SECURITY NOW!

Transcript of Episode #642

BGP

Description: This week we examine how Estonia handled the Infineon crypto bug; two additional consequences of the pressure to maliciously mine cryptocurrency; zero-day exploits in the popular vBulletin forum system; Mozilla in the doghouse over "Mr. Robot"; Win10's insecure password manager mistake; when legacy protocol come back to bite us; how to bulk-steal any Chrome user's entire stored password vault; and we finally know where and why the uber-potent Mirai botnet was created, and by whom. We also have a bit of errata and some fun miscellany. Then we're going to take a look at BGP, another creaky yet crucial - and vulnerable - protocol that glues the global Internet together.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-642.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-642-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk about the strange case of the Estonian ID cards. A weird bug or actually flaw introduced into Microsoft's Windows 10. We're still not sure exactly who got it and why. We'll also talk about the case of the Firefox plugin promoting "Mr. Robot," and a whole lot more, all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 642, recorded Tuesday, December 19th, 2017: BGP.

It's time for Security Now!, the show where we cover your security online with this guy right here, the Explainer in Chief, Steve Gibson. Happy holidays, Steve.

Steve Gibson: Hey, Leo. I get to say now that this is the penultimate episode of 2017.

Leo: It is indeed.

Steve: Which we've all learned is not the last one, but it's the second to last one. So, yeah. Next week we're going to do a fun "blast from the past" special for people who haven't listened for - I think it was seven years ago that we discussed something. We'll let it be a little bit of a surprise for anyone who has joined us in the last seven years, missed this particular episode that we're going to repeat. Not the Portable Dog Killer. Not any of the other things that we've repeated before. This is a first-time repetition. So I think it's going to be good.

But this week, for the penultimate episode, I want to talk about something that we've touched on by necessity through the years in the 12-plus years we've been doing this. Wait, now, we're in year 12, so 11-plus years. And that is BGP, the Border Gateway Protocol. It's in the news because of a recent mistake. Well, no, actually it wasn't a mistake, and that's what sort of raised - it's the fact that it wasn't a mistake that puts it more on our map. This is significant because the protocol has been around from the beginning of the Internet. It's at v4, BGP 4, where it's been for a long time. It's sort of done. There have been security efforts made, but none that have gained traction.

So it's surprisingly kind of out there hanging, just waiting to get blasted. And there have been attacks on it, subversions of it through the years, which is why we've touched on it a little bit. But I thought, on the occasion of a clear, probably state-based deliberate subversion, we need to just plant a stronger flag in this and talk about it because I think it's very clear that, as we move sort of toward a world of state-sponsored manipulation of the Internet, we're going to be seeing more Border Gateway Protocol-based attacks. So that's our main topic.

But of course we've got lots of news, as we always do. We're going to talk about how Estonia handled the Infineon crypto bug, which crippled so many of the smart cards that we've talked about for the last couple months. They came up with their own approach, which I wish the other - I'm sure the other countries are thinking, ooh, why didn't we think of that, because it's kind of clever.

We've got a couple new consequences of the recent pressure to maliciously mine cryptocurrency as a consequence of the crazy valuation of the Bitcoin and Monero and other cryptos. Actually two zero-day exploits in the very popular and latest version of the vBulletin forum system, which those guys, the vBulletin maintainers did not respond to. So the guys who found these problems lost patience and just said, well, here it is. And they're bad. And so all vBulletin forum systems are currently vulnerable to these zero-day exploits.

Mozilla, believe it or not, is in the doghouse over "Mr. Robot," which is not a phrase I thought I would ever hear myself saying. But yes, it's true. We'll talk about that. And then Win10 had a weird thing happen where they bundled what turned out to be an insecure password manager by mistake. That got caught by the guys at Google. And I also want to talk about when a legacy protocol comes back to bite us, as some more actually Project Zero guys at Google found. Also a way to bulk steal any Chrome user's entire stored password vault, which is sort of simple. And Google said, yeah, we're not fixing that. So I just want to make sure our listeners understand that this is possible.

Also, we finally know where and why the uber-potent Mirai botnet was created, and by whom. So that we kind of, in time for Christmas, we get to put a little bow on that one. We also have a bit of errata; some fun miscellaneous stuff; and, as I said, we're going to then finish by taking a look at BGP, which is another critical, yet crucial and vulnerable protocol that glues the entire global Internet together. So I think as our penultimate podcast of 2017...

Leo: It's a good one, yeah, yeah.

Steve: ...we've got a goodie. This Picture of the Week I just had in my archive. Sometimes more than one comes in that's just too fun. And so I have a backlog, and no particular picture caught my eye this week. So I thought, okay, I'm going to pull one out of the archive that we've never shown before. I just got a kick out of it. It's just very clever. It shows a CAPTCHA license plate with the explanation: "To ensure that a human is writing my ticket." And so the guy is holding up a Texas license plate. It says "The Lone Star State" down below. And you could imagine that some image recognition license plate snapping, you know one of those photo ticketing stoplight cameras could be challenged by this. I mean, we can kind of say, oh, that's BJN1484. But, ooh, boy, if you're not up to speed, especially if you're not expecting to have a highly obscured license plate, well, yes, that's the CAPTCHA. So I thought that was very clever.

Leo: We are not advocating this. I think it's probably illegal. But go ahead, yeah. You can't modify your license plate like that, I'm sure.

Steve: Yes, I'm sure you cannot obscure it to make it difficult. I have GRC space COM, and I did put a big...

Leo: I put a dot on ours, too.

Steve: ...color matching dot right in the middle. And one guy drove past me and was all incensed and said he was going to report me. And I thought, really? Because, I mean, if anything it makes it more memorable, rather than less memorable. Suddenly it's not just six unrelated letters, it's clearly a domain. But anyway. Can't make everybody happy. Oh. What? Hello?

Leo: Hello.

Steve: Oh, I'm sorry. I wasn't sure if we had lost our connection.

Leo: I just stopped making noise.

Steve: Oh. Leo.

Leo: I'm still alive.

Steve: Okay, good. Okay. So the case of the Estonia ID card. Estonia, as we've talked about before, is a very crypto technology progressive country, and they just jumped on all kinds of latest technological solutions. Well, they got bit in this case as a consequence of the very widespread use of the Infineon library-based crypto cards which, as we know, generated a weak private key, I'm sorry, a weak, yes, a weak public key which was factorable down to discover the private key that was hidden within it. So there was some good coverage of this, and I just wanted to - I'm going to share what was written because it explains Estonia's position.

The coverage wrote: "The Estonia ID card is used nationwide for both governmental and private sector services. Several critical processes rely on the operability of the digital identity infrastructure, while some of the systems support the ID card exclusively. In Estonia, mobile ID is also available for authentication and digital signatures in many, but not all services." So they explain that shutting down all ID cards, which this problem would have created, would have had a severe impact on the entire country, including the economic impact to businesses, probably resulting in the deployment of substitute measures with lesser security standards, while making several services more costly and time-consuming. So they bought full into the use of this secure crypto smartcard for all kinds of identity, for signing contracts and so forth.

So they write: "Replacing all the cards physically would have taken a long time given the necessary steps for creating an entirely new card: choosing the chip, programming and testing the application, acquiring the necessary certification, and procuring the new cards equipped with the new chips. Only after those steps would the actual replacement procedure take place, limited by the low amount of available personnel in the border and police service points."

They wrote: "According to our estimates, the process would have taken at least a year, if not more, to complete. The alternative was to create a solution that would bypass the vulnerability by updating the existing cards. There is a requirement that keys" - and this is key for basically the guarantee. They wrote: "There is a requirement that keys must be generated on the card and never leave the card. This is required in order to be able to use the ID card to give legally binding digital signatures. The vulnerability that we," they wrote, "had to bypass was found in the on-chip RSA key generation procedure. We had to abandon using RSA altogether.

"Thankfully, the ID card chip also supported elliptic curve cryptography, which was not affected by the vulnerability that had been discovered. The solution was to update the cards to use elliptic curve crypto instead of RSA. We analyzed the possibilities to continue with RSA by using alternative key lengths, but ruled them out for several considerations." They said, for example, there's no capability of generating 3K-bit keys within the chip. And the problem was that the only way to solve the problem that it was too easy to factor the output would have been to dramatically increase the size of the primes in order to bring the difficulty back up, but these chips couldn't do that.

So they said: "Moreover, the decisions had to be made before" - I love this - "before the article on the vulnerability was published. In that situation, migrating to ECC was the most viable decision."

So in our initial coverage we talked about how it was stakeholders in this library, the major producers of the cards and the countries that were using them, did have an early pre-announce window. And these guys are so committed to the security that they were okay relying on the known defective signing, the known defective encryption, as long as it wasn't made public. So their window in which they had to fix this was when this knowledge would go generally public.

So they said: "Devising the concept for the solution itself was done rather quickly, mainly due to the lack of alternatives." There wasn't much to choose among. They said: "Most of the time was spent on the development and testing of the ID card base software and card application, retuning the remote update system, updating the service provider systems to support elliptic curves."

So they had to update the infrastructure to support this and then basically reprogram the existing cards. They said the two core functions of the Estonian ID card are authentication and digital signatures. Legally binding digital signatures in Estonia have always been time-stamped, retaining the authenticity of digitally signed documents even throughout this situation, as it is possible to provide evidence that the signature was given before the information about the vulnerability became available.

So we had only talked about this Infineon library problem being a complete catastrophe and basically shutting down everything. Because the Estonian government chose to, even though they use the Infineon library, the cards that they used also supported elliptic curve, they were able to move the entire country's infrastructure from a crypto based on the difficulty of prime factorization over to elliptic curve. They didn't talk about how the keys would have gotten much smaller as a consequence, which we know is the case because the elliptic curve problem is so much more difficult to solve that you need fewer bits, where essentially each bit is stronger than you do to protect against prime factorization.

So I wanted to - I think this is probably the end of our discussing this Infineon problem, but it was nice to see that, as a consequence of the fact that the processor in the smart card could do elliptic curve crypto, that they were able to essentially sidestep the problem. Certainly, I mean, it was a big deal to switch the entire country's infrastructure over. But they had many months of foreknowledge before the factorization problem became public; and that meant that, before that happened, everybody was switched over. And thanks to the fact that signatures are timestamped, they would be able always to know whether an RSA-based signature had happened after the disclosure or before, trusting it before and not after, and the crypto should have been moved to elliptic curve anyway by the time of the disclosure.

Leo: Very smart of them to have elliptic curve as an alternative device. I mean, that's really interesting that they put both RSA and elliptic curve.

Steve: Yes. Yeah. And I didn't include it in my notes here, but in their discussion that went on at some length, they recognized how fortunate they were and suggested that, as a matter of policy, any similar future use of widespread crypto for applications like this should absolutely include the ability to update the technology. It's just, you know, and that's been the history of this podcast is we keep seeing protocols which were initially believed to be secure are found to have some problems. We keep moving hashes up from, you know, used to be MD4, then MD5, then SHA-1. Nobody wants to use any of those, although we all did at one point.

Now it's SHA-256, and we're even looking to future next generations beyond that hashes. And the same thing with crypto. Some of our early podcasts talked about all of the problems with the WiFi protocols that we were once using. It's like, whoops, can't use that anymore. So I think it's very clear we're learning lessons. We're learning that IoT devices have to have a means of keeping themselves current and updating themselves because they're computers, even though they look like toasters. So, similarly, a smart card has to have a way to be able to update, to be updated when problems are discovered.

Leo: Did they, I mean, it's more than just an update. They had the capability in there; right? I mean...

Steve: Yes, yes.

Leo: So they had to have the foresight to do that. I'm sure it cost more to do that. They had to have more memory and more capability. But they had the foresight to do that, which is really remarkable, really.

Steve: And to have chosen a chip that had a processor that was able to do it, yeah.

Leo: Really impressive.

Steve: Yeah. So I read this, and I thought, what? Except it was covered by Bleeping Computer. These guys really know what they're talking about. Kaspersky Labs has discovered an evolved version of an older, from a couple years ago, 2015, older Android malware family, which at the time was known as Podec, P-O-D-E-C. The updated strain goes by the name Loapi, I guess, L-O-A-P-I. It has been found in a large number of apps hosted by third-party, not Google Play, repositories, Android app repositories, masquerading either as adult-themed apps or bogus antiviral utilities. But the kick here is, okay, so if users don't remove this from their Android devices, when left to its own means, this Loapi malware, which is itself capable of all kinds of different malicious behavior, will eventually get around to downloading a Monero cryptocurrency miner, which it runs, ignoring all - yes, exactly.

And I said at the top of the show there are two stories which are a consequence of the pressure now to mine cryptocurrency. Web browsers are doing it, and they're just trying to, you know, everybody's trying to sneak this in and steal CPU cycles. So this thing will overheat and overwork the phone's components, ignoring all power management throttling, which ends up making the battery bulge and deform the phone's cover or worse. So it's literally destroying, ultimately destroying people's phones. The original malware, which was just used to bypass the advice of charge, so-called AOC, and CAPTCHAs, which then subscribed their victims to premium rate SMS services, that was a couple years ago. Today this Loapi malware is far more advanced. Kaspersky experts called it a "jack of all trades."

First of all, as we just saw, it can mine the Monero cryptocurrency. It can install a proxy to relay the phone's traffic; inject ads in notification areas. It can inject ads in other apps; open URLs in browsers also used to show ads; download and install other apps launch DDoS attacks; interact with the Phone's SMS functions; and crawl web pages, apparently to subscribe users to premium SMS services and more. So the real takeaway here, folks, is really be careful where you get your software. The fact that this is - I saw a picture of the icons that this malware was behind, and I was sort of wishing they had done a better job of blurring the adult themed icons because they weren't quite blurry enough.

So there are all these apps in the non-Apple Play store, non-curated repositories, and I'm sure there are people who are wanting to get things for their Android devices that you can't get from the Google Play Store. But getting them really is a mixed blessing. So I would say, first of all, be very careful with where you obtain apps for your Android devices. And also, the first moment you notice your phone getting hot, consider what you may have recently installed and get rid of it, if you can, because if it happens to be this thing, it looks like it just very basically melts down your phone, given time. Which seems a little antithetical to its purpose. If it destroys the hosting platform, then it's not going to get any more crypto mining there. But I guess they just want to squeeze it as hard as they can for as long as they can. And then...

Leo: And you'll get a new phone, and then [indiscernible] because you need that wallpaper app. You've got to have it.

Steve: That's right.

Leo: Although I have to say you can't trust stores anymore at all. You saw that Apple had two counter - we talked about one last week. And now there's another counterfeit app. Who's minding the store? I don't understand how this happens. I mean, that's not malware. It was just ripping people off.

Steve: Yes. Or in this case who's "mining" the store.

Leo: Now you've got your title, I tell you.

Steve: That's right. And in a similar - I said that there were two instances of consequences of this increased pressure to mine. There's another campaign which has been named Zealot, Z-E-A-L-O-T, by the guys at F5 networks who found it. It is also mining Monero, which seems to be the one to go for these days, on Linux and Windows servers. The guys at F5 Networks found evidence of an aggressive and sophisticated malware campaign currently underway, targeting Linux and Windows servers with an assortment of exploits whose goal is installing Monero cryptocurrency mining malware.

And it's interesting, I mean, this makes sense when you think about it because, first of all, servers have a public Internet-facing presence so you can scan for them and find them. And they're typically beefier than end-user - certainly they're beefier than light bulb processors. A server machine has got a lot of RAM and lots of cores and heat sinks and fans. And it's going to be a much more effective miner. So again, it sort of makes sense that, if somebody was thinking, scratching their head, huh, where could we find some ready-to-go CPUs to use for cryptocurrency mining, it's like, ah, servers. Well, if you can get them.

So the F5 guys noted that the attackers are apparently fans of StarCraft since many of the names found in the code and the files of the campaign are taken from the game. First of all, of course, Zealot. But then also Observer, Overlord, Raven, and others, all apparently familiar to StarCraft fans, one of which I am not, but I'll take their word for it. So for Linux servers they are scanning for the now famous Apache Struts flaw, which as we know was the flaw used for the Equifax breach, on the Linux side, and the so-called DotNetNuke ASP.NET Content Management System vulnerability, which exists in unpatched Windows servers.

So they're targeting both Linux and Windows using well-known previously fixed flaws, assuming that there are some machines they can find that are publicly exposed and don't have those things fixed yet. And essentially, so they're using those problems to get into the system, then using Windows PowerShell to download and install the final stage malware in the course of Windows, or the Apache Struts flaw to use Python scripts, which apparently were lifted from the EmpireProject, which is a post-exploitation framework, very popular, to then install the Monero miner software. So again, it's sort of clever that they're using scanning for publicly available servers and targeting servers because they're more effective crypto miners than people's light bulbs and toasters and DVRs that typically have much more modest processing power.

I mentioned also at the top of the show something of concern currently, as in right now, for vBulletin. vBulletin is one of the oldest, longest running, very popular forum software. It was written, I think, initially in 1999, so 17 or 18 years old. It's gone through several versions. And then actually there's been some churn and some controversy as developers came and left. And Zen 4.0, which is the web-based forum software that I ended up choosing, was written by some guys who left vBulletin in order to do a complete rewrite. And so there was some litigation as a consequence of that that ended up getting settled a few years ago.

But in this case, web forum systems have been traditionally very difficult to secure. I mean, they have been, though not a lot recently, historically they have really been a problem because their nature inherently allows remote users to submit posts, which are stored, typically on SQL server backends. Then those postings are parsed by the server and displayed on everyone else's browsers.

So as we know, the famous example of a user naming themselves Johnny Drop Table, which combines some SQL commands with something that the server would read from the database while it's displaying the forum page and, I mean, actually has in the past caused tables to be deleted from the forum software. And cross-side server vulnerabilities, I mean, all kinds of problems have surfaced on forum software because the framework being used had vulnerabilities which clever hackers were able to leverage for their own advantage.

Leo: Also PHP is kind of horrible, and they're almost always in PHP.

Steve: Yes, yes.

Leo: I think vBulletin, was it a successor to PHP My Bulletin? I bet it was.

Steve: Yes, I think that's the case.

Leo: And, yeah a more modern version of that.

Steve: And so they've gotten better. And for what it's worth, because of this historical problem, I have built another server. Actually, I did it last summer, ready to be used.

Leo: Isolated, yeah.

Steve: Yes. And it is on physically separate hardware. I have a physical firewall between it and the rest of GRC, and the entire network is on its own physically isolated network because I just absolutely, I mean, I'm not writing a bulletin board. Everyone's glad that I'm not writing my own bulletin board system from scratch. Yes, I would love to; but, no, that's just not practical. So what I'm doing is I'm using the best that I could find, and I really like Zen 4.0. But I can't trust it, and I can't allow anything that happens there to infect the rest of GRC. So as you said, Leo, it's on a physically separate box with complete network isolation. So, yes, if something nasty crawls into it, well, at least it'll be contained.

So two different groups found zero-day, well, found existing vulnerabilities in vBulletin, bad ones, remote code execution vulnerabilities that not only allow them to execute code, but to do so with elevated privileges. In both cases they contacted the vBulletin maintainers and, after more than a month, or I guess not quite a month, actually, got no response. Never received an acknowledgment, an "Oh, crap, we'll fix this immediately," nothing. So they went public with them. And not only with full disclosures, but also proof-of-concept code demonstrating these vulnerabilities. So they are in the wild. They are effective against the most recent v5 of vBulletin. And they've not been patched.

So I don't know, unfortunately, there's no real takeaway. I don't know, it's not clear whether the servers that are hosting this vBulletin would be subject to compromise. I mean, certainly they would. I guess it depends upon what the attacker wants to do. If the server's compromised, that allows them to get into the network that the server is residing on and then probably spread from there. So they might just be using vBulletin's flaws as a way into the Intranet that is hosting that server, meaning that users, external public users of that vBulletin forum would not be attacked. On the other hand, if the vBulletin system has been compromised, it could be used to turn around and maliciously attack visitors to that forum.

So I'll be keeping an eye out for updates. And of course the problem is that many of these sites are not being updated regularly. So although it doesn't look, as far as we know, it doesn't look like there are any fixes to that now, which is why these problems were made public. Even when they are, forum software tends to lag and not to be kept current. And because of the history of problems and security problems, they are being updated, at least by people who are being responsible. I can't explain why there was no response from the vBulletin people. I haven't looked into whether, in general, they are being responsible or not. If anyone's interested in more details, I have links and the various CVEs that have been issued to cover these vulnerabilities.

But, I mean, and in fact you mentioned, Leo, about PHP, and that is absolutely the case here. The first vulnerability discovered in vBulletin is a file inclusion issue that enables remote code execution. A remote attacker is able to include any file from the vBulletin server and execute arbitrary PHP code against any file installed on Windows. The disclosure includes working proof-of-concept exploit code to show the exploitation of the vulnerability; and in this case a CVE, the Common Vulnerabilities and Exposures number, has not been issued.

The second vulnerability has been assigned a CVE. It's 2017-17672 and is described as a deserialization issue that an unauthorized attacker can exploit to delete arbitrary files and execute malicious code in some circumstances. The vulnerability arises from the unsafe usage of PHP's unserialize function on user-supplied input. So in both cases, as you thought, PHP is the underlying problem because that's the implementation language for this BBS software, this forum software.

Leo: It always is. The problem with PHP, it can be written securely, it's just that it's so easy, and it allows anybody to write code and doesn't really, yeah, gives you all sorts of dangerous tools.

Steve: Yup.

Leo: It's not good.

Steve: Yup. Now, okay. And get a load of this. When I first saw this I thought, what? Mozilla has gotten themselves in trouble.

Leo: Yeah, this pissed me off, actually.

Steve: Yeah. By force installing what appeared, well, what is a promotional add-on relating to, of all things, "Mr. Robot." I mean, the "Mr. Robot" show. So Mozilla and Firefox stumbled into the doghouse last week when they used a facility called Firefox Studies which is built into all recent versions of Firefox and enabled by default. So the takeaway for our listeners may be that you might want to consider disabling this because you can, which would have prevented this from happening. It didn't affect all users. But essentially what happened is that - and it's not clear whether it was done as a test, or if they wanted to stick their toe in the water, or what. Because it didn't affect all users. But they installed this apparently promotional add-on without their users' knowledge or consent.

So, okay. So first of all, this Firefox Studies facility is something that allows Firefox's developers to update their users' browsers with additional code for whatever purpose they may have. In this case, it took the form of an add-on named Looking Glass v1.0.3. And Looking Glass is a mystery. It's not clear what that's about. However, this Firefox Studies can be disabled on Firefox by opening, under the Settings, going to Privacy and Security, and turning off "Allow Firefox to install and run studies."

So this was done under that setting's default enabled state. So Mozilla's own support page said, which they put up to explain what was going on, they called it "Through the Looking Glass," and they said: "What's happening?" They ask rhetorically: "Are you a fan of Mr. Robot? Are you trying to solve one of the many puzzles that the Mr. Robot team has built? You're on the right track. Firefox and Mr. Robot have collaborated on a shared experience to further your immersion into the Mr. Robot universe" - this is them writing this - "also known as an Alternate Reality Game (ARG). The effects you're seeing are part of this shared experience. No changes will be made to Firefox unless you have opted into this Alternate Reality Game." Okay, except they installed an unwanted add-on, so I would call that a change made to Firefox.

Then they ask themselves: "How do I opt in or out? To participate, install the Looking Glass add-on from" - and then they have the add-on Looking Glass link. I have it in the show notes, if anyone wants it. This add-on is available only in the U.S. in English. Then they say: "If you no longer wish to participate in the shared world experience, enter about:addons into your address bar and remove Looking Glass."

And then they say, kind of confessing here, "Looking Glass was previously delivered as a shield study, so you might see Looking Glass II or Pug [P-U-G] experience in your past studies in about:studies. It has already been removed as a study and moved to an add-on so you do not need to take any further action."

Anyway, this whole thing is a mess. And as you said, Leo, it has angered a lot of Firefox users that Mozilla was doing something like involving something that seems to be promotional. It's like [crosstalk]...

Leo: Well, it has. I mean, I don't know if - they say they weren't paid. But that's just dumb, then.

Steve: Yeah. Yeah, exactly. Exactly. Not to have at least gotten something in return.

Leo: They didn't make it better, just made it dumb.

Steve: Yeah. So, wow. And the problem is I guess they didn't want to push a question out to all Firefox users. Are you watching "Mr. Robot"? Do you want to participate in an alternate reality game? You know, that would have been annoying, too.

Leo: If I do, I will install the extension. Don't put that in your main install. That's ridiculous.

Steve: Yes, yes. And don't push it in anyway. Somehow, I mean, I don't think there is a good way to ask a generic browser user who you have no reason to believe has any interest whatsoever in "Mr. Robot," good as it may be, there's just no way to inflict that upon them. That's not Mozilla's job. They really do, as you said, they need to stay neutral. So bad on them.

And this wasn't advertent or deliberate on Windows 10's part. But it's kind of a problem when Windows 10 has chosen to bundle and install a password manager in all Windows 10 systems which turns out to be vulnerable. And turns out to have known that it could have been vulnerable. Our good friend Tavis Ormandy, he wrote in the Chromium bug project, under Project Zero, his title was "Keeper: Privileged UI injected into pages again." And he wrote, Tavis wrote: "I recently created a fresh Windows 10 VM with a pristine image from MSDN" - that's the Microsoft Developer Network. I'm a member, and Microsoft developers are. You're able to get - what?

Leo: Well, I just want to point out that seems to be in the MSDN version. It's not in any version, I've tried all my Windows machines. Keeper is not installed on any of my Windows machines.

Steve: Oh, no kidding. So it's...

Leo: It must be something they put in MSDN for developers as a kind of [crosstalk].

Steve: Oh, good. Oh, okay.

Leo: I mean, at least I can't find it. I'll ask tomorrow on Windows Weekly. Obviously we'll be talking about it.

Steve: There is a reddit link I have in the show notes where - and again, I didn't go there to look. So let's hope that it never went public. That would be great because...

Leo: Well, they put stuff in MSDN versions of Windows they don't put in other versions of Windows, I presume, for developers.

Steve: Okay, good. So I don't know either way. But anyway, Keeper that was in this MSDN version, which, well, it did have a problem that Tavis had previously reported to them. He did say he's not the only person who's noticed this with their link to reddit. But it may have been other developers. I don't know whether it was in the public or not. So I'm glad you add that caveat, Leo.

He says: "I assume this is some bundling deal with Microsoft." He said: "I've heard of Keeper," he said, "and I remember filing a bug a while ago about how they were injecting privileged UI into web pages." He says: "I checked, and they're doing the same thing again with this version. I think I'm being generous considering this a new issue that qualifies for a 90-day disclosure as the same attack [still] works. Nevertheless," he said, "this is a complete compromise of Keeper security, allowing any website to steal any password. Here is a working demo that steals your Twitter password." And he provides a link to a demonstration of this happening.

So the tech press coverage had headlines like: "For eight days, Windows bundled a password manager with a critical plugin flaw." Which may or may not be overblown, depending upon whether this was actually pushed out to end users. So the good news is the Keeper folks responded instantly to Tavis, fixed the problem by removing some functionality that contained this problem, and had all of the patches pushed out within 24 hours.

So Keeper itself I guess has been around for years, and it's a well-known password keeper extension available on Edge and Chrome and Firefox. The takeaway for our listeners is you want to make sure, if you're a user of Keeper, that you're at 11.4.4 or newer since they fixed that moving forward. So anyway, I guess we'll find out whether it ended up being made public, or whether this was just MSDN. I don't know. I think that Tavis is suggesting that this was the image that, well, we know he's suggesting or he's saying that it's the one that he got from Microsoft. Who knows whether it went out public. And Leo, it's time for me to sip my coffee.

Leo: Indeed, indeed, indeed. And I'm just asking the chatroom if any of them saw Keeper installed. And it looks like - I have Keeper on the Fall Edition, just my luck. So some people might have it. That's [chatroom handle]. That's interesting. I mean, I've used Keeper. I wonder if that bug in Keeper is similar to the bug - there was a cross-site bug in LastPass, as well, that was fixed; right? Is it similar to that, I wonder? Or maybe something else?

Steve: I think what upset people, as I understand it, is unlike other Windows 10 bundles where you get a link to install something, this actually installed it. I mean, it was there installed by Microsoft with this update. And as we now learn, for a window of eight days, apparently, there was this vulnerability present.

Leo: Yeah, hmm. More research will be done, and we will report back. Yeah, I mean, it may be they put an ad for it, but that still doesn't install it. You have to...

Steve: No, no, no. It's installed.

Leo: You say it's installed, yeah.

Steve: Yeah. So, for example, here in reddit: "I just reinstalled Windows 10 today, and I was uninstalling all the bundled apps like usual, and I noticed that Keeper password manager is pre-installed now. I've never seen this come installed with Windows before."

Leo: The Windows 10 I'm using was installed from the Windows 10 Media Creator and does not have Keeper installed, does not even have anything promoting Keeper installed. So I just don't know. Yeah. It's unclear. But I'll ask Paul Thurrott. Windows has many versions. Many versions of Windows exist.

Steve: Ah, yes. Not simple. The coverage of this talked about Windows 10, but this is really not a Windows 10 issue. The problem is much bigger than that. There's a technology which has been around since the very beginning of the early days of the Internet, even before 1996, known as "proxy auto-config." The problem that this was intended to address is that a browser comes alive, wakes up launched in a machine, and it needs to have a way to get to the public Internet. Yet there may be no direct connection. It may need to run through a proxy, but it may need to be configured for that.

And those of us who've been around a long time may remember the days when you could manually configure the proxy, and those dialogs are still around if for whatever reason there isn't an auto-config. Well, there is this auto-config capability which is a protocol that Netscape, I mean, the Netscape, back in the early Netscape Navigator days, they designed it. And they decided that the file which would be used, a PAC file, would be JavaScript. So not XML, not something less capable, but full-on JavaScript. So probably because that's the language that they had in the browser already, and they said, oh, let's just have the auto-config operate by running a JavaScript script. So, okay, we have that.

Now the next problem is where do we get this file? How does the browser know to get the file? Or maybe how does the Windows OS itself know to configure access for itself to the Internet? And so essentially all Windows systems have this proxy auto-config and the protocol that carries it, WPAD, Web Proxy Auto Discovery protocol, enabled by default. It exists, because it's a standard, it is also - it's not just something random that Netscape came up with. It exists as an IETF draft which, although it expired in 1999, here we are in 2017, and every Windows machine will ask the network when it boots up, essentially, hey, where can I find a JavaScript file to execute? And it looks, it asks for this through DHCP and then DNS and then the WINS, W-I-N-S, and maybe even some other solutions.

So basically it casts about all over the local network, querying for anyone who will respond with the URL where this JavaScript file can be found and executed. Even Chrome does this, but it executes the JavaScript in a sandbox to limit any damage that can be done. The problem with Windows is that, even though there are newer JavaScript interpreters available, it still uses the old original JScript engine which has a whole bunch of known vulnerabilities.

So Google's Project Zero guys took a look at the JScript engine and generated a complete soup-to-nuts execution exploit. And what's a little more worrisome is that there are a number of ways of actually pulling this off. For example, if you were a machine on an Intranet where a Windows machine comes up and first sends out a DHCP broadcast saying "I need the WPAD PAC [Proxy Auto-Config] file," if you beat the legitimate DHCP server that may be running on that Intranet, assuming that there is one, if you beat it to the punch, you're able to provide the URL to the Windows machine for its configuration and essentially cause it to run JavaScript of your own design.

And unfortunately, since it runs it on the older JScript engine, probably for compatibility reasons because in fact it was one of the - the lack of features in JScript gave the Project Zero guys some headaches because they couldn't run their normal JavaScript find-bugs-in-JavaScript code because there were too many things that were not supported by the old JScript. So I'm sure that it's still running JScript, specifically to be able to leverage for compatibility reasons the code that once a long time ago ran and is still being run today.

So the big problem is that there's a list of things that's being done. If no WPAD PAC file URL is obtained from DHCP, then DNS is queried. And believe it or not, it's queried publicly. And there have actually been exploits in the wild where a public WPAD DNS URL was used, was being asked for, and a URL returned that a system would then execute. And it's .local is the domain, but it is being made publicly to the public DNS. So it turns out that the Google guys came up with a way of answering the query, which are going out into the public, generating some code which would be executed on any version of Windows. They talk about Windows 10, but there's no reason that it wouldn't work on any version of Windows for the last 10 years, at least. And take advantage of flaws in the JScript engine in order to execute their own code.

So they found multiple vulnerabilities; engineered a highly reliable proof-of-concept exploit. And the reason they just went public with this is that there's nothing necessarily to be fixed. I mean, this is all using long-established standards through multiple editions of Windows. And they suggest that the only solution they came up with was for Windows to disable looking for this WPAD PAC, the proxy auto-config file, or to block it at your border so that your own Internet queries don't go out into the public, or make sure you provide an answer locally so that you sort of snub the systems looking for this.

And I don't know if anybody here who's listening to the podcast has sniffed Internet traffic and Intranet traffic. You see these WPAD queries all over the place. I mean, Windows systems are actively asking, making these queries, looking for an answer. And the Project Zero has demonstrated that this represents a significant danger to Windows, for which there's no clear solution at the moment. So I thought that was a little bit chilling. But it's when old protocols that are still in use come back to bite us.

We've talked about Mirai extensively, of course, because it generated the largest denial of service attacks the Internet has ever seen. It brought down the very large DynDNS service. Brian Krebs of Krebs on Security was under such a sustained high-bandwidth attack that his longstanding service, Akamai, said, "We're sorry, Brian, we'd like to help, but these attacks are larger than we're able to afford and absorb. So you're going to have to find somebody else to host your DDoS protection for you." And we've talked about it. It was, what, some routers and DVRs and basically Internet appliances, Internet-connected appliances that the Mirai botnet was more effective in corralling and commandeering and finding and spreading to than anything we'd seen before.

And we've said on this podcast months ago, why? What is its purpose? What is its application? What is it trying to do? I mean, because it really frightened the security engineers and folks who track these things, generating multiple terabits of thousands of gigabits, thousands of thousands of megabits of attack traffic. Essentially, something that nobody could withstand. Well, the FBI got on the case and tracked down the origin of this.

Leo: Who was it? North Korea? China? Those Russkies? Who was it? Who did that?

Steve: Three college kids. They were running...

Leo: Not so bright college kids, either, by the way.

Steve: They were, yeah, they were running a Minecraft server, and they wanted to attack other Minecraft servers and force them off the 'Net in order to capture some of their Minecraft users. Apparently, and you know about Minecraft more than I do, Leo, but apparently if you run a popular Minecraft server, you can make money hosting Minecraft.

Leo: Yeah, sure you can, yeah.

Steve: And so these guys - so again, the dollar underpins the motivation. They wanted more users on their Minecraft server. So they said, okay, let's go blast some other Minecraft servers. So they built this botnet which ended up being way more powerful than they needed. So not only could they blast any Minecraft server that they could find, but it turns out that because DDoSing Minecraft servers is a thing, there are Minecraft DDoS hosts that specialize in hosting and providing bandwidth to Minecraft servers to protect them from DDoS attacks.

Well, the Mirai botnet was so powerful that they were able to bring down the DDoS protectors, the Minecraft DDoS protectors, and essentially take everybody, take all of the Minecraft servers off that were behind the Minecraft DDoS protectors. I mean, it was just this uber powerful, as we've discussed, just crazy powerful system. So the lead guy just pleaded - he's a college kid.

Leo: Rutgers, yeah.

Steve: Yes, at Rutgers.

Leo: Very proud of him.

Steve: Just pleaded guilty in an Alaskan court. He had two guys that he was working with. And our listeners will remember when we covered the fact that the Mirai source code had appeared online.

Leo: Senpai.

Steve: Yup, exactly. And we believed that - it was presumed that the authors of Mirai had let it go public in order to obscure themselves so that other people would pick it up and start using it, in order to sort of, like, spread the blame around, essentially. Well, before they did that, the FBI was already zeroing in on this little trio. And so it turns out that they were not the people who blasted the DynDNS and brought all this attention to Mirai. Their own use of their own original Mirai botnet was more focused on Minecraft. But by releasing it to the public they turned a lot of other - and here's your favorite word, Leo - "miscreants" lose.

Leo: You miscreants. Get off my lawn.

Steve: And it was other people who weren't probably themselves sophisticated enough or hadn't done the work of reverse-engineering the various IoT devices. Remember that Mirai was sort of a living organism. Shortly after a new exploit was found, it would get incorporated into this botnet, so it was constantly increasing its power and sophistication. By releasing the source publicly, anybody could get on the Mirai bandwagon.

And in fact it was some other people that were involved in blasting DynDNS off the 'Net and causing a big chunk of the eastern portion of the U.S. to go dark as the caches of existing DNS queries drained, and the resolving DNS servers tried to go back to Dyn in order to get their records refreshed, but found that Dyn was off the 'Net. So anyway, that's the story behind Mirai. It was three college kids who didn't like competition from other people's Minecraft servers and thought, okay, we're going to blast them off the 'Net, cause their systems to be slow, and thus collect other users onto our server.

Leo: Were they at some point selling DDoS as a service?

Steve: Yes, yes. There was also - it occurred to them, hey, you know, we could make some money by selling this to other people.

Leo: Hooligans.

Steve: Yes. What are you going to do? Unfortunately, I guess they're probably 21 by now, so they're probably being held responsible - or probably 18, rather. I'm sure they're 18.

Leo: Oh, yeah, yeah, yeah.

Steve: So, yikes. So, okay. I'm less overheated about this than several of our listeners who sent this to me. But Google does classify this, not as a bug, but as a feature. And that is the fact that anybody can essentially steal all of another Chrome user's saved passwords, form fields, bookmarks, and history, if they have physical access to their browser. Okay, well, now, the reason this is not that big a deal is that we've already talked about how it's possible, if you sit down in front of someone's computer who's using Chrome, with a few pokes around the UI, you can go to their passwords and tell it to show the passwords, and there they are.

So, okay. So if you allow anyone physical access to your Chrome browser, or for that matter many other browsers, they'll let you see the passwords, without having to authenticate. I mean, it's just there. So someone posting on Medium.com explained how that process can be done for all of Chrome's synchronized saved things. And it doesn't take any great rocket scientist to do this. You sit down in front of somebody else's Chrome browser. Go to settings/manageProfile, and then on that page - so it's chrome://settings/manageProfile. And then click on Edit Person, or you could go to chrome://settings/people. Then you sign that person out. So you sign them out of the browser. You then click Sign into Chrome and use your, as the attacker's, Gmail account.

So then Chrome helpfully notes that another user - and provides their Gmail account name - was previously using Chrome, and you're then given the choice of choosing between "that wasn't me" or "yes, that was me." So you say yes, that was me. And along with the "that was me" is "add my bookmarks, history, passwords, and other settings to" - and then the attacker applied Gmail account. And then you click Continue. And essentially that's exactly what happens is all of the, essentially, the profile of that Chrome user who you just signed out from is assumed to be the same person because you just said it was, as this other Gmail user and all of their profile, including all their passwords, are merged into the newly logged-in Gmail account.

So again, the takeaway is just be aware of that, that this browser synchronization will work across Gmail accounts and that, if somebody had access to your Chrome browser, they could sign out, sign in as a new person and say, oh, yeah, that's me, too, and merge all of your stored Chrome profile stuff into their own account and then browse through it, use it, log in as you, assuming there's no second-factor authentication, at their leisure. So again, not a huge issue because that could be done on a site-by-site basis just through the Chrome UI. But this does do the whole thing at once, which certainly could be a concern.

I didn't know where to put this. I thought, okay, we haven't had any errata for a long time. So I got a tweet from a Drew Green who said: "@SGgrc Been listening to previous episodes and heard you and Leo discussing Google's Advanced Protection Program. A Bluetooth two-factor authentication device isn't necessary." He said: "I found it works with an NFC YubiKey in place of a Bluetooth device."

Leo: Yeah. You can use the NEO, but you can't use it on iOS.

Steve: Ah, okay. And so he said: "Works with an NFC YubiKey in place of a Bluetooth device, with the other device being a different USB-only YubiKey." So anyway, I wanted to add that to our knowledge base, that you don't need Bluetooth. You can use an NFC YubiKey.

Leo: Unless you have iOS or a phone without NFC.

Steve: Right.

Leo: So YubiKey offers the NEO, which is an NFC-capable YubiKey, as well as the YubiKey 4, which you should use because it's the more modern one. But you want the Bluetooth just because there's devices that don't use NFC.

Steve: Ah, okay, good.

Leo: I think that's why Google just says it. You're right, they could qualify it and say, "If you have a device that doesn't support NFC, get the Bluetooth dongle."

Steve: Right, right. And a little bit of miscellany. There is a "Be a Coder" Humble Bundle which had a whole bunch of No Starch Press books available.

Leo: Yeah, some of my favorite LISP books in it.

Steve: Yes, yes, yes, yes. And they look like actually a lot of fun titles in there.

Leo: Oh, they're great, yeah, yeah.

Steve: So there's about 13 days remaining as of this podcast on December 19th. So it's called "Be a Coder." So you can just go to www.humblebundle.com, and you'll find it there, and four different sets of books. And I browsed through them. As you said, there's LISP, and there's...

Leo: "Learn You a Haskell for Great Good!" is such a good book.

Steve: Uh-huh.

Leo: I have many of these books, actually. It is such a good book. I have not read Erlang. That's a kind of a copy of the classic by Miran. But, yeah, these are great: "Clojure for the Brave and True," have that. "Realm of Racket," have that. "Land of LISP," have that. You probably have "The Art of Assembly Language," I don't know.

Steve: I don't, but I know the author. He's at UC Riverside and has put together a very nice introduction to assembly language.

Leo: Randall Hyde, yeah.

Steve: Yeah.

Leo: Now, these are just eBooks, though. You don't get the physical books; right?

Steve: Correct, correct.

Leo: "Ruby Under a Microscope," "Understanding ECMAScript 6" - there's some good stuff in here. "The Book of R."

Steve: Yeah, of course ECMAScript 6 is actually JavaScript, so that is the current scripting language for our browsers. So the funds are donated. They offer donating, or they suggest donating to the EFF, which of course is a great cause that we all support. So take a look and check out this "Be a Coder" bundle. As you said, Leo, there's a bunch of good stuff there.

Leo: Good stuff. Some real classics in here, yeah.

Steve: Somebody got a kick out of doing some math with the current bitcoin valuation. ZeroHedge noted - and thank you, Simon Zerafa, for forwarding his tweet. Okay. Back when the FBI confiscated the bitcoins which were in the possession of Silk Road back then, they liquidated their bitcoins and got a tidy sum, $48 million.

Leo: Oh, not bad, hmm.

Steve: But they would currently be worth 2.4 billion.

Leo: Geez Louise. I just don't get this. It's a bubble; right? It's not - I don't understand.

Steve: I like it. I like it.

Leo: Yeah, you got some sitting in the corner there. What would it have to get to before you said, yeah, I really feel I should - you know, I was thinking, you have a bad hard drive. Well, gee, it's too bad you don't know of any programs that could fix bad hard drives. Here's the guy who makes SpinRite, and he says, "Oh, it has a bad hard drive. I don't know if I want to."

Steve: Well, actually, as far as I know the RAID is fine. It was the motherboard that got crazy. And so I just moved the RAID to a different motherboard.

Leo: Yeah, that's easy.

Steve: It's probably there. So Tavis...

Leo: You're just going to sit on it, aren't you.

Steve: I'm going to kind of wait and see what happens.

Leo: What if it got to like 100,000? Then you'd be worth five million bucks. What if it got to a million? Then you'd be worth 50 million bucks. I'd take it. But then [crosstalk]...

Steve: Yeah, well...

Leo: Maybe it'll get to five million.

Steve: Maybe it's going to pop and crash.

Leo: It's going down. It's now at 17.5 again.

Steve: Yeah.

Leo: I don't know what to do. I don't know what to do.

Steve: Okay. So Tavis tweeted a little something sort of short and pithy, which I just liked because this is what we do on the podcast. He tweeted: "Everyone wants there to be simple answers in security. But sometimes there are no simple answers." And we've often said that. I was thinking of that in the context of SQRL, that it's a two-party system. I want it to be as easy to use as possible. I want it to be bulletproof. But as I have said, the user has to bear some responsibility. That's going to be a sticking point because everybody is used to having an, oh, you know, I don't know what that is. I forgot my password. Click this link. I mean, it's incredibly insecure to do that. I mean, it's all we've got. It's possible to have much more secure solutions. But they're just not there. As Tavis says, everyone wants simple answers in security, but sometimes there are no simple answers. It's true.

And I did want to just follow up with the iOS 11.2.1, both for iOS and for tvOS, that the anticipated patch that we discussed last week to fix the problem in HomeKit did happen. So iOS is now at 11.2.1, which fixed that HomeKit vulnerability and restored that remote access for shared users which they had briefly neutered at the server side in order to fix a couple of those problems that we discussed at greater length last week.

And also, speaking of bitcoin and liquidating, Leo, a bunch of people said to me that any sale would qualify as a capital gain, as long as the asset had been held for more than a year. I wanted to verify that because it would be a significant effect, and I verified, yes, they are in fact...

Leo: Yeah. They're an investment, really; right.

Steve: It is the case that I've had them for many, many years. It would be treated as a capital gain, as a sale of a capital gain held for a long time, or long term. So in the U.S....

Leo: If this new tax bill passes, you're a corporation. It was a corporate creation. It's a mere 21 percent.

Steve: Well, actually, as a capital gain it's 15.

Leo: Oh, 15. Oh, well, we're happy now.

Steve: Yeah.

Leo: Or you could just pretend you didn't get anything. How would they know? Maybe they listen to the show.

Steve: Yeah, well, and Leo, as I said, if I were behind bars, people might look askance at my security advice.

Leo: Good point.

Steve: How good a security expert is he when he's been locked up?

Leo: Excellent point.

Steve: Anyway, and we all know I'm not greedy by nature, so what the hell. Okay. Two notes about SpinRite real quick, just tweets. Nothing makes my day more than #SpinRite saves the day. Philipp with two Ps, P-H-I-L-I-P-P, tweeted: "SSD drive was failing. Had everything backed up" - dot dot dot - "except my private GPG key. Damn!" in his tweet. He said: "A Level 2 on the drive took 26 hours, but the drive was mountable again, and I rescued the key. Thanks for a great product."

Leo: Nice.

Steve: So once again, we are now, I mean, this is why there will be a SpinRite 7 is because it turns out it fixes solid-state drives, too. Who knew? And I'm going to keep the name. It's not going to be SSDRite or anything, no. It's got to be called SpinRite, even though it's not spinning anymore.

Leo: Someday you'll have to do a bit saying, you know, it's called SpinRite because in the old days hard drives moved.

Steve: Yeah, once upon a time.

Leo: You may not remember this.

Steve: And also Gregory Paul said: "My two USB keys booting my FreeNAS failed, both around the same time. Thanks to @SGgrc for SpinRite, which bring them back online" - okay, I didn't really read this closely, which he's saying "brought them back online in time to migrate from USB keys to an SSD." So he said: "#FreeNAS and #SpinRite." And I should mention that one of the things you want to make sure FreeNAS is not doing is swapping at all. You want to make sure that your swap drive is in RAM because you absolutely don't want to have an unnecessary swap drive for an instance of Linux or FreeBSD which is actively swapping on solid-state memory because it'll kill it.

So I don't know that that's how these two USB keys died because we know that they do tend to die. Thank you, Gregory, for sharing the fact that SpinRite was able to bring them back. And I'm really glad because it's much easier, obviously, just to copy those over to an SSD and then boot that. But for what it's worth, anybody running any sort of a turnkey system from a solid-state drive, whether it's an SSD or any kind of non-volatile memory, make sure that your swap drive is not on that physical media. You want it to just be in RAM because, as we know, that media does not like to be written to.

Leo: Okay. Border Gateway Protocol, BGP.

Steve: Border Gateway Protocol, BGP.

Leo: Not Bill Graham Productions, Border Gateway Protocol.

Steve: Well, that is a blast from the past, yeah. Okay. So we've talked a lot in the early days, where we were discussing the way the Internet works, about basically the invention, which is what it was back in the DARPA days pre-Internet, ARPANET, of autonomous routing, the idea being that you would have a federation of routers that are interconnected in just sort of an ad hoc, kind of random way. You want enough connectivity that, if a link between any two routers went down, there would be like another way to get the data around where it had to go. So the idea being multiple paths.

The Internet is inherently a multipath system where there are many ways, an incredible number of ways today, for data to get from point A to point B. Thus we get redundancy and resilience and so forth. At every router, in every router, there is a table, not surprisingly called a "routing table," which is a list of IP addresses, or more technically or more correctly IP address prefixes, which instruct any incoming packet, with the help of the router, which outbound link to forward that packet to. I call them "address prefixes" because you couldn't and you wouldn't want every single IP address on the Internet, not just for the time being, restrict us to IPv4 for the sake of ease and explanation. There's 4.3 billion IPv4 addresses. So it would be impractical to have every IP address associated with which interface it goes out of.

And it's not necessary because one of the cleverest things, one of the many clever things about the Internet is that networks are groups of adjacent numbered IPs. So, for example, and we've talked about this, HP once upon a time had 15-dot everything. In other words, they had the entire - they had every IP address - I think it was 15 and 14, come to think of it. They had every IP address beginning with 15 or 14, which meant that anywhere else in the world, if a packet was bound for an IP address beginning with 14 or 15, it could be sent toward Palo Alto, back say when HP was only in Palo Alto.

The point being that one entry in a router, in the routing table of every router in the world, was able to just aim, essentially, if it was two IPs, 14 and 15, that would be one 128th of the entire address space, which is two IPs in Class A of the first byte, one 128th, at a single location, dramatically simplifying the challenge of routing all 4.3 billion IPs to their destination. So this is this notion, this way the IP space is divided is the bits on the left, the most significant bytes and technically bits specify the network number, and then the right-hand bits are the machine within that network.

So what that means is that, for example, a given company may have several blocks of IP addresses that are adjacent, several, for example, Class C networks, a Class C being 256 IPs. Maybe they have four of them, so that's 1024 IPs. Well, that means that the last 10 bits don't matter for routing because they just have to get to the company. Once they get to the company, then internally the 1024 machines on those IPs will respond.

But that means all the routers in the world, as long as the upper - let's see. We have the lower 10 bits, so the upper 22 bits, that specifies that company. So again, the routing table is simplified. And the company is probably buying its IPs from its ISP, and the ISP is selling similar blocks to many other companies, meaning that the ISP has a much larger network of, like, 10 for this company, 10 bits for that company, and so forth. So even fewer IPs or bits on the left side will get the packets to that ISP. Then it uses the next chunk of bits to decide which of its customers it routes the packets to. And then each of those companies routes them to its system. So just this incredibly brilliant hierarchical system.

And to us it's sort of - we can take it for granted now because we've been living with it for decades. And you could almost say, yeah, well, that's obvious. Well, yeah, in retrospect. But these guys thought of this, invented it from scratch. So all over the world we have routers containing these tables which need to figure out where to send stuff. And the challenge is this is not - the Internet is not a static environment. It is dynamically changing.

And as I mentioned before, links between routers are coming up and down. If a link between two routers drops, then suddenly the routing table of both routers is wrong because they were wanting to send packets to each other, if each of them was nearer the destination of where those packets were trying to go. But if that link goes down, suddenly they need to update their routing tables to find the next best route in order to send the packets to, since the better route, the preferred route, is no longer available.

So think about that for a minute. You have routers all over the globe that have tables that need to be managed continually and dynamically. You've got ISPs adding customers, removing customers. You've got, remember also, we've got IPv4 address space depletion so that there are governments that used to have big blocks of IP space where IPv4 is suddenly valuable. It's a commodity. So they're selling off chunks of their network to other people who want to buy them. So suddenly IPs that used to route into that country now go somewhere else entirely. The point is this cannot be done by hand. You cannot do - there's no possibility of managing and maintaining the tables of the routers that are global manually.

BGP, the Border Gateway Protocol, is the protocol that automates this management. It is carried over TCP. So all routers establish persistent TCP connections to every other router that they're connected to. And I forgot what port number, it's 179 maybe, something like that. It's a well-known port in the same way that 25 is for SMTP and 80 is for the web and 443 for HTTPS and so forth. There is a well-known port which accepts TCP connections and only TCP connections because we need to prevent spoofing. And routers establish a link to another router and communicate using BGP.

Okay. So let's step back a bit. Understanding how routers interact, the necessity of routing tables being maintained dynamically, the need to automate that process, and that BGP is what does it. The reason this all popped up on my radar is that exactly one week ago - there is a great site, BGPmon.net, B-G-P-M-O-N dot net. They're just a freely available Open BGP monitoring site. They were purchased some time ago by OpenDNS; and, as we know, OpenDNS was subsequently purchased by Cisco. They watch, among other things, but as BGPmon would suggest, they watch what's going on with BGP on the Internet. A week ago one of their guys posted the following:

"Early this morning (UTC) our systems detected a suspicious event where many prefixes for high-profile destinations were being announced by an unused Russian Autonomous System." So let's stop back. We need to define some terms. "Early this morning our system detected a suspicious event where many prefixes for high-profile destinations were being announced by an unused Russian Autonomous System."

So first of all, "prefixes" is the term for the right-hand side of the IP. So that's called the "prefix" as opposed to the - sometimes it's also called the "network number" as opposed to the right side, which is the machine within that network. So those are prefixes. They use the term "announced" because that's also part of the BGP jargon. A router "announces" to the world via BGP the networks that it has responsibility for. So, for example, an ISP's border, it's called an "edge router" because it's on the edge between the ISP and the public Internet. An ISP's edge router will announce that it's managing the traffic. Behind it are the following networks. And so it announces it by announcing prefixes, essentially saying all packets where the left number of bits match this prefix should come to me because the machines for those prefixes are behind this edge router, so send all that stuff to me.

An autonomous system is, again, the official nomenclature, the name. It's called an "AS number," and they're by number. It is the numerical designation for an ISP like I was just describing. So Cox or Level 3 or Google, Apple, any major player on the Internet is an autonomous system. And I kind of regret that I didn't apply for one myself for GRC many, many moons ago, years ago, because what you get if you're an autonomous system is an AS number and your own allocation of IP space. The point is that, when I've moved from, as I used to be at XO, to Level 3, and before that I was at Verio, and then Cogent, all of those movements have required me to renumber my IPs for GRC. It's not painless, and it can be challenging to not have any downtime or to minimize downtime.

But if you are an AS, you have your own IP space, and it's portable, meaning that, had I done that, I could have said to Level 3, okay, I want to now be hosted in your datacenter, and my AS number is as follows. And so essentially their edge router would have begun announcing the prefix that I own, and the same IPs that were once going to, for example, Verio, would have been rerouted, literally rerouted to Level 3. No DNS would have changed. That IP space would have just moved to a different location. So, I mean, this is how cool this system works. And frankly, we owe the fact that this is even possible to the cleverness, the automation of this routing system behind the scenes.

But as BGPmon wrote: "Early this morning our systems detected a suspicious event where many prefixes for high-profile destinations were being announced by an unused Russian Autonomous System." Now we know what that means. A Russian-located AS, like a big ISP, was announcing prefixes, announcing that it was the owner of prefixes, that is, big chunks of IP space, for high-profile destinations.

They write: "Starting at 04:43 (UTC), 80 [eight zero] prefixes [blocks of IPs] normally announced by organizations such as Google, Apple, Facebook, Microsoft, Twitch, NTT Communications, and Riot Games were now detected in the global BGP routing tables with an Origin AS of 39523 [which is DV-LINK] out of Russia." Meaning that this entity, this Russian entity was claiming that it was Google, Apple, Facebook, Microsoft, Twitch, NTT Communications, and Riot Games, among others.

They write: "Looking at the timeline, we can see two event windows of about three minutes each." So they were not long-lived. "The first one started at 04:43 UTC and ended at around 04:46 UTC. The second event started at 07:07 UTC and finished at 07:10 UTC. Even though," they write, "these events were relatively short-lived, they were significant because it was picked up by a large number of peers and because of several new more specific prefixes that are not normally seen on the Internet. So let's dig a little deeper," they write.

"One of the interesting things about this incident is the prefixes that were affected are all network prefixes for well-known and high-traffic Internet organizations. The other odd thing is that the origin [which was] AS 39523 has not been seen announcing any prefixes for many years," with one exception which they note later. "So why does it all of sudden appear and announce prefixes for networks such as Google?" Okay, I'll stop at this point.

So the idea, when they mention "peers," is that not only was it, this AS, this Russian Autonomous System, announcing that it controlled IP space that it did not, but the nature of BGP is propagation. That is, when I was speaking earlier, I talked about how routers are all linked between, they're all interlinked inherently. And they establish links to all the routers that they connect to. And then they share their routing tables. So this bogus announcement, which is essentially bogus routing information, bogus routes, were stuck in one router, and it shared them with its peers. The peers said, "Oh, wow, look, this AS 39523 suddenly is in control of these IPs. So that means, rather than sending them somewhere else, we need to send them there."

The other thing that I haven't mentioned yet is specificity, that is, the way routing tables work is that among a table with many entries, the entry which is most specific is the one that wins. Now, this is important. Take, for example, that government I mentioned. We have a government that owns a huge block of IPs, say a whole Class A. Call it, I don't know, just out of the blue, 200-dot everything everything everything. Then it sells a chunk of its IP space to someone else. So the point is that it is - so in order for that IP space to be useful to somebody else, they have to obviously be able to receive the traffic bound for that IP space.

So in the routing tables of the world, every single routing table in the globe, there is still the general entry 200-dot anything anything anything goes to whatever continent that government is on. But also now in the table, having sold a subset of their IPs, 200.20 dot anything dot anything goes to the buyer. So the point is there are now - the way the routing table works, the incoming packet containing a destination IP is caused to match the most specific rule, the most specific entry in the routing table, in this case while, yes, an incoming packet to 200.20 dot something dot something does match 200 dot anything dot anything dot anything, it more closely, that is, it more specifically matches 200.20 dot anything dot anything. So that's the entry in the table that is the controlling entry, and that specifies where to send the subset of the 200 dot anything anything anything packets.

So here's the kicker is that, as this posting on BGPmon mentioned, the entries, the bogus entries that appeared in this Russian router were more specific. They were deliberately more specific in order to capture the routes and the actual traffic that would have otherwise gone to Google and Microsoft and Apple and Facebook and so forth.

Okay. That was a week ago. That was seven days ago. We've discussed these things in the past, and I want to remind us briefly about them. On Sunday, February 24, 2008, also at BGPmon because these guys have been around for a long time, they said: "For two hours viewers across" - get this - "most of the world were unable to reach YouTube. The root cause of this outage was a flaw in the Border Gateway Protocol which governs how routing data propagates across the Internet.

"It began the previous Friday when the Pakistan Telecommunications Authority ordered the nation's ISPs, that is, the Pakistani Internet Service Providers to black out a YouTube video it feared would trigger riots. Pakistan Telecommunication Company Ltd. (PTCL), Pakistan's largest Internet provider, responded to the order by the government by blocking the entire YouTube site" - in this case YouTube network - "through a quirk in BGP that's readily exploitable by anyone" - listen to that, anyone - "who controls a BGP-enabled router" - which is an edge router or a border router - "of an autonomous system (AS) such as PTCL is. PTCL connects to the Internet solely through another AS, in this case PCCW Ltd., a Hong Kong telecommunications company. PTCL technicians" - that is, the Pakistani technicians - "failed to warn PCCW" - that is, their parent provider, the ones that they connect to - "to block the bogus route."

So essentially the Pakistani Company PTCL deliberately put a bogus route for all of YouTube IP space into their own router so that nobody in Pakistan would be able to reach YouTube. But they did not tell their parent, PCCW, not to propagate that bogus route. Consequently, that bogus route spread through PCCW and out to the rest of the Internet. That route got propagated globally, and for two hours most of the world was unable to reach YouTube. Not from a denial of service attack. Not through anything like crazy, some active powerful attack. Just an engineer in Pakistan entered a command into a terminal and forgot to prevent that known bogus BGP route from going global. And it did, and it created a two-hour global blackout of a major network on the Internet.

Leo: This seems like a huge vulnerability.

Steve: Exactly.

Leo: You could do it maliciously; couldn't you?

Steve: Exactly. So that was February 24. The same summer, attackers arranged to cleverly misuse the bitcoin stratum protocol by deliberately hijacking IP addresses of the pool server IP block. An attacker was able to steal, at the time, $83,000 U.S. dollars' worth of bitcoin. This was an advanced attack, not just from the bitcoin perspective, but also from the BGP perspective. In an attempt to hide the attack, the originator used autonomous system path prepending, which is a way of hacking BGP, with a range of autonomous systems.

And I'll just note that later in that same year, in 2014, it was the same technique of spoofing BGP was used by spammers. An unknown spammer was able to manipulate BGP to cause a block of IPs to get routed to them. While it was routed to them, they used it for spamming; and within a couple hours, before the antispam systems could blacklist their IPs, they shut down that block and opened another. So they were able to walk through a chunk of the Internet's IP space briefly, commandeering it for the purpose of sending out a huge amount of spam which would not be blocked by IP address filters because it was brand new and had never been seen sending spam before. And then before they could get filtered, or maybe the moment they sensed they were being, they shut that down and grabbed another block of space.

So you got exactly the point I wanted to make, Leo, and the reason I wanted to spend some time talking about this is that we have a fundamental weakness in the Internet. And mistakes happen, and abuse happens. Routing errors can happen either deliberately or inadvertently. And we absolutely rely on this. There is continuous BGP protocol noise as routes are changing. On the order of 50 to 200 routing updates per minute are propagating through the Internet, and there are high periods of churn where those who monitor see 9,000 updates of routing information per minute. So it's not something anybody is able to manage or take advantage of. And at this point we're vulnerable.

And so I would not be surprised, the reason I wanted to take the instance of this occurring for a couple minutes suspiciously by a Russian AS exactly a week ago is that I'm virtually certain that during the balance of this podcast's life we will be talking about the apparent deliberate abuse of BGP, probably by state actors who certainly have the power to perpetrate these kinds of deliberate Internet misrouting attacks. Really, really interesting.

Leo: Fortunately, nobody would want to do that. That would be such a terrible thing to do.

Steve: No, that's just theoretical.

Leo: They'd have to have some status, though, some standing in the network. I mean, people don't just blithely accept new BGP rules; right? I mean...

Steve: They do blithely accept new BGP rules.

Leo: Well, that's terrible.

Steve: I know. It is. Yeah. And the problem is you can't - so like there's no vetting system. If that government sold a block of IP space, all it means is that somebody on an edge router somewhere says we're now responsible for this block of IPs. Please send them to us. And it happens. There's no...

Leo: It's not the first time we've seen this. This has happened many times over the past years; right?

Steve: Yes.

Leo: These mistakes, BGP mistakes.

Steve: Yes. And they can be mistakes, or they can be deliberate. And there is...

Leo: We've not seen any deliberate attacks using this though, at this point. Or have we?

Steve: Well, yeah. We, for example, in that case of spammers squatting on stolen - basically they're stealing IP space in order to route spam and then releasing it and then stealing some more. So it is the case that you need to be an autonomous system. But the problem is there's no oversight. There's no clearinghouse. There's no one that you have to check with before you enter a block of IPs into a router, claiming that it's yours, and it propagates across the globe, and now all of those IPs some to you. I mean, it is that bad. It's really, really worrisome.

Leo: It's pretty amazing.

Steve: It is.

Leo: All right. Well...

Steve: And on that happy note, we end 2017.

Leo: The midnight ride of Steve Gibson is concluded. The BGPs are coming, the BGPs are coming. Yes, and next week a fun one. We're going to have a lot of fun, and I'm looking forward to that. But that is a - I don't want to say a rerun, but a return of a classic episode. And not the whole episode, just the timeless part.

Steve: Yes. And it is an important, I think, something that our listeners will really find very interesting. And then we'll be back all fresh-eyed and bushytailed or whatever that expression is.

Leo: Bright-eyed and bushytailed.

Steve: Bright-eyed. Bright-eyed and bushytailed in 2018.

Leo: January 2nd. I don't know how bright-eyed or bushytailed I'll be.

Steve: Yow. Ooh, yeah.

Leo: But we will be here, 1:30 Pacific, 4:30 Eastern, 21:30 UTC on Tuesday, January 2nd. But again, next week is going to be a fun one. And I wasn't here for that episode. Tom Merritt was filling in. So it'll also be the return of Tom Merritt to our airwaves. You must listen.

Now, there are numbers of ways to listen to any given show. Certainly the first place to go is GRC.com. That's Steve's website where he has all sorts of good stuff, including, of course, his day job, SpinRite, the world's best hard drive recovery and maintenance utility. But he's got lots of freebies, and he has not only the audio version of this show, but transcripts, not just of this show, but all of them. You have all the shows, all 642?

Steve: I had Elaine go back and do them all.

Leo: She went back in time, okay. Wow. Which makes it a very searchable archive, as well. So go to GRC.com and get that. You can tweet Steve, @SGgrc. He takes DMs of any length, as well. And you can also find video as well as audio of the show on our site, TWiT.tv/sn, or subscribe wherever you get your podcast. That way you'll get every episode, including next week's special holiday episode.

Steve, I hope you have a great Christmas and a happy New Year, and I will see you in 2018. Wow.

Steve: Will do. And I look forward to hearing about your catamaran adventure.

Leo: Whoo. Can't wait.

Steve: Don't try to get too much sun.

Leo: I think it's going to rain the whole time.

Steve: Ah. In that case...

Leo: So that'll be a problem. But I will be taking my Vitamin D, thanks to you.

Steve: Oh, good. And on to 2018.

Leo: On to 2018. Thanks, Steve.

Steve: Okay, bye.