up in Their Business."

Leo: What does that mean?

Steve: We are all up in the Russian business. I could not believe, I mean, on this day between this time and that time, such and so did the following Google searches upstream by three hours to research some English to get the English right on a blog posting where they created the Guccifer 2.0 identity in order to rebut the WikiLeaks claims and blah blah blah. But, I mean, the level of detail, I mean, if I were the Russians, I'd be looking around thinking, what? I mean, and of course it's always a problem because, when you divulge this sort of detail, there's this assumption of sources and methods. And so oftentimes you're wanting to keep what you know somewhat under wraps because, if you say too much, then there can only be one or two ways such and such is known.

But anyway, so we're going to wrap up this week by taking a look at some of the allegations in the indictment. But what's stunning to me and any technologist who listens to the podcast would be, okay, wait a minute. How do they know that? I mean, like how retrospectively? Because this wasn't known at the time. This had to be forensically, like, gone back and figured out, which suggests a number of things that we'll talk about, about the level of monitoring of the Internet that this suggests we have, which makes Edward Snowden look like he was using training wheels. I mean, it's just astonishing.

So anyway, but other things happened, too. As I mentioned, we have new Spectre-related attacks. We've got some highlights from last Tuesday's monthly patch event; and a couple of interesting advances in GPS spoofing technology where the researchers went beyond, like, telling someone to turn left in the middle of a freeway, which is like, what?, to actually figuring out how to spoof GPS to the level required to give somebody believable, workable, alternative driving routes to take them somewhere of the spoofer's choosing, which opens up a whole - oh, and for less than $300, just using a Raspberry Pi and a software-defined radio. I mean, so like the bar has been lowered and the capability expanded. So we're sort of just keeping ourselves up to date on what's possible.

GitHub also added another piece to their very welcome help with informing repository owners or managers of dependency problems with their projects. Chrome apparently has pushed out - but they didn't push it out to me, so I don't know. It's supposed to be 99 percent pushed, but I don't know why I don't have it. But something known as "site isolation" has been added that we're going to talk about.

Also I keep saying, when we're talking about how Shodan can be used to find vulnerable routers that traffic can be bounced off of, our listeners have heard me saying, yeah, just wait till those hackers start looking inside the networks of the routers that they're just currently bouncing traffic off of. Well, that happened, and with some interesting consequences.

Also, there's been a Portuguese ISP abusing their role as an Internet exchange point. Well, they're now dark. Nobody is sending them any traffic anymore. We'll talk about that, and then wrap up, talking about - oh, I did have a follow-up to last week's discussion of using SpinRite with a RAID where the guy who wanted to run SpinRite on his drives didn't take want to take his RAID down overnight. Somebody who knows some details about the way that RAID works came up with a way that doesn't require a big rebuild afterward. So I wanted to share that. And then we'll talk about this stunning 29-page document that, I mean, it's just - it's amazing what we know. So yikes. Really interesting stuff.

Leo: Yes. And we can show you the illustration, which you won't be able to read.

Steve: Won't be able to read, but you get a sense of the connectivity, yes, our Picture of the Week.

Leo: Okay, Steverino.

Steve: So our Picture of the Week isn't, as you said, super informative because it is the best resolution I could find, but it's blurry. So I didn't expand it. It's just not very clear. But it does show the interconnection of the various players in this drama, what roles they had, where their identities were used and so forth. So it's just sort of - it's the sort of thing we see with forensic investigations where who was in charge of what and who called who and who had which identities and so forth. The details in the text are very clear, and we'll be talking about that in a few minutes.

I did want to mention, as I said at the top of the show, and also as I said at the top of the year, about the speculative execution problems, that the problem is, like, fundamental to the architecture of today's processors. And as I've thought about this more, and I read this paper, these guys, they earned their $100,000, their bounty awarded by Intel for this work. It is the clearest and most, well, I would say damning or worrisome or earthshaking indictment of any sort of tricks which are played for speculation that I've seen. I would argue that they have advanced the state of the art in this paper significantly.

And there's really no immediate takeaway for us because the good news is end users, as we know, have never really been at too much risk. The big risk is where there's a potential to have an adversary sharing the same processor. Well, since in a single user system like we're all using with our laptops and desktops and our mobile devices, we're the only one using it. So if malware is in the device, well, we're already compromised. But in a virtual server, virtual machine cloud environment where the design of the system is multiple customers sharing the same hardware, the problem of the cross-VM information leakage which had until the beginning of the year been just assumed because we had implemented the architectures to isolate processes, that process and machine isolation was a given. And it was the rare exception, I remember there was a floppy driver a few years ago in the VMware where you could give it some weird instructions and break into the kernel. But that was the exception to this isolation rule.

Where I'm going with this is that I wouldn't be surprised if, moving forward, the only way we solve this problem is by not allowing sharing of a core and the core's resources, which means its caching also, among systems that may be hostile to each other. The whole point of speculation is that this processor learns what it's doing and is better at doing it in the future. That's what caching gives us. It's what branch prediction gives us. It's what all of these tricks where the system only runs at the performance it does because it's remembering, it's taking advantage of the fact that code that has just run tends to run again.

Without that, you know, main memory is just incredibly slow relative to the processor speed, which is why we have all these multimeg caches now in order to bring that local. I just don't see how it's possible to share that without carefully tagging which core owns which items in the cache, which could be done. And possibly just giving up on the idea of a multicore processor being considered as a symmetric thread pool where anybody can run any thread from any core. It may be necessary to partition in the future, to partition in these environments where hostility is possible, to partition cores to virtual machines so that there just isn't cross-core leakage because I can't see any way for someone to securely share a single core, given that we need speculation, unless everything was tagged.

And frankly, tagging everything would not reduce performance, and it would solve the problem, but at horrendous hardware expense. That is, right now the presumption has been there was no need to tag everything. And when I say "tagging," I mean identify which process caused what modification for the future which would again, if the tags match, we reuse what we learned of that process. Another process is not going to be using the same architecture. So we actually could see an improvement in overall performance by heavily tagging what the processor learns about what the processes are doing on it, and we would get a performance benefit because we wouldn't be flushing what we learned from one process by another, which would be a benefit.

But, boy. And it doesn't really - it's not a conceptual stretch. Everybody knows how to do a fully tagged system. But it would expand the die size and increase the complexity of the hardware significantly. Either that or strict processor partitioning is probably where we have to go. Anyway, so I'll just say that what these guys did, they named these two new findings which fell out of their very powerful research, they called them Spectre 1.1 and 1.2 because they're related to the original Spectre 1.0 flaw. Spectre 1.0 was all about speculative reads. What they realized was speculative stores could be used in a similar fashion. And so they've extended our understanding of the speculation risk significantly, essentially by not missing anything in their analysis.

And what's significant to us, well, or to the industry, is that their approach bypasses the mitigations, the software mitigations which were put in place to solve the Spectre 1.0 problems, meaning that there is no current protection from what these guys have done. And at the same time there also never was, as we've said, a huge risk to end users. The risks have been bigger to cloud providers.

And so I don't know whether Intel will respond, whether they can. It may be that we'll see another round of software mitigations. We might see more microcode, again, thinking that we were finally done after half a year of this microcode mess that we went through. This just happened. So there is yet no real sense.

Intel has acknowledged that they - in addition to paying $100,000 they've acknowledged that, yes, their systems are vulnerable to these two 1.1 and 1.2 vulnerabilities, as has ARM. AMD so far has been silent. But AMD is traditionally slower to respond to security issues. They're looking at it, I'm sure, and will probably end up saying, yeah, this gets us, too. Because basically this just puts another nail in the speculation coffin.

And, I mean, the paper is amazing. If anyone, again, I will just - I've got a link in the show notes for anyone who's interested. It's a tour de force in really understanding the nature of this problem and argues convincingly that there isn't, as I said, there just isn't a safe way to do this. If something your code does changes the future, and it has to in order to get the performance it needs, then if that changed future can also affect a different process, you've got cross-process information leakage. And as we saw recently, and as these guys reiterated in their paper, things like fuzzing the timing in the browsers only slowed down the bandwidth or the bit rate of information leakage. It didn't end it. So it's still a problem.

Anyway, really, really great work. And in fact Bleeping Computer covered this and had on their page a really nice summary of where we've been so far this year with Variant 1, and now we have 1.1 and 1.2. And of course we have 2, 3, 3a, and 4. And so it just, you know, and 4 was, remember Spectre Next-Generation, or SpectreNG. So no word yet on AMD's effect, nor anyone's formal response. This all just happened. But there's no reason to panic because, as we've seen, even the original problems were more of a computer science concern that we had to take seriously because everything we've learned says these problems only get worse. And this is what we're seeing. I mean, now, as our understanding seven months later is continuing to mature, it's looking to be bleaker than we even thought.

So hats off to these guys. And they really did earn their hundred grand from Intel. And than you to Intel for offering a bounty. I'm sure they would have done the research anyway. But if it encourages people to spend the time, which was extensive in this case, to understand it, then yay.

Last Tuesday was the second Tuesday of the month, and Microsoft fixed 53 vulnerabilities occurring across 15 of their products. Nothing really stood out except I did want to close the loop on a topic we discussed a couple weeks before, which was the Lazy FP State Restore. This was yet another vulnerability, not technically speculative, but this was the one where Intel added a feature where the expensive-to-restore state of the floating point unit, due to all of the large registers that floating point math uses, having to reload them when a thread starts to execute is expensive. And so someone somewhere I'm sure did some profiling years ago and said, hey, you know, sometimes a thread gets execution, and it never does any FPU stuff. So let's not bother to restore the FPU registers unless it tries.

Anyway, so yes, there's another optimization. And sure enough, it can be abused in order to leak information, as we saw. So Microsoft did issue a patch for that, essentially not taking advantage of that optimization. And as we also mentioned when we talked about it the first time, a number of other OSes had already decided the edge wasn't even worth the complexity that it added. So they had backed off and were never affected by it. So now, after last week, Windows isn't either.

The winner of last week's Patch Tuesday was Adobe, who fixed more than twice as many vulnerabilities, count them, 112. And also what was interesting was how lopsided their location was. Almost all of them were in Adobe Acrobat and Reader. They published two for Flash Player, three for their Experience Manager, three for Connect. And the balance, 104 vulnerabilities, were fixed in Acrobat and Reader. And we would argue that Acrobat and Reader are Adobe's largest interpreters. And as we know, interpreters are notoriously difficult to make work in a way that prevents their abuse. And sure enough.

And what I'm interested in knowing - there's no information about this without a lot of digging. It would certainly be possible to determine. But I'm wondering how far back the vulnerabilities go and how many of them were introduced in the last, oh, say 10 years because we had Acrobat Reader 10 years ago, and it worked just great. And PDFs popped up, and you could read PDFs, and everything seemed fine. And I just wondering what percentage of these are new problems that Adobe has introduced into their own software as a consequence of continually adding new features because that's what they have to do for their own corporate interest.

I just wish, once the bugs are out, they would leave it alone because, as we know, we had that e=mc2 Picture of the Week a few weeks back, and we know that the number of security vulnerabilities tends to increase exponentially as code continues to grow in size because it's just difficult to keep everything straightened out and not interacting with each other.

Anyway, so all of those are fixed. I'm sure everyone has updated Windows by now. And of course, as we know, it's becoming increasingly important to do so, maybe not for the typical end user, but certainly if you're any sort of an attack target. And I'll make sure to mention it when we're talking about the way the Russian agents managed to get into these systems. Nothing was super high-tech or fancy. Largely it was social engineering. And I just completely distracted myself and lost the thread.

Leo: What you looking at? Squirrel!

Steve: Yeah, it's like, where am I going? What is that about? I have no idea. Oh, about if you might be the target of attacks, then you really - we know that when vulnerabilities are patched they are now often very quickly reverse engineered to figure out what it is they fixed. And there's then a window of opportunity that opens, during which time, from the time the vulnerability is known to the time the target system actually gets updated, someone can be taken advantage of. And if nothing else, as a consequence of all the press that has occurred in the wake of the U.S. election two years ago, there can't be any navet on the part of those involved in the political system. As we will see here by the end of the podcast, people were clicking on links they shouldn't have. And the rest now is the subject of an indictment with lots of details.

Okay. So the paper was jointly published by researchers from Virginia Tech, the University of Electronic Science and Technology in China, and Microsoft Research. Those of us who have been around the Internet for a while, and certainly you and I, Leo, will recognize the play on words where the paper's title was "All Your GPS Are Belong to Us."

Leo: He he he he.

Steve: Yes. It was "all your base"; right? "All your base are belong..."

Leo: "All your base belong to us," yes.

Steve: Yes.

Leo: It was a poorly translated videogame.

Steve: Right. So this is "All Your GPS Are Belong to Us." The subtitle was "Towards Stealthy Manipulation of Road Navigation Systems." So essentially what's happened is, as a consequence of this research, the GPS spoofing state of the art has leapt...

Leo: "Somebody set us up the bomb. We get signal. What? Main screen turn on." This is the game.

[Clip] All your base are belong to us.

Steve: And that became an Internet meme that was quite entertaining for some time. "All your base are belong to us," yes.

Leo: "All your base belong to us." From Zero Wing, a game that came out in 1989.

Steve: Okay.

Leo: Yeah, so it's a pretty old meme.

Steve: Yeah.

Leo: But it's still wonderful.

Steve: It lives on, lives on. So, okay.

Leo: Lives on.

Steve: So, okay. So what these guys did was to take off-the-shelf, readily available equipment. They said: "We show that adversaries can build a portable spoofer" - that is to say a GPS spoofer with low cost, they say about $223 - "which can easily penetrate the car body to take control of the GPS navigation system. Our measurements show that effective spoofing range is 40 to 50 meters, and the target device can consistently latch onto the false signals without losing connections. The results suggest that adversaries can either place the spoofer inside or under the target car and remotely control the spoofer, or follow the target car in real time to perform spoofing."

So the spoofer hardware was four components: A HackRF One based frontend, which is the software-defined radio; a Raspberry Pi; a portable power source, in this case it's a rechargeable battery; and an antenna. It fits in a small box, like smaller than the length of a pen, which they showed in their photo to give some sense of scale, and just uses readily off-the-shelf equipment. What they did, the way they went further than had been before, was that GPS, you know, fouling up GPS has been possible. But essentially replacing valid GPS navigation data with spoofed navigation data which could be acted on had never been done before. And that's what these guys did.

So one can imagine, I mean, we've talked about how, as autonomous vehicles happen, interstate and cross-country truck driving may end up being replaced by automation. And certainly the GPS system would be one of the main signals that these systems would take in. So you could imagine a scenario, and I'm sure it'll be the subject of fiction if it isn't already, well, now it's much less fictional than it was, where a high-value cargo was rerouted, literally, by not coming into contact with the self-driving truck in any fashion, but by spoofing its GPS so that it thought it was somewhere that it wasn't and causing it to take a wrong turn, literally, and then continue on a course to an alternative spoofed site.

That's what these people did. They developed the technology. They came up with a system which, when tested against 40 humans, 38 of the 40 did not detect anything amiss. That is, they were following their turn-by-turn navigation. Everything looked fine. They believed what their navigation system was telling them. And the system managed to bring them to a destination well away from where they were intending to go.

So again, it's not difficult to imagine this being put to various nefarious ends. So the state of the art has jumped from something like where your navigation system is clearly wrong, and many people have had mapping software fail and tell them to drive off the pier, and they say, no, I don't think I want to do that. In this case, it just tells them to take a turn or a series of turns where everything they're doing matches the map that's being seen, but they're not at the destination they believe. So interesting piece of work by those three groups. And they talk about the need to harden GPS and navigation, that is, not to just blindly believe the positioning information that we receive because, as this demonstrates, it can't be believed.

Back in November, GitHub announced a very cool new service for the maintainers of various repositories which they said was going to be applied to Ruby, JavaScript, and Python, although until last week Python hadn't been implemented. It's not clear to me, except it's just a lot of work probably necessary to get it done. But Ruby gems and

JavaScript NPM all had the benefit from last November. And Python just got it. And what it is, is really a nifty service that I was delighted - I missed it last November and didn't pick up on it until just now, where GitHub is now looking at the manifests of the projects in the repositories and scanning them for known vulnerabilities in the dependent software packages upon which the target projects are built.

So there's a tab in GitHub called the Insights tab. And one of the subcategories is Dependency Graph. And what they've been doing for Ruby and JavaScript since November and have now added Python projects is taking responsibility, within reason, for notifying the maintainers of these repositories if at some point in the future a vulnerability is discovered in any of the packages that their projects depend upon.

So, for example, back when they were explaining this in November they wrote: "When GitHub receives a notification of a newly announced vulnerability, we identify public repositories," and then they said, "and private repositories that have opted in to vulnerability detection that use the affected version of the dependency. Then we send security alerts to owners and people with admin access to the affected repositories. You can also configure security alerts for additional people or teams working in organization-owned repositories."

They said: "We detect vulnerable dependencies in public repositories by default." And this looks like they're repeating themselves: "Owners of and people with admin access to private repositories can also opt into vulnerability detection for the repository. For more information, see 'Opting into or out of data use for your private repository.'" Oh, and they did also note that they will never show the information on this dependency graph page to anybody who's not an admin. So you need to be an admin in order to see it.

But they will proactively notify, or if you go to the Insights tab under the Dependency Graph, you will then see, for projects that qualify, a clear security warning if your project depends upon something with a known security vulnerability. And they encourage you there to update yourself to the latest version that has been fixed. So just a note for those who are maintaining projects in GitHub that that's there; and, if you weren't aware, that they just added that feature for Python. And I wasn't aware of it at all, so I'm glad to have found out. Very cool.

Okay. Now Google and Chrome. The claim is that, starting with I think it's Chrome 63, a new capability was added, but was disabled by default. And they call it Full Site Isolation. And this week's news is that Google has enabled this site isolation feature for 99% of Chrome's desktop users. Well, I'm using the latest 67.0 point whatever it was, let's see, 67.0.3396.99, which is current. And in digging into this, I wasn't enabled yet. So our listeners can check. And Leo, you can. If you open Chrome and put chrome://flags into the URL bar, there's just too many of them. So then you need to search, just put the word "isolation" in, which brings up a nice subset of individual flags that Chrome knows about. The first one that comes up is named Strict Site Isolation. And are you disabled on all of those?

Leo: No, it looks like, well, let me look at Isolation first.

Steve: Looks like enabled. Oh.

Leo: Disabled, disabled, no. Here's disabled for strict site isolation.

Steve: Yeah, yeah, yeah. So...

Leo: The Trial Opt-Out is default. The others are default.

Steve: Yeah. So I don't know why, I mean, maybe this is like just about to happen. Okay. So what they're doing is interesting, which is why we're talking about it. We've known that Chrome has adopted a process per tab approach, and that Firefox recently has gone to something similar, the idea being that we're trying to minimize the attack surface. What Chrome is doing is utilizing the operating system's interprocess isolation and allowing the browser to inherit that so that web pages occupying separate tabs are in separate processes. And essentially that means that the renderer for a tab, and that's the thing that reads the page, reads the JavaScript, does everything, it's in a process that has no contact with another tab's process and thus renderer. Whereas in the old days, when you just ran a browser and up came one process, well, inherently there was some risk because all of the tabs were being shown by a single process. And if something could break out of its own tab, potentially any other data currently open in the browser was available.

So first thing that Chrome did was to break out a single process per tab. This goes further. This is a single process per site, meaning per domain. Which means multiple processes per tab if a single tab hosts content from multiple domains. And as we know, many tabs do. If you've got ads in iframes, those are from other domains. Now, the reason this is off is that this incurs a significant overhead. And Google's aware of that. So they're saying between a 10 and 20% memory overhead and lots more processes.

So right now, I mean, maybe they've already backed off of it, which is why by the time I saw this happened it had already been turned off. I don't know. Our listeners can experiment with it if they're interested because it's right there. You put chrome://flags, search for "isolation," and there is Strict Site Isolation. And Chrome says it is a security mode that enables site isolation for all sites. When enabled, each renderer process will contain pages from at most one site, that is to say, one domain, they say, using out-of-process iframes when needed. Again, meaning that, if you have an iframe, the content in that iframe will not be rendered by the page's process.

Chrome will launch a new process containing a web rendering engine to render the contents of that iframe and show it in the tab. So what this does is it really increases security by creating process boundaries, now with subpage process boundaries, but at a potentially significant expense. What's exciting is that, as part of the announcement of them deciding to go mainstream with this - but it was supposed to be Chrome 67. Like today's Chrome is supposed to have this on, and you and me - or you and I, sorry. You and I, Leo, both have them disabled.

Leo: By default, yeah.

Steve: So, yeah. So maybe it's still coming. Maybe they decided to take it more slowly. There is an interesting - down toward the bottom was site isolation, or the second one that comes up is Site Isolation Trial Opt-Out. What Google was going to do, with the instrumentation that they have, was to just sort of turn this on for some people and monitor what it means for the browser in the real world. And so what they did was they offered an opt-out override for people who didn't want to have this turned on for them by Google and suddenly, I mean, I run often with Task Manager open, and I'm like looking at all the Firefox processes, and when I have Chrome open, all the Chrome processes, even when there's, like, not a lot is going on. And so I'm thinking, wow. If you went to a frame-heavy page with this turned on, you probably have to scroll your Task Manager through pages of Chrome.exes in order to see what's going on.

So, oh, what I was going to say was when as part of the announcement they were so bullish about this that they were planning, if this went mainstream, to back out of and remove the Spectre mitigations. That is, their feeling was, if they could break a page apart into per-domain processes, then it's no longer necessary to fuzz the JavaScript timers. And they specifically said that the shared array buffers that we were talking about a couple weeks ago as something, a feature that was not going to be implemented with WebAssem because it allowed high-resolution timers to be crafted by crafty JavaScript, they were saying they're going to put shared array buffers - allow them again, and not worry about Spectre because it was only - apparently it was hostile iframes that they worried could break out of local page containment and affect other and get information from the same page.

So now they're saying, well, if we move forward - and this has like been going on since 63, which is earlier this year. If we move forward and give a process per iframe, then we don't have to worry about hostile iframes and ads and so forth, hosted content from other domains on the main page having that level of granularity and access. So anyway, it'll be interesting to track this.

Again, anybody who's interested, you can turn it on and see what happens. I haven't done that yet. I think I will after the podcast because I'm just curious to see. You have to restart Chrome afterwards so it can come back up in its new mode. Be interesting to see what it means in terms of performance. It's going to slow things down because you're talking about launching processes.

There is some logic in Chrome to recycle existing processes because they recognize it takes a while to launch a process. So rather than, as you move among pages, rather than launching and destroying processes, Chrome probably creates a process pool. But I salute them for the work they're doing. I mean, this is the future of how we stay secure when we're using a browser which is increasingly becoming our portal to the world, and also needing to thwart the bad guys. So the idea that you could use this rather than browser-based Spectre mitigations, which as we're seeing only slow things down, don't really solve the problem, that's really encouraging.

But at this point, anyway, I don't know whether they've already said ouch and backed off, or maybe it was a prospective announcement when they said it was enabled for 99% of Chrome desktop users. Or maybe, Leo, you and I are just a small little 1%.

Leo: I can see why they wouldn't want to enable it, if it's that much of a hit.

Steve: Oh, yeah. I'm terrified.

Leo: I'm surprised that they actually were thinking of enabling it by default.

Steve: Yeah, yeah.

Leo: I mean, you think it's important for security, obviously; right?

Steve: It would definitely enhance security, yes. And I have a link to a support page in the show notes here under Google Chrome Help. It says: On your computer, open Chrome. In the address bar at the top, enter chrome://flags/#enable-site-per-process, that's hyphenated, and press Enter. Next to Strict Site Isolation, click Enable. If you don't see Strict Site Isolation, update Chrome, meaning if you don't have the latest, and you and I both do, Leo. And then click Relaunch.

And so under Threat Model they said: "For a 'one-site-per-process' security policy, we assume that an attacker can convince the user to visit a page that exploits a vulnerability in the renderer process, allowing the attacker" - and remember, the renderer is a massive interpreter - "allowing the attacker to run arbitrary code within the sandbox. We also assume that attackers may use speculative side-channel attacks, for example Spectre, to read data within a renderer process. We consider attackers that want to steal information or abuse privileges granted to other websites."

And then they finally said, under Requirements for this mitigation, they said: "To support a site-per-process policy in a multiprocess web browser, we need to identify the smallest unit that cannot be split into multiple processes." That is, they don't want to break anything. They said: "This is not actually a single page, but rather a group of documents from the same website that have references to each other. Such documents have full script access to each other's content, and they must run on a single thread, not concurrently. This group may span multiple frames or tabs, and they may come from multiple subdomains of the same site."

So they are saying that you could have multiple tabs from the same site, that is, from the same domain. Those would be safe to share a process. But so they're sort of refactoring the processes and the tabs so that multiple processes might be using the same page but would not be sharing the same renderer, which seems like a good thing. It'll be interesting to see whether this appears in the future, or whether it's just something that they were experimenting with, but when they actually ran real-world instrumentation turned out to be too expensive.

Okay. So I've been saying for, well, for as long as we've been talking about Universal Plug and Play problems with routers and the fact that routers are now being used increasingly as routing nodes on the Internet, that we're all sort of, well, not we all, but those who are blindly ignorant of the fact that their routers have been commandeered and compromised are lucky so far that the hackers are more interested in outward facing DDoS redirections and spoofing and using the routers to anonymize their source than they are interested in what is going on on the LAN side behind the router.

Well, it turns out that at least one hacker did get curious. The security firm Recorded Future discovered sensitive military documents being offered for sale on various hacker forums. Bleeping Computer in their coverage of this reported that some of the sensitive documents put up for sale, and this is from what Recorded Future had said, include maintenance course books for servicing the MQ-9 Reaper drones, various training manuals describing deployment tactics for improvised explosive devices, an M1 Abrams tank operation manual, a crewman training and survival manual, and a document detailing tank platoon tactics. And the hacker was asking between 150 and $200 for these, which was considered, like, almost nothing, given how sensitive some of this information was. Recorded Future said that it engaged the hacker online and discovered that he used Shodan to hunt down Netgear Nighthawk R7000 routers that are known to use a default FTP password, believe it or not, Leo.

Leo: Admin/admin.

Steve: Ugh. So the people using these routers - and oh, by the way, there's 4,000 of them - turned on the FTP server, exposed it to the WAN, and didn't change the default login. So the hacker used the default FTP password to gain access to some of these routers whose owners had not bothered to change the default. Based on the documents and the details he shared online and with the researchers in private conversations, that is, with Recorded Future, one such location was the 432nd Aircraft Maintenance Squadron Reaper AMU OIC, whatever that is, stationed at the Creech Air Force Base in Nevada.

Here, writes Bleeping Computer, "he used his access to the router to pivot inside the base's network and gain access to a captain's computer, where he stole the MQ-9 Reaper manual and a list of airmen assigned to Reaper AMU," which must be some sort of maintenance unit. "The MQ-9 Reaper drones are some of the most advanced drones around and are used by the U.S. Air Force, the Navy, the CIA, Customs and Border Protection, NASA, and other militaries of other countries." So anyway, as I said, the routers have default credentials. When Netgear was notified of this two years ago, in 2016, they responded by putting up a support page with information on how users could change their router's default FTP password. Which boggles the mind. I mean...

Leo: They've soldered it in or something.

Steve: Just incredible. Oh, boy. Again, I'm gobsmacked, as they say in the U.K., that first of all, I mean, I have to hold Netgear responsible for not somehow using a random admin and password which could have just, I mean, like the first time the router comes up it could see that they're blank and just use random stuff for the admin and password fields, which the user could copy and then use if they chose, or change to something that they want them to be if they don't like random gibberish. But to have them start up in the firmware as statically global known things, I mean, it's just...

Leo: Most routers do that, though. Like just for users, admin/admin.

Steve: I know. I know. I mean, and this is a practice that just has to change. The firmware should see that they've not been initialized when it comes up. And then just based on - it could use packet noise. I mean, there's plenty of sources of entropy available to a router. So, I mean, now a lot of the new - these routers typically use ARM chips, and they've got good random number-generating hardware in them. So just pull some random numbers. Put gibberish in there that will annoy the user. Well, which, first of all, the user could copy and use, which would be very good username and passwords, which would never occur again. Or if the user doesn't like them, then they can change them to be what they want. But just the idea that they're just, I mean, it's one thing to have the LAN side admin default to admin/admin. But to have the WAN side ever default to anything is unconscionable from a security standpoint.

Leo: If you defaulted to WAN Administration Off, that would do it; right?

Steve: Well, but clearly these people wanted remote access.

Leo: Oh, they turned it on; right.

Steve: They turned it on.

Leo: C'mon, though. You've got to give the user some responsibility. If you're going to turn on WAN access and leave the default password, you're kind of at fault; right?

Steve: I don't disagree, except that defense in depth is the goal here.

Leo: Right. The router needs to help you, yeah.

Steve: Yeah. Why not? Why not have it do everything it can?

Leo: I'll tell you why not.

Steve: I mean, it's just so trivial.

Leo: Yeah, but you know why not. Because Netgear doesn't want all those calls from people saying, "Hey, I didn't write down that password." Right? I mean, I've reset routers many times because I forgot the administrative password.

Steve: Yeah. Which is a good thing.

Leo: Yeah.

Steve: Because that means that you didn't use the default.

Leo: Yeah, right. That's true. I guess it could do that, though, every time it's reset - generate a new set, new password, using your method. Yeah, that would be a good thing to do. It's just a lot of coding; you know? That's work.

Steve: Yeah. Oh, darn. Right.

Leo: Oh, darn.

Steve: Oh, darn. Okay. So when BGP abuse becomes sufficiently blatant, even the reluctant to respond with anything anybody could call a kneejerk reaction, Internet managers finally do. So we've talked about BGP routing in the past, and we've talked about mistakes that do happen from time to time. As we know, the big routers on the Internet connect to multiple other routers, and they build a routing table which instructs them how to forward incoming packets bound for, like, trying to go out another one of their connections. So what happens is, when the router has a table of routes that it knows how to forward packets for, it advertises that in this Border Gateway Protocol, BGP. So that allows the other routers it's connected to to know that, oh, if the packet is bound for somewhere else, where should they forward their traffic to?

So this is the way that the Internet self-organizes, and it's brilliant. And what it really does is it actually organizes itself, which thank goodness we have technology to do that. Otherwise it would always be broken and out of date and so forth. So when someone hooks up a new IP subnet to a router, they say, okay, this range of IPs is going to be connected to this interface that goes off to this customer. So that update propagates through. Using BGP, it propagates far enough out until that subnet becomes incorporated by a larger subnet that means that the route doesn't need to propagate any further. In other words, and we've talked about this, how the classless interdomain routing, where you look at an IP address as a prefix and a suffix, where you can route everything with the same prefix to the same destination, hugely simplifies this task.

Okay. So we're only human, we people who maintain routers on the Internet, and from time to time a mistake gets made. So when that happens, somebody entering into the router which range of IPs this new interface should be connected to might put a typo in. And the router doesn't know it's a typo. It says, oh, okay, I'm now in charge of this block of IPs. They go out that interface. So just as happens when it's the right news, it happens when it's the wrong news. And so suddenly an advertisement, as these things are called, these updates are called, an advertisement goes out, advertising that it is responsible for a block of IPs it is not actually in charge of.

And there's no, like, overlord. It's all sort of a peer agreement system where everybody hopes not to do any typos, and things more or less work. Except they don't. Then something breaks. And then suddenly all of the Internet gets routed to some one location that melts down because it's way too much traffic for it, and they go whoops, and they remove that mistake from the routing table, and then the Internet repairs itself, and we go back to business as usual.

Unless it's done deliberately. And there's nothing to prevent this being done deliberately. As we've discussed over the years, there's a lot of the Internet that is still dark. There are a lot of people squatting on IP allocations that they may grow into over time. There have been some great companies, like Hewlett Packard, that had three, I think it was, Class A networks, and gave a couple of them back, thus releasing, what is it, 16 million IPs back to the world, essentially, for reallocation, and things like that. Or companies that have been able to, you know, that were sitting on a Class A and realized, you know, we're never going to need 16 million. So they pushed themselves all down toward one end and then gave up half of their allocation. Thank you very much.

But what there still are, are lots of small little bits all over the place, which nobody - the owners don't want to give up because they might legitimately grow into them. And they're not making anymore IPv4 IPs, as they say, so people are holding onto the ones they've got.

Okay. So it turns out that for years a BGP provider, an exchange provider in Portugal named Bitcanal has been reselling, for lack of a better term, profiting from certainly, IPs that it doesn't own. It's been advertising, deliberately advertising small blocks of IPs that its router then says it owns. And because they're small, they tend not to get coalesced with other prefixes - that's why I mentioned this before - which allows them to stay separate as they propagate because they're just small little /24 networks that don't stand out very much. And also they sort of tend to stay off of people's radars that way. They're just little bits of space that are slack in the system that everybody knows about, but they're sort of where they're supposed to be.

So NANOG, N-A-N-O-G, is the North American Network Operators Group. And Ronald Guilmette started off his posting to the NANOG mailing list saying: "I mean, seriously, WTF?" And then he went on to say: "As should be blatantly self-evident to pretty much everyone" - and Leo, if you're at this place in the show notes, I have a Pastebin link that's worth clicking on just to show these are the current misallocation of IPs by this

Portugal-based, this Bitcanal ISP. These are the IPs that they have stolen from the Internet. Statically, they have them.

He says: "As should be blatantly self-evident to pretty much everyone who has ever looked at any of the Internet's innumerable prior incidents of very deliberately engineered IP space hijackings, all of the routes currently being announced by AS3266" - that's the Autonomous System number of Bitcanal. And he says, parens: "(Bitcanal, Portugal) except for the ones in 213/8" - and so that's the one network that they are entitled to - he says: "...are bloody obvious hijacks."

Leo: Wow.

Steve: He says: "That's 39 deliberately hijacked routes, at least going by the data visible on bgp.he.net. But even that data from bgp.he.net dramatically understates the case, I'm sorry to say. According to the more complete, up-to-the-minute data that I just now fetched from RIPEstat, the real number of hijacked routes is more on the order of 130 separate hijacked routes, for a total of 224,512 IPv4 addresses," and then the Pastebin link.

He says: "In simpler terms, Bitcanal has made off with the rough equivalent of an entire /14 block of IPv4 addresses that never belonged to them. And of course they haven't paid a dime to anyone for any of that space." Anyway, he alleges, and I think it's pretty clear it's the case, that Bitcanal is doing all of this, this hijacking of BGP routes, for the purpose of reselling, and in fact they know for a fact, reselling the hijacked IP addresses to spammer groups.

Leo: Yeah, of course.

Steve: Which in turn use them - yup.

Leo: Because they don't care if the address stops working after a while.

Steve: Exactly. They just want IPs that are not in the current blacklists. And so this allows them to establish connections to SMTP servers from unknown IPs and to stay hidden that way. So anyway, the upshot is this finally rose to the level where - and again, it's just like we've seen with Google making moves very deliberately, but also in a very considered fashion, or the larger industry pulling a certificate authority's rights to sign. I mean, you have to really be bad in order to have the industry say, okay, look. You made a mistake. You didn't report it. You lied about it when we told you. Then you didn't tell us about the other things that you found out that you also did, blah blah blah blah blah. I mean, we've talked about this through the years. What finally happened was they went dark. The Internet said goodbye, and basically no longer routes any traffic to that Portugal-based ISP, nor accepts any from them. They are out of business as a consequence. And maybe we'll see a drop in spam, which would be a nice side effect of that.

Leo: Yeah.

Steve: And I did finally want to finish up, I wanted to follow up from a comment last week. John Doe, as he described himself, writing from Moonbase 17 - which I don't think is actually a place. The subject was "MDADM RAID 6 SpinRite Comment." This was sent on the 12th of July. He said: "Hopefully this gets to you in time, or you can forward it to the person that asked the question." Or I can share it on the podcast so that all of our other listeners who may find this interesting, and actually it is kind of cool, may benefit, as well.

He says: "On your last podcast (SN-671) there was a question about running SpinRite on a five-disk RAID 6 with MDADM on Linux." And get this. Here it is. "If the person turns on write-intent bitmaps, then the rebuild time can often be reduced to a few minutes when a drive is put back into the RAID." He says: "I see five- to 10-minute resyncs." I have a link in the show notes to the raid.wiki.kernel.org that describes it. And in that write-up they said: "When an array has a write-intent bitmap, a spindle," they said, "a device, often a hard drive, can be removed and re-added. Then only blocks changed since the removal, as recorded in the bitmap, will be resynced."

And then they add: "Therefore a write-intent bitmap reduces rebuild/recovery time if the machine crashes with an unclean shutdown; or one spindle is disconnected, then reconnected." And then they go on to say: "Write-intent bitmap support is only available for RAID geometries causing data redundancy. For example, as RAID 0 has no redundancy, it cannot be inconsistent. So there is nothing to record in such a bitmap."

But anyway, this was very cool. What it means is that, while the drive is offline with SpinRite running on it, the RAID can continue to stay up. It can continue to function. And when the SpinRited drive rejoins the RAID, the write-intent bitmaps will be compared, and essentially an incremental update will be made to the rejoined drive that only takes a few minutes and avoids a full RAID rebuild. So thank you very much for the tip. And for anybody who's interested, it makes it easy then to pull a drive out, let SpinRite run on it, even SpinRite 6 as it is now, and then have it rejoin the RAID later without a great impact to the RAID.

Leo: That's cool. I didn't know about that. That's really fast, yeah. Mr. "Zero Trust" Gibson is here.

Steve: Yes. Leo, you would have fun pronouncing the Russian names of these agents.

Leo: I love it. I love it.

Steve: Yes. I, however, am unable to do that.

Leo: Viktor Borisovich Netyksho. Boris Alekseyevich Antonov. Dmitriy Sergeyevich Badin. I love it. You're right. I am having fun.

Steve: I knew you would. The good news is their first names are almost unique. There's Victor, Boris, Dmitriy, Ivan, Aleksey, Sergey, Nikolay, Pavel, Artem, Aleksandr, Aleksey, and Anatoliy.

Leo: Sounds like the bridge of the Starship Enterprise.

Steve: Well, there are two Alekseys. So there's a little bit of - there's Aleksey Lukashev.

Leo: Aleksey, yeah, Lukashev, yeah.

Steve: Lukashev and Potemkin, yes. So other than that, their names are unique. And as I said at the top of the show, I was going to talk about, I was going to title this podcast "Spectre 1.1 and 1.2" or "Spectre Keeps Delivering" or something. But I read the 29-page indictment, and I was just stunned by the level of detail that it alleges about the behavior of these agents. And, I mean, it's amazing. And so...

Leo: I think there are addresses. The unit they're in command of. What they did?

Steve: Yes, yes.

Leo: It feels like we had a tap in here; right? It feels like that.

Steve: Well, the only thing I could think is that, to be able to do this retrospectively, the NSA, and I guess it makes sense, that's what the huge server farm, the hard drive farm in Utah is doing is literally recording all traffic, maybe not interstate, but international. That is, it must be that the trunks coming in and out of the United States are just - they're just being recorded.

Leo: Which, by the way, I just want to point out, there's a reason why there's all this detail in the indictments. It's just a little warning shot, isn't it, for anybody else who might be involved, that we know everything.

Steve: Yes. So as I'm reading this, I'm just - I'm stunned by the level of what is understood. First of all, it starts out with those 12 people. It names them each and describes their job functions and their lines of reporting and which aspects of the hacking they were involved in. So it sort of gives you a per-defendant overview of who did what.

Leo: We rarely see this kind of detail when you hear about a hack. I mean, they talk about exactly how they executed it. It's amazing.

Steve: Yes. And I highlighted - I didn't want to go through - I got overly ambitious as I [crosstalk]...

Leo: [Crosstalk] the entire indictment into the show notes.

Steve: I know. I was trying to skip things. And I won't drag our listeners through it. But, for example, well, actually it reads: "For example, on or about March 19, 2016, Lukashev" - and I don't remember his first name. He was, where is he, Aleksey...

Leo: Lukashev, not Potemkin. Get the right Lukashev. Get the right Aleksey.

Steve: Right, right. He's the - we have to disambiguate him.

Leo: Yes.

Steve: Created and sent a spear - now, get this. On March 19th this guy in Russia "...and his co-conspirators created and sent a spearphishing email to the chairman of the Clinton Campaign. [Aleksey] used the account 'john356gh' at an online service that abbreviated lengthy website addresses (referred to as a 'URL-shortening service'). [Aleksey] used the account to mask a link contained in the spearphishing email, which directed the recipient to a GRU-created website. [Aleksey] altered the appearance of the sender email address in order to make it look like the email was a security notification from Google (a technique known as 'spoofing'), instructing the user to change his password by clicking the embedded link. Those instructions were followed. On or about March 21, 2016, [Aleksey] and Yermakov" - I forgot his first name - "and their co-conspirators stole the contents of the chairman's email account, which consisted of over 50,000 emails."

And it goes on like that. On or about March 20. We have a March 19th. "March 25th [Aleksey] used the same john356gh account to mask additional links included in spearphishing emails sent to numerous individuals affiliated with the Clinton Campaign, including Victims 1 and 2." And those were earlier described in the indictment. "[Aleksey] sent these emails from the Russia-based email account hi.mymail@yandex.com that he spoofed to appear to be from Google."

And then I've jumped down again, and I highlighted something else: "On or about April 6 the Conspirators created an email account in the name (with a one-letter deviation from the actual spelling) of a known member of the Clinton Campaign. The Conspirators then used that account to send spearphishing emails to the work accounts of more than 30 different Clinton Campaign employees. In the spearphishing emails, [Aleksey] and his co-conspirators embedded a link purporting to direct the recipient to a document titled 'hillary-clinton-favorable-rating.xlsx.' In fact, this link directed the recipients' computers to a GRU-created website." And it goes on like that.

At one point - and they're not clear here about the network penetration. They said: "Beginning in or around March 2016, the Conspirators, in addition to their spearphishing efforts, researched the DCCC and DNC computer networks to identify technical specifications and vulnerabilities." Now, okay. Think about that. These guys, in addition to the spearphishing, researched the computer networks to identify technical specifications and vulnerabilities. How do we know?

I mean, the level of detail here demonstrates we do, as they're about to explain. But it's amazing to me because this wasn't, I mean, this is not something that - we know from the reporting and actually from what's in this indictment that it wasn't until some time later that the presence of malware was discovered; and then Company 1, as it's called in this indictment, was a security firm that was brought in to figure out what was going on. This Company 1 almost cleaned things up but left behind this X-Agent malware in a Linux machine that allowed them to maintain a persistent presence in the network.

What they said, for example: "For example, beginning on or about March 15 Yermakov" - I guess that's still [Ivan] - "ran a technical query for the DNC's Internet protocol configurations to identify connected devices." Now, I can't technically disentangle what that actually means, but that means something. On or about the same day this Yermakov, [Ivan], searched for open source information about the DNC network, the Democratic Party, and Hillary Clinton.

So I think in this case they don't mean open source software, they mean like did Google searches. And in fact there are instances later where they list the search terms that were used in their searches before, as I mentioned at the top of the show, generating a blog post under this Guccifer 2.0 moniker. On April 7 he ran a technical query for the DCCC's Internet protocol configurations to identify connected devices. And what we do know is that in or around April, within days of [Ivan]'s searches regarding the DCCC, the Conspirators hacked into the DCCC computer network. Once they gained access, they installed and managed different types of malware to explore the DCCC network and steal data.

On April 12 they used the stolen credentials of a DCCC employee, who is identified here as DCCC Employee 1, to access the DCCC network. Employee 1 had received a spearphishing email from the Conspirators on April 6 and entered her password after clicking on the link. Between in or around April 2016 and June 2016 the Conspirators installed multiple versions of their X-Agent malware on at least 10 DCCC computers, which allowed them to monitor individual employees' computer activity, steal passwords, and maintain access to the DCCC network.

Now, probably some of this came from that Company 1, the security firm that was brought in while this was ongoing. And so I'm sure that the DoJ researchers contacted them and said, okay, tell us everything you know about what you found when you went over to the campaign networks and looked around. But anyway, this goes on like that. There's multiple instances of malware. There is an instance where somebody, I think it was on the DNC network, had access to the DCCC network. They installed keystroking and screenshot functions. Oh, yeah, here.

"X-Agent malware implanted on the DCCC network transmitted information from the victims' computers to a GRU-leased server located in Arizona." So they weren't even exfiltrating directly from the U.S., but going to another server in Arizona. There's also one, I think it's in Indiana. "The Conspirators referred to this server as their 'AMS' panel. Kozachek, Malyshev, and their co-conspirators logged into the AMS panel to use X-Agent's keylog and screenshot functions in the course of monitoring and surveilling activity on the DCCC computers. The keylog function allowed the Conspirators to capture keystrokes entered by DCCC employees. The screenshot function allowed the Conspirators to take pictures of the DCCC employees' computer screens," as we understand all that.

They also used a relay known as the "middle server" that was located somewhere else. I've skipped over that. I'm just looking for anything else that's particularly interesting, I mean, in terms of details. They were proactively covering their logs and cleaning things up. They even used CCleaner, Leo, at one point in an attempt to clean off evidence of their activity.

Leo: Well, there's their problem.

Steve: Oh, yeah, Illinois. On or about April 28 they connected to and tested a computer located in Illinois using what was described as X-Tunnel. So it sounds like they had some sort of a VPN or an encrypted tunneling system. They also used, sort of reverse-engineering some of the nonspecific language in the document, they used an open source compression tool to archive in some cases gigabytes' worth of documents and then exfiltrated them through this X-Tunnel software which was installed on the machine in Illinois in order to take it out.

Oh, and get this. And, like, how do they know? Between on or about May 25 and June 1, so a period of, what, about a week, the Conspirators hacked the DNC Microsoft Exchange

Server - and that's understandable, that is, how they might know that from records, and stole thousands of emails from the work accounts of DNC employees. But here it is. During that time [Ivan], who's very busy, researched PowerShell commands related to accessing and managing Microsoft Exchange Servers. So as I said, we really were all up in their business in order to have this level of detail of the things they were doing behind the scenes on their end to prepare for what we know they were doing at our end, apparently based on logs that somebody had somewhere. Because it sure seems like the Democratic election group were clueless about a lot of this. Deleted logs.

"Despite the Conspirators' efforts to hide their activity, beginning in or around May of 2016 both the DCCC and DNC became aware that they had been hacked and hired a security company, Company 1 [as it's referred to here] to identify the extent of the intrusions. By in or around June, Company 1 took steps to exclude intruders from the networks. Despite these efforts, a Linux-based version of X-Agent, programmed to communicate with the GRU registered domain linuxkrnl.net" - that's krnl.net - "remained on the DNC network until in or around October of 2016."

And they talk about keystroke logging, tons of snapshots. They then created, it says, oh, they even found them - they know when they were unable to get something that they wanted, suggesting that it wasn't just monitoring DNC networks. In or around September of 2016 they "successfully gained access to DNC computers hosted on a third-party cloud computing service. These computers contained test applications related to the DNC's analytics. After conducting reconnaissance, the Conspirators gathered data by creating backups or snapshots of the DNC's cloud-based systems using the cloud provider's own technology. They then moved the snapshots to cloud-based accounts they had registered with the same service, thereby stealing the data from the DNC."

And also: "More than a month before the release of any documents, the Conspirators constructed the online persona DCLeaks to release and publicize stolen election-related documents. On or about April 19, 2016, after attempting to register the domain electionleaks.com, the Conspirators registered the domain dcleaks.com."

So, okay. How can anyone know that they attempted but failed to register the domain electionleaks.com? It's mindboggling. So anyway, they got dcleaks.com "through a service that anonymized the registrant." And we don't know if that's Tor or what, but maybe. "The funds used to pay for the dcleaks.com domain originated from an account at an online cryptocurrency service that the Conspirators also used to fund the lease of a virtual private server registered with the operational email account dirbinsaabol@mail.com." The dirbinsaabol email account was also used to register that john356gh URL-shortening account used by our friend Aleksey to spearphish the Clinton Campaign chairman and other related individuals.

And this is something we've seen before where any linkages between what would appear to be otherwise separate events, any reuse of IPs, of email accounts, of obscure domains and so forth, that can be used if you have, like, encyclopedic knowledge of every packet that transited the Internet during that time. You can figure all this out. But I don't know how you do this, short of having that kind of visibility into Internet traffic. To me, this suggests there is a level of surveillance far in excess of what I at least have assumed was technically feasible. It's amazing.

Leo: Does it give you a little cause for concern? Because...

Steve: Well, I would be concerned if I was Aleksey. I'd be looking over my shoulder.

Leo: Yeah, but, I mean, do you think they have this level of detail about everything that's happening?

Steve: You know, it certainly does suggest that the future will be interesting.

Leo: Yikes.

Steve: Yeah, wow. On or about June 14 the DNC through Company 1, remember, that's the security company they brought in, publicly announced that it had been hacked by Russian government actors. In response, the Conspirators created the online persona Guccifer 2.0 and falsely claimed to be a lone Romanian hacker to undermine the allegations of Russian responsibility for the intrusion. On June 15 the Conspirators - here again, get this - logged into a Moscow-based server used and managed by Unit - and I haven't talked about the unit numbers. There's two different unit numbers, 25 something or other, but this one is 74455. And we know the street addresses and like where the nearest Starbucks is. It's just amazing.

Between 4:19 p.m. and 4:56 p.m. Moscow Standard Time, after just having gotten some lattes, Unit 74455, they searched for certain words and phrases between 4:19 p.m. and 4:56 including "some hundreds of sheets, dcleaks, illuminati, worldwide known, think twice about, and company's competence." Later that day, at 7:02 p.m. Moscow Standard Time, the online persona Guccifer 2.0 published its first post on a blog site created through WordPress titled "DNC servers hacked by a lone hacker." The post used numerous English words and phrases that the Conspirators had searched for earlier that day, which is to say between 4:19 p.m. and 4:56 p.m.. Three hours earlier they made sure their English was correct when they made the Guccifer 2.0 posting alleging that, no, I'm just some guy in Romania.

This says the Conspirators conducted operations as Guccifer 2.0 and DCLeaks using overlapping computer infrastructure and financing. In other words, again, there were collisions. There is a - I skipped some of the details, but here's one: "To facilitate the purchase of infrastructure used in their hacking activity, including hacking into the computers of U.S. persons and entities involved in the 2016 U.S. presidential election and releasing the stolen documents, the Defendants conspired to launder the equivalent of more than $95,000 through a web of transactions structured to capitalize on the perceived anonymity of cryptocurrencies such as bitcoin."

And I'll skip a bunch of this because there's a whole bunch of bitcoin stuff. But for example: "The Conspirators used several dedicated email accounts to track basic bitcoin transaction information and to facilitate bitcoin payments to vendors. One of these dedicated accounts, registered with the username 'gfadel47,' received hundreds of bitcoin payment requests from approximately 100 different email accounts. For example, on or about February 1, 2016, the gfadel47 account received the instruction to 'please send exactly 0.026043 bitcoin to' a certain 34-character bitcoin address. Shortly thereafter, a transaction matching those exact instructions was added to the blockchain." So again, this is where we see bitcoin payments not actually being untraceable and anonymous if you have complete vision, complete sight of everything going on.

Anyway, and I finished, the last piece of this I grabbed was: "On occasion, the Conspirators facilitated bitcoin payments using the same computers that they used to conduct their hacking activity, including to create and send test spearphishing emails. Additionally, one of these dedicated accounts was used by the Conspirators in or around 2015 to renew the registration of a domain" - that's that linuxkrnl.net - "encoded in certain X-Agent malware installed on the DNC network."

So we know a few things after a reading of this indictment. We know that standard kind of like off-the-shelf in the sense that we've talked about it on this podcast for years, social engineering attacks, spearphishing, email spoofing, link following was the way they got in. They researched people to design email campaigns which would be convincing. They managed to get some unwitting DNC employee to click a link which was pretending to be a Google security advisory, please reenter your password in order to prove you're you. That allowed them to get the person's password, and then they were able to get into the network. And once there, they transferred remote access trojans onto the network, and then from that point spread onto at least 10 different machines.

They were somewhere else I didn't read here, but it's in the show notes if anyone cares. But it's in the - I have a link, by the way, to the PDF. It's 29 pages, the entire indictment, if anyone is interested. But they know how many machines in each of the two networks, the DCCC and the DNC, there was malware found on. They established a presence in the U.S. where they had other staging machines. They used bitcoin in order to register domains and to pay for the various services that were available, used anonymizing services, used VPN or some sort of a tunneling system in order to get the data out.

So all of that is sort of by the book. I mean, this is the danger that any organization is in that is targeted. And that is, you know, we talked about it with the Sony hack, where one executive assistant clicked the wrong link, and that's all it took in order to set up an advanced persistent threat inside of Sony and then cause all the damage that they did. So the human factor is still the weak link. So there's that.

So it was interesting to read this indictment from the standpoint of what exactly was done. Is there anything new or surprising or, like, what, how did that happen? And from the angle of what they did in order to perpetrate this intrusion, no. We learned nothing. In fact, we know the names of these things that are anonymized in this indictment, pretty much. What is shocking is what we know. And Leo, as you said...

Leo: Total information awareness. Remember that?

Steve: Yes.

Leo: James Clapper.

Steve: Yes. It is just astonishing. And, whoa. So, yeah, it has to be that last Friday, when this went public, the GRU agents in these two units said, whoa. First of all, their mistakes were highlighted. The instances where any overlap in infrastructure or email accounts, I mean, anything they reused was linked back together in order to build this web and build this case. And really that's what that, even if we can't read the labels on the diagram which is our Picture of the Week, that's what that is. That's a web of interconnection that demonstrates what the various actors were doing, what roles they had, what they registered and when and how they paid for it and, I mean, it's just like, wow.

Leo: I think you could make the case, though, that while we have this total information awareness, it took a special prosecutor and the Mueller investigation to take these disparate silos of information and, as you say, knit them together.

There was a great movie with Sean Connery called "The Anderson Tapes." He plays the ringleader of a gang that robs a whole apartment building. And there's recordings. There's phone taps. They have the whole story. But at the end of the movie all the agencies with all the illegal taps and all the information decided, rather than prosecute, just let's erase all the tapes. And the movie ends with the tapes being erased because no - but, see, so I have a feeling that we could do this with almost any situation if you could get somebody with the extra legal ability and powers to say I need this, I need this, I need this. That's why you get a hundred subpoenas at a time; right?

Steve: Yeah. Well, and as I said, I'm sure that the company that was brought into the DNC, the so-called Company 1, I'm sure they've turned over all the records they had of what they found.

Leo: There were 140 servers. There was cloud. But there's more than that. There's clearly surveillance; right?

Steve: Because, Leo, retrospectively, that's just it. It's like, I mean, they were talking about the way this began, not the way it ended.

Leo: Right.

Steve: Which is like, oh.

Leo: They knew how they did it.

Steve: Yes.

Leo: But, I mean, I guess you leave breadcrumbs. If you knew where to look, you'd see it, especially if you're recording everything.

Steve: Well, but also to know that they logged into their own machines to perform searches of English phrases, that suggests we're up in their business.

Leo: Yes, we're up in their business, yeah.

Steve: Yeah.

Leo: Pretty much, yeah. The people reading this most closely are the GRU; right?

Steve: Oh, yeah. They're thinking, time to stop using Windows.

Leo: Right. By the way, during this very show I want to say congratulations to Semstix. He's a geocacher, and he found our TWiT geocache. His comments, from today, "Found while listening to Security Now! live. Glad to see Leo is a geocacher. TFTC."

Steve: Very cool.

Leo: So he's listening. Our geocache is outside in the parking lot. I didn't see him do it. But he must have gone and found the geocache. He was listening on headphones live. So I presume you still are, Semstix. Enjoy.

Steve: You know, and this does - it's hard for law enforcement, after seeing this, to argue that they don't have sufficient visibility as it is now.

Leo: Oh, yeah.

Steve: I mean...

Leo: Oh, yeah. No, they didn't go dark. Oh, no.

Steve: Yeah, it's like, where's the darkness? I don't see any darkness here.

Leo: Did I tell you we had Phil Zimmermann on a couple weeks ago on Triangulation, the creator of PGP.

Steve: Yeah.

Leo: And he said this is like - and by the way, it's called Zimmermann's Law that, as technology advances, the ability to surveil advances at the same rate. He says it's like they have a giant big screen TV, and they can see everything. There's just one or two dark pixels in encrypted phones, things like that. And they don't like any, you know, they don't want any black pixels.

Steve: Actually, that's a great analogy.

Leo: Isn't that a good analogy?

Steve: That really is a good analogy, yes.

Leo: Yeah. Clearly, there's little they're missing.

Steve: Wow.

Leo: What a great piece. Thank you, I appreciate your talking about this. Very interesting.

Steve: Well, again, it would have been easy to gloss over it and say, oh, they know a lot. But I thought - what hit me was the level of detail and the dates and the times. Oh, yeah, you know, in the afternoon they did this, and then that evening, three hours later, they did that. It's like, whoa, okay.

Leo: One story that you did miss, but I just wanted to mention, Election Systems & Software is a company that maintains the counting stuff at the county seats of all the voting. They've been putting pcAnywhere on their counting machines.

Steve: Oh, I know, the remote access. I just saw that. Yeah.

Leo: Ron Wyden, Senator Wyden sent them a letter in April saying, "Is there remote access software on our election boxes?" Oh, yeah. Starting in 2000 in a small number of cases. A small number of cases.

Steve: Oh, goodness, yeah. What could possibly go wrong?

Leo: Well, and then I didn't remember this, but apparently pcAnywhere's source code had been hacked around that timeframe, even though Symantec didn't tell anybody till six years later. So...

Steve: Yeah.

Leo: Yeah.

Steve: Yeah.

Leo: It had been stolen years earlier. Steve, you did it again. Thank you, sir. I'm sure there will be more to talk about next week, as we continue to get all up in their business.

Steve: We are never running out of things to talk about, Leo. My pleasure, and we'll see you next week.

Leo: Yes, sir. This show airs - we do it live. You can watch us make it live. Even if you're geocaching, you can listen as we're doing it live. Every Wednesday, I'm sorry, Tuesday, 1:30 Pacific, 4:30 Eastern, 20:30 UTC. Join us in-studio if you wish. We have a visitor from Northern Ireland here today. It's nice to have you. All you've got to do is email tickets@twit.tv. Or no matter where you are in the world, you can go to our live streams at TWiT.tv/live. There's audio and video.

After the show is over, our editors get to work, and Steve posts an audio version of the show, followed a few days later by a transcript written by Elaine Farris, she does a great job, so you can read along as you listen. We have audio and video at our site, too. Steve's site is GRC.com. Go there, by the way, not only to get the podcast, but to get SpinRite, the world's best hard drive recovery and maintenance utility. GRC.com. And there's all sorts of other stuff there, too. It's really a treasure trove. It's like being inside Steve's brain.

Our site is TWiT.tv/sn. And of course it's on every podcast application. You can even listen on your Amazon Echo or your Google Home. Just ask for Security Now!, you'll get the most recent episode. Thank you, Steve. We'll see you next week.

Steve: Pleasure, my friend. Till then.