"MemCrashed" DDoS Attacks

Description: This week we discuss some very welcome microcode news from Microsoft; 10 (yes, 10!) new 4G LTE network attacks; the battle over how secure TLS v1.3 will be allowed to be; the incredible Trustico certificate fiasco; the continually falling usage of Adobe Flash; a new and diabolical cryptocurrency-related malware; the best sci-fi news in a LONG time; some feedback from our terrific listeners; and a truly record-smashing (and not in a good way) new family of DDoS attacks.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-653.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-653-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Great show ahead for you. We're going to talk about the world's biggest DDoS attack, unbelievable. TLS 1.3 is coming. And the little certificate authority that couldn't. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 653, recorded Tuesday, March 6th, 2018: MemCrashed.

It's time for Security Now!, the show where we cover your security and privacy online with this man right here, Mr. Steven "Tiberius" Gibson of GRC Corporation. Hello.

Steve Gibson: This is sounding a little bit like the game show host voice.

Leo: Don Pardo speaking. So hello, Steve. I'm excited because we've been talking before the show about something you're going to talk about a little later on.

Steve: Oh, yes. Yes, yes, yes. In fact, I started by saying I was sorely tempted to name this podcast something, but I won't say, for those who - because it would give it away. And for the last couple weeks there's been sort of this lull after the whole Spectre and Meltdown specter. And but not this week. This one is jam packed with stuff, so much so that for a while I was thinking that I would do a dual-subject, like two main subjects. Then it was, wait, well, maybe three. And, okay, finally I said, okay, no, no. Let's just focus on the biggie because this one is titled Security Now! 653: MemCrashed DDoS Attacks.

Leo: Oh, this was wild. The biggest DDoS in history. Massive.

Steve: Yes. And in fact the record that was broken four days ago was again broken yesterday.

Leo: Oh, it's back.

Steve: We're now at 1.7 tbps.

Leo: Is it terabits? Oh, yeah, I guess it's terabits. Wouldn't be petabits.

Steve: Yeah, terabits.

Leo: Yeah, that'd be crazy. We're talking crazy here.

Steve: So we're going to talk about this. It's got, I mean, this one really has the attention of the Internet because this is an ELE-level event, as they call them, an Extinction Level Event for the Internet. This is, I mean, this is really bad. And it hasn't even really gotten going yet.

Leo: Uh-oh.

Steve: Anyway, we will be talking about that. And it uses, I mean, there's never been a service more designed for DDoSing than this. It's called Memcached, or Memcached - "D" as in daemon. It's service which many Linux systems run. There's a MemCached.com where up until a few days ago they were really happy with themselves. And it's a simple tag-and-value database store that's used for caching. But unfortunately, like so many of these things, it was never supposed to be exposed to the Internet. It was only for internal use. Shodan reports 87,800 and some individual exposed instances of this. And so far we've only seen maybe five or 6,000 of them in use. So anyway, we'll be talking about that later, even though I'm having a hard time restraining myself, obviously.

Leo: It's kind of stunning, I know, I know.

Steve: But in the meantime we're going to talk about some very welcome microcode news from Microsoft, which I sort of gave a hint about possibly happening last week. And the good news is, and it's really good news, we have 10, yes, 10 new 4G LTE network attacks as a consequence of some researchers who took a good close look at the way our cellular network works and found out it kind of barely does. We have the looming battle over how secure TLS v1.3 will be allowed to be, following up on our original coverage of this some months ago where we talked about that. And so I'll bring people up to speed on that.

We have the incredible Trustico certificate fiasco of the last week; the continually falling usage of Adobe Flash; a new and diabolical cryptocurrency-related malware; the best, and thus the challenge for the title of the show, sci-fi news in a long time; some feedback from our terrific listeners; and then we're going to get into, roll up our sleeves and get into exactly what it is that has really got the Internet operators worried as a consequence of a so-easy-to-perpetrate, catastrophically large DDoS attack capability.

Leo: Fascinating. I had never heard that acronym, the ELE. That's interesting.

Steve: Yes, Extinction Level Event.

Leo: Extinction Level Event, yikes.

Steve: That's like the large asteroid heading towards us.

Leo: Right.

Steve: It's like, oh, well.

Leo: It's just extinction we're flirting with.

Steve: That's an ELE, yup.

Leo: No big deal. Nothing to worry about.

Steve: So, okay, I've got really good news for everybody. Really good news. Microsoft has decided to step up and take responsibility for the Intel processor microcode themselves. I mentioned last week that a couple...

Leo: This is instead of Intel?

Steve: Well, okay. So Intel's doing the microcode. But all of the advice had been necessarily to go get the update, the BIOS update, from your system supplier. That's not going to be necessary.

Leo: Hallelujah.

Steve: Yes. I mentioned last week that there were two DLLs in the Windows System32 directory, which were Intel and AMD microcode patch files; that Windows had the capability, much as Linux does, to perform an on-the-fly boot time update of Intel and AMD processor microcode and has in the past. The question was would they do that here.

And it's difficult to imagine they would not because of the nature of our supply chain being as fragmented and unreliable as it is with BIOSes. They need to get microcode into their systems.

So they have released the first of a growing family of updates. It's not being pushed out, at least for the time being, through the regular Windows Update channel, and it's not clear whether it ever will be. Maybe they just want to have it be on-demand for the time being. If you google, and here's the knowledge base number, you want to google "KB4090007." That's I think the easiest way to recognize that as 4090007.

Leo: Warning. Warning. Danger. Danger. What is that sound?

Steve: 4090007. That will take you to Microsoft's announcement of what they currently support. At this moment they're supporting only a couple of their Skylake processors, probably because that's the strongest, most secure firmware that's known to be rock solid. They're both sixth-generation Intel core processors. And only Windows 10 version 1709, meaning the very latest Fall Creators Update version of Windows 10, both on x86 and AMD platforms, so both 32 and 64, but only a small handful. It's like basically two different processors that have a couple different names. No word on Windows 7 or 8.1, although I did read some coverage of this suggesting that Windows 7 was going to still be supported within the window that would cause this to be expected to be supported. So there's hope that there will be updates for Windows 7 which will also be able to do this.

What I'm going to do, because this is tied to hard-to-determine processor specifics, you know, like I have a Skylake I think, but I don't know which one, and there's like a bunch of them, I'm going to give InSpectre, my little tool, a rev in the next couple days, it won't take long, to show the processor ID, the internal processor code which Microsoft uses and Intel uses, but which is not otherwise readily available, as far as I know, anywhere in the user interface. And that will allow people to see which one they've got and then check this table as it evolves to see whether they've got support.

So what this means is that right now, for people with the supported Skylake processors, who also are on the latest Fall Creators Update of Windows 10, whether you've got Intel or AMD, or I'm sorry, whether you've got 32 or 64. I keep thinking AMD, it technically is the architecture of the 64. So Skylake processors from Intel, either 32- or 64-bit versions of Windows 10, you can get this and add it to your Windows 10. And even if you do not have an updated BIOS from your system manufacturer, this will patch the one most problematical Spectre problem, which is that variant 2 of the Spectre vulnerability. And then you'll be good to go.

And what we expect is, as Intel is able to mature and confirm other microcode releases, and it goes through whatever vetting Microsoft gives it, then this list will expand, both in terms of supported platforms and also because, for example, they did support the Meltdown under Windows 7 almost immediately. So they're intending to mitigate these vulnerabilities probably from 7, 8.1, and 10, but then also for other processor architectures as they become available.

And as I said, I'll rev InSpectre just to add a little bit of information about the availability of this for people who show that their system is still vulnerable to the Spectre problem, and then also to reveal the processor ID, the CPU ID, which Intel and Microsoft are referencing in these documents to see whether they qualify yet. And then maybe, we don't know, maybe Microsoft will end up pushing this out. But if anyone wants to get it beforehand, I'm sure after everything we've gone through so far this year Microsoft is not

making this available until they're sure that they've got it nailed. So again, you can just google KB4090007, and that'll take you to the knowledge base article that's got links.

Leo: So I had a caller on Sunday on the radio show. I felt really bad for her. She had an Asus gaming machine she had built, and there was a four-year-old Asus motherboard. And they're probably never going to patch that.

Steve: No.

Leo: This would be in lieu of that; right? This would...

Steve: Correct.

Leo: Well, it won't be for her because this is obviously not going to be a Skylake machine. But if they do this for other processors, that would be enough?

Steve: Yes, yes, yes.

Leo: You don't need a BIOS update, in other words. You just need this microcode update that Microsoft's going to actually provide.

Steve: Right.

Leo: Are they providing Intel's fix?

Steve: Yes. They're getting this from Intel. So Intel provides it to them. And so the way you can think about this is that the processor comes with microcode which, when it powers up, it copies microcode in an internal ROM into RAM for lightning speed access. So there's like an internal microcode RAM store. So that can be overridden. Microcode can be patched by something that has the privilege to do so when the system starts up. That's what BIOSes do when they've got updates to the microcode, so when like Lenovo patched various systems. Dell had a patch for some laptops.

So the BIOS can contain that patch, and then the BIOS is able to say to the processor, oh, I've got some updates for your microcode. And so every time it boots, the processor initially puts the microcode in that it has, but then the BIOS is able to say, oh, we've got some fixes. And so it rewrites a little bit of the microcode in the chip, which then the chip uses for the rest of the duration of it being powered on.

But that doesn't have to only be done by the BIOS. The OS can do it, too. And Linux has the ability to do that. And in fact earlier Intel published a file that Linux users could use to patch themselves immediately. They then withdrew it when they realized they had problems with the patches. And I don't know if it's been reissued yet by Intel for Linux. But Microsoft can do it, too. And so this is the beginning of that.

So with additional processors going forward, what this essentially means is it will absolutely not be necessary if you have Windows 10 Fall Creators Update. We don't know for sure how far back they're going to go. What I've read, because I was trying to get some clear sense for that, and Microsoft isn't saying, but those who seem to know believe ultimately all of the supported OSes will get coverage. And really there's no reason for them not to. It's a simple thing to do, if you're doing it for one OS. It's just, basically, it's a couple DLLs. The patch, they're less than a meg. They're 773K or something because it's not a lot of - it's micro, after all.

So anyway, that's great news. And so for this caller who you spoke to, for example, if she's got an OS that Microsoft is supporting, then either by going and manually getting it, maybe automatically having it pushed, it's not clear yet, but at least right now they are making it available for some users of Skylake.

Leo: That's good. That'll help a lot of people. Probably not her, but it'll help a lot of people, yeah.

Steve: There was, about a year and a half ago, a group of researchers, two guys at the State University of New York in Binghamton and another at UC Riverside, developed a side-channel attack on branch target buffer collisions. It's closely related to everything we've been talking about with this Spectre problem, that is, that this branch target buffer is a cache that saves on computation and speed. So that was a year and a half ago. I was reminded of it.

And the thing that's significant to me, and I knew when I saw this graph, I thought, oh, Leo's going to love this one. On the next page of the show notes, Leo, is a chart showing the timing of using this speculative execution branch prediction when they get a hit versus when they get misses. And it's just - it's beautiful because what they're doing with this attack 18 months ago is they're essentially managing to remove address space layout randomization.

So you've got that on the screen now. You can see along the bottom they are trying different groups of high-order bits for where something is located. And you can see the one that they score on is - so there's sort of a statistical distribution. It's noisy all over. But it ranges from, like, about 45 cycles to maybe 65. But when they hit it, it's suddenly a little over 90. And just one, just sitting out there all by itself, which says, okay, bang, we had the collision that we were looking for.

Anyway, there's really no other news here, but I just love this graphic that was in their PDF because it demonstrates the sort of thing that we see when we get one of these statistical timing-based results is the amount of time that something takes changes dramatically. And in this case, with this research, they were able to, in 60 milliseconds, completely strip all address space layout randomization, either from another user process or from the kernel. So these sorts of attacks have been around for a while. And as we know, well, of course it's been - most of the news of the year so far has been about this. But I just really liked that chart because we hadn't seen something as clear in the previous coverage of that.

So three researchers from Purdue University and one at the University of Iowa took a good hard long look at our current 4G LTE network protocol. They developed a tool they call the LTEInspector to essentially more closely examine the protocol we're all using. All of our cellular 4G LTE network is based on this. In the abstract for their paper they wrote: "In this paper we investigate the security and privacy of the three critical

procedures of the 4G LTE protocol (attach, detach, and paging)..." which are internal names for specific actions in the protocol.

They said: "...and in the process uncover potential design flaws of the protocol and unsafe practices employed by the stakeholders. For exposing vulnerabilities, we propose a model-based testing approach." And then they introduce "LTEInspector, which combines a symbolic model checker and a cryptographic protocol verifier [in what they call] the symbolic attacker model." So basically, they built, they modeled the protocol in this LTEInspector and then ran it through its paces looking for problems.

And they say: "Using LTEInspector, we have uncovered 10 new attacks, along with nine prior attacks, categorized into three abstract classes - security, user privacy, and disruption of service - in the three procedures" - that is, these three, attach, detach, and paging - "of the 4G LTE protocol. Notable among our findings is the authentication relay attack that enables an adversary to spoof the location of a legitimate user to the core network without possessing appropriate credentials. To ensure that the exposed attacks pose real threats and are indeed realizable in practice, we have validated eight of the 10 new attacks and their accompanying adversarial assumptions through experimentation in a real test bed."

So anyway, what we have here is the kind of vetting that it's easy to wish afterwards that this had happened beforehand. But these protocols evolved sort of organically. Things get tacked on. Somebody, a committee says, oh, let's add this feature or that feature. And at first blush it looks fine. Unfortunately, it turns out when you really, really look closely, they've become so complex that it is difficult to make any true security assertions, and it takes careful modeling of the protocol and then a deliberate attack on that model in order to find problems. So they stopped short, they deliberately stopped short of publishing any proof-of-concept code because they don't want to make it that easy. They argue that it's difficult to see how this can be fixed through further patching.

I loved their language somewhere. They said, oh, they said: "We deliberately do not discuss defenses for the observed attacks as retrospectively adding security to an existing protocol without breaking backward compatibility often yields Band-Aid-like solutions which do not hold up under scrutiny," meaning it may be difficult, if not impossible, to fix this.

So what we have is a better sense than we had before, you know, we've talked about problems with our cellular communications, the whole problem with the signaling system which is old now and never had authentication built into it. Well, we've been trying to move the system forward and essentially go back and shore it up a little bit with, as this demonstrates, some mixed results. So what this probably means is that we need to more carefully recognize that the security of the transfer protocol, the security of the transit protocol cannot be relied on.

And so things like Signal and Telegram and Messenger, encrypted protocols that run on top of this questionable transport protocol is what we need to rely on going forward, which would argue for having stronger encryption and security for the protocols that we use on top of this 4G LTE and future cellular protocols. They also note, for example, that faking things like emergency alerts of the kind which caused the hysteria in Hawaii earlier this year, although of course we know that one was a human error, some guy pushed the button next to the one he meant to push. But still...

Leo: No. They found out he was...

Steve: Oh.

Leo: He just - he misunderstood the...

Steve: He deliberately pushed the wrong one?

Leo: He received an alert that said, "This is a drill. This is a drill." And he wasn't paying attention, and he pushed the alert button.

Steve: Oh, oh, oh. So he just didn't...

Leo: He ignored...

Steve: ...hear the "this is a drill" part.

Leo: He didn't understand the "this is a drill" part. He no longer works there.

Steve: Yeah.

Leo: Yeah.

Steve: And you always wonder, you know, in the movies where they say "This is not a drill."

Leo: Right.

Steve: Well, okay, well...

Leo: Sure sounds like a drill.

Steve: If it is a drill, and they don't say that, then you know it's a drill; right? So that never made any sense to me. It's like, here's an alert, and we're serious this time. Until now, eh, we haven't been that serious. But this time really...

Leo: This time we mean it.

Steve: Please pay attention. This is not a drill.

Leo: Please, it's not a drill.

Steve: On the other hand, if it was a drill, and you wanted to actually drill people, that's what you'd say, too. So I don't know, it just never seemed to make much sense to me.

Leo: Yeah, yeah.

Steve: So, okay, speaking of what's not making much sense. Well, although I guess I can understand this, we've seen over and over and over how difficult it is to move security forward, and especially to retire things that have long since had better solutions. We talked some months ago about the next version of TLS, the Transport Layer Security. TLS is the new name for what we used to call SSL - SSL 1 and 2 and 3 and lots of sub versions. And now, you know, 3.0 of SSL was sort of TLS v1.0. There was sort of a - it was a transition that we made. Well, 1.2 was ratified in 2008, so eight years ago. And we're just now to the point where we're beginning to get ready to say, okay, we're absolutely going to depend upon that.

But, as is always the case, we're wanting to create, we're wanting to have the next version ready so that support can begin to get created for it. And that will be v1.3. The 1.3 is many good things, and we'll go over it in detail. I'm sure that there will be a podcast whose title is TLS v1.3 when it happens, when it finally gets ratified, because it hasn't yet. And so they're still arguing over the details and over the features. Overall, it's really a good thing. I'm excited about it because it follows the KISS principle, you know, keep it simple. And also a protocol should be as simple as possible, but not simpler.

So what 1.3 does is deliberately abandon a bunch of old junk which has been hung on and tagging along and supported, just because it once seemed like a good idea. We long since learned otherwise, but it's still kind of there. So it's like, okay, no. No more of this old baggage. This is the baggage removal version of TLS, which is all for the good. But the big controversy, and this is what we discussed before, is that it provides, for the first time, and it's a little surprising to people that we didn't already have this, but it provides for the first time what's known as "forward secrecy," sometimes "perfect forward secrecy," PFS.

What that means is that conversations which are encrypted, as they are under TLS, cannot be later decrypted if in the future something is discovered. Believe it or not, we don't have that today. Even with TLS v1.2, as we've discussed in the past, part of the secret of the key which is negotiated on the fly is the server's private key. That's felt to be safe because it's private, and servers have a huge stake in keeping it private.

So what happens is, right now, up until 1.3, if the NSA, just to pick on somebody - and of course they do have a massive datacenter in Utah with its own power and cooling lakes and massive, god knows how much storage. If all of the traffic everywhere were just siphoned into this monster facility and stored, even though they can't see it because it's encrypted, right, if the handshake of TLS up to v1.2 is recorded, and if at any time in the future, even after the server's private key expires, if at any time in the future the private key of the server which was in use at the time is disclosed, is found, is discovered - or maybe quantumness happens in the future and so you can get it somehow. Who knows? It's possible to take the private key and the handshake observed and obtain the session key, which is what is used to encrypt the conversation. So that's no forward secrecy.

TLS v1.3 is bringing us forward secrecy, which is a huge improvement. I mean, that's good news. The problem is, and this is what we discussed about this before, it's facing some backlash because there have been some ways that datacenters have used the lack of that forward secrecy, for example, to terminate TLS version up to 1.2 connections at their border so that inside the datacenter the traffic is not encrypted. That is, so they've got something on their edge which clients connect to as a proxy, essentially, a proxy for the servers. And then inside the datacenter, everything's in the clear. So that's been a convenience for network guys who are then able to monitor traffic, do intrusion detection easily, see what's going on. They have complete visibility into the traffic. Essentially, in their datacenter, nothing is encrypted, which makes people nervous, but makes the network guys happy because they're able just to see everything, which is convenient.

Turns out that coming late to this is what's known as BITS, B-I-T-S, which is the technology policy decision of the Financial Services Roundtable which is the industry group for financial services in the U.S. All the banks and other high-end financial groups have this roundtable group that they're members of. And these guys are, like, suddenly woke up to the fact that, oh, my god, this might happen. And we don't want to spend any money on our datacenters essentially is what this comes down to. They're fighting it just because, like, they'll have to do something. They'll have to invest in more security, essentially, in order to work around what this means.

So there's been some - and this is where it's really coming down to some push for a mode for TLS v1.3 that would not be forward secret. And the people who want forward secrecy, like everybody else, the people who care about secrecy and security, are saying no. If we make this be a bit you can flip, or if you can have a downgrade to pre-1.3 forward secrecy, then that's what people are going to do, or use.

And so, for example, what that would mean is that the banking industry, where you might argue it would be important to have forward secrecy, in order to save themselves the trouble of changing anything in support of TLS v1.3, they will refuse that mode during negotiation, which means all of the public traffic with those organizations will lack forward secrecy, which arguably is probably where you need a lot of it.

So anyway, this battle is looming. I just sort of wanted to touch on this because the final ratification meeting with the IETF is supposed to occur later this month. The purists are absolutely refusing any compromise, saying as I said, that if there's a way to not do it, those who don't want to invest in what it takes on their side will set that bit. I mean, and it's not like right now anyone has to use it. I mean, it's not even born yet. And so we'll be with 1.2 for a long time.

At some point, for example, 1.0 and 1.1 is now old enough, since 1.2 has been around for eight years, that they're on the way toward being deprecated. I received a note, for example, from my back end that I do ecommerce with, the financial clearinghouse that my own ecommerce system interacts with, just a day or two ago, an alert that this summer they are going to stop supporting TLS v1.0 and 1.1. And so make sure that your infrastructure supports 1.2. Mine does.

And so this is an example of what's happening. Someday 1.2 will be phased out, once 1.3 has been well enough established. Probably not for eight years. So again, if it follows the same time course, as it might - because of course 1.2 is very good. We fixed a lot of the problems in 1.0 and 1.1, so we're pretty happy at 1.2 for now. But ultimately we'll be going to 1.3. What these guys want is for 1.3 to be useful, to be the right thing to move to, and not to have pre-crippled it because some people who were saying, oh, we're going to have to buy some more wires, are being too cheap to do that. So I'm expecting this is going to happen without the weakening provision.

Interestingly, Matthew Green was one of the people who was trying to find a compromise and was rebuffed by the committee, saying, you know, thanks, but we're really going to try to stick with this and get this thing to pass through the IETF as an all-or-nothing, and in this case as an "all" for encryption so that it isn't weakened ahead of time.

Leo: Okay.

Steve: So...

Leo: You know, a lot of times you heave that sigh on the show. And I know, I know something's coming. Something not good.

Steve: Yeah, yeah. So there's a U.K. - or maybe was, I'm not sure if this company's going to survive this - a U.K.-based reseller of Symantec certificates, so sort of purporting to be a certificate authority. They didn't have their own CA infrastructure because of course that's a big deal. To run a CA you've got to have a really good network. You have to really be serious about security. There's all kinds of ways you can screw up, so you have to not do that. But you need to have, for example, run certificate revocation services and have those up and available and reliable. And, I mean, there's a lot to it.

So it's easier for a company to say, as Trustico did some time ago to Symantec, "Hey, we'd like to be a reseller of your certificates." So you go to Trustico, and you say, "Hey there, folks, I'm in the U.K., too, and so are you, so I want to get a certificate from you." And they say, "We'd be happy to sell you a certificate." What they're actually doing is selling them a Symantec certificate with Trustico sort of as the frontend. So it's much easier to resell than it is to actually be a CA.

So the problem, of course, is that as we know, Symantec screwed up, actually with some of their partners, kind of like Trustico. And so Symantec's certificates are going to be declined in the future because we can't really be sure what's what with them. As a consequence, that impacted Symantec and, as we've covered on the podcast, Symantec looked around for somebody reliable and trusted that they could sell their certificate customer base to, essentially. And that turned out to be my favorite certificate company, DigiCert.

So maybe as part of this, in some sort of an awkward way, Trustico decided - and some of this is not clear. I read a bunch of coverage about this, trying to really put the timeline together. But Trustico decided they didn't want to stay with Symantec/DigiCert. They wanted to go to another shining gem, Comodo. So they sent a letter to DigiCert, Trustico did, saying we want you to revoke 50,000 certificates.

Leo: And we know how well certificate revocation works.

Steve: Ah, yes, thank you for that, Leo. Exactly. We've spent - I dragged our listeners through my fury over certificate revocation some years ago, which of course is completely broken, especially for Chrome, which doesn't really even try. So DigiCert says, what? First of all, we're not even sure you as not even quite a CA yourself can ask

us to revoke certificates on behalf of your customers. They're your customers. It's up to them if there's a problem with the certificate to say, hey, whoops, we think we lost control of our private key. We need you to revoke the certificate, please. So DigiCert says, what? No. Like the only possible way we could revoke certificates is if there was a clear breach of their private key. Whereupon some jamoke at Trustico zips and emails DigiCert some 23,000 private keys.

Leo: No. No. No.

Steve: It's unbelievable. So wait a minute. Wait a minute. So not only did they zip them up and email them...

Leo: I hope they put a password on the zip.

Steve: No. They were non-password-protected private keys, as a matter of fact.

Leo: Oh, my god.

Steve: Not only that...

Leo: That's one way to get them revoked.

Steve: But they shouldn't have ever had them. A CA should never have your private key.

Leo: Your private key, yeah.

Steve: Yes. The private key. For example, when I have DigiCert sign a key, they're signing my public key. So the server, a GRC server makes a public key pair, consisting of a private key which never leaves its control and the public key. The public key is what is going to be sent out with every TLS connection. That's the thing that you want the CA to sign so that when the client, the browser receives the public key, it looks at the signature and goes, oh, this is signed by DigiCert. I like them. And therefore I'm going to trust the public key. The point is DigiCert never gets my private key. They don't want it. I mean, nobody wants the responsibility. Somehow Trustico had the private keys of its customers.

Leo: Just in case. It was a convenience thing.

Steve: Well, get a load of this. It turns out that, for convenience, they had a web page where, if their customers were for whatever reason unable to generate a key pair themselves, Trustico, handy-dandy Trustico would do it for them. And yes, Leo, did it in JavaScript, with JavaScript from five or six different sources, including ads on the same page. So nothing has ever been less secure. I mean, you can't make this up. This is just,

as I said...

> **Leo:** Actually your Picture of the Week might have something to do with it. Perhaps there is a less secure way to store private keys. What do you think?

Steve: Yes. The Picture of the Week is just so perfect. This was somebody's response to this debacle which is just wonderful. It shows like some cheesy liquor cabinet. I mean, there's a screw top bottle of red wine on the left and some Band-Aids, just in case you've had too much to drink and you slip and cut yourself. And some dry gin and a half-consumed flask of whiskey, I mean, okay. That's on the lower shelf. On the upper shelf there's this beaten-up box that says #Trustico, and then in big capital letters, "PRIVATE KEYS. PLEASE DO NOT OPEN." Yes, we've stored all of our customers' private keys in this box whose lid is sort of damaged, too. I mean, oh, this is just amazing.

So of course - so no one can believe this. This is just incredible that they would, first of all, have private keys. Then, when challenged by DigiCert correctly quoting the terms of the CAB, the Certificate Authority group, the CAB Forum, saying, what do you mean, revoke 50,000 certificates? And remember, these were not certificates that DigiCert ever issued. Symantec issued them, and they sort of inherited them as a consequence of the Symantec acquisition. So it's like, what? So they say, here, here's a bunch of private keys. They're disclosed now, sucker. So it's like, oh. Now, DigiCert isn't quite sure what to do about the other 27,000 that have not yet been received in email, so there's that problem. So as part of this, DigiCert, doing the right thing, posted to a common forum that they all use about what was going on. This is the Mozilla security mailing list.

Scott Helme, who is an information security consultant and an expert in the CA domain, was quoted in some of the coverage saying: "To arrive at the conclusion that Trustico have been anything other than grossly negligent here is rather difficult. Generation, storage, and the apparent ease and willingness to further compromise the keys are all outrageously inappropriate. They could have trivially proven ownership of those keys without the need to zip 24K-plus of them and send them via email. If these actions were motivated by business/politics, as some suggest, it'd be ironic if their actions resulted in their removal as a reseller."

And indeed it's difficult to imagine how even Comodo could say, yeah, at this point we'd love to have your reseller business. I mean, yikes. DigiCert reported the revocation incident on the Mozilla security mailing list, which we've quoted from from time to time. It's often used to discuss affairs of the CA Browser Forum. And not surprisingly, members of the mailing list had concerns regarding Trustico's access to the private keys. Eric Mill, who's the senior advisor at the U.S. General Services Administration's Technology Transformation Service, posted: "Trustico doesn't seem to provide any hosting or CDN services that would make use of the private key, nor do they appear to explicitly inform users about the storage of this private key, thus suggesting that there was no apparent reason for Trustico to keep copies of the private keys."

Eric also wrote: "The storage of private keys appears to be done without the user's knowledge or consent. Given everything that's known, then regardless of who emailed whose customers when and why, I think it's clear that Trustico compromised those keys and has been routinely compromising customer keys for years. Given that there's no evidence that Trustico indicated any intent to change their business practices, then I believe it's appropriate for all CAs to immediately suspend or terminate their relationship with Trustico," Eric added.

So anyway, DigiCert, of course, was in the middle of this because they were the inheritor, essentially, of the certificates which were originally issued by Symantec via Trustico. So they said Trustico - this is DigiCert said: "Today DigiCert issued the following statement regarding Trustico certificate revocation. Trustico requested revocation of their Symantec, GeoTrust, Thawte, and RapidSSL certificates" - okay, so I forgot to cover that. They were getting more of them than just from Symantec - "claiming the certificates were compromised. When we asked for proof of the 'compromise,' Trustico did not provide details on why they were requesting the immediate revocation." And remember, this is 50,000.

"Trustico's CEO indicated that Trustico held the private keys for those certificates, and then emailed us approximately 20,000 certificate private keys. When he sent us those keys, his action gave us no choice but to act in accordance with the CA Browser Forum Baseline Requirements, which mandate that we revoke a compromised certificate within 24 hours. As a CA, we had no choice but to follow the Baseline Requirements. Following our standard revocation process, we gave notice via email to each certificate holder whose private keys had been exposed to us by Trustico, so they could have time to get a replacement certificate." I imagine there's some scrambling going on at the moment.

DigiCert's announcement continues: "In communications today, Trustico has suggested that this revocation is due to the upcoming Google Chrome distrust of Symantec roots. That is incorrect," DigiCert writes. "We want to make it clear that the certificates needed to be revoked because Trustico sent us the private keys. This has nothing to do with future potential distrust dates." And then they conclude: "The upcoming Chrome distrust situation is entirely separate. We are working closely to help customers with certificates affected by the browser distrust, and we are offering free replacement certificates through their existing customer portals. That process is well underway."

So, wow. As I said, yes, Leo, a heavy sigh at the beginning of this one. This is just, you know, bizarre.

Leo: Really.

Steve: And I don't know anything about Trustico.

Leo: What's the vetting process of setting up a CA? I mean, presumably McAfee had the responsibility to vet these guys, or Symantec. But doesn't ICANN put something in place? I mean, these guys were clowns, obviously.

Steve: They were clowns. And, I mean, to be minting public key pairs...

Leo: In JavaScript on a public page.

Steve: ...in JavaScript on a public page with JavaScript hosted from five or six other domains including ads, is unconscionable. I mean, it just - it boggles the mind. Yet they were doing it until, I mean, the thought I had, and this is following up on your question, is you would imagine that there would be some obligation that Symantec would have had to vet the ongoing conduct of their reseller. But on the other hand, it is due to the conduct of a different reseller that Symantec lost their whole CA business in the first

place.

Leo: Yeah, they obviously don't care.

Steve: So let this be a lesson to other CAs who are essentially extending their trust into third parties by allowing those third parties to resell their certificates. This is what that looks like when it goes sideways, first with Symantec and then with this other reseller. Again, this makes Symantec look even worse than they already did because this was the conduct of another one of their resellers. Wow. So with any luck, Trustico is gone because this will get enough coverage, and enough people will realize, you know, I don't care how cheap these certs are, I've got to run around in circles and worry about being compromised. And now I've got to come up with replacement certificates within 24 hours because - not that revocation is going to have any effect.

Oh, I forgot to mention that DigiCert will be providing the keys to the members of the browser community so that, for example, browsers like Chrome, if it chooses to, and/or Firefox and others, could blacklist those certs. And there's been some discussion of proactively blacklisting them, which is a pain in the butt because this is no small number...

Leo: It's done by hand; right?

Steve: Yeah. Yeah, it's no small number of certs to have to deal with. What a mess. Incredible mess.

A little bit of good news. Flash usage, Adobe Flash usage has declined from 80% in 2014 to under 8% today. There was a keynote speech in San Diego last week given by Google's Director of Engineering at the Network and Distributed System Security Symposium. Her keynote, among other things, showed a slide that I've got here on the next page of the show notes in case you want to put it up on the screen, Leo, showing Flash usage decline from around 80% in mid-2014 down to about 8% today. And it's - I'd have to call it "precipitous." Unfortunately, it looks a little bit like Bitcoin valuation, too. So anyway.

What they did was they looked at the percentage of daily Chrome users who've loaded at least one page containing Flash content per day. So that's the metric. Full-time Chrome users probably who encounter Flash at least once per day has dropped from around 80% four years ago to 8% today. And of course, as we've previously covered, Adobe has announced they're formally planning to give up on Flash. It's funny, I've seen by the end of 2020. But I also, yeah, by December 2020 is when they're saying, okay, no more. So only a few more years, and it's really, really gone.

On the other hand, its usage has gone from, like in the various browsers, Chrome, Firefox, and Edge, from what you might call "enabled" or "enabled and pray" to "click if you dare." So it is now - it no longer runs by default. You have to say, oh, yes, I want to do this whatever it is that Flash is asking. So, I mean, we are seeing updates. I was watching a system update just yesterday, I think it was a Win10 machine, and Adobe Flash updates, like oh, okay, well, fine. At least it's not going to just launch. It's going to say, hey, wait, whoops, hold on, do you want to do this? So that certainly mitigates a lot of the advertising-based and drive-by Adobe Flash attacks that we've seen in the past.

And of course HTML 5 can now play video. And most advertising networks and video streaming portals have already moved away from Flash plugins to pure browser-based HTML5. So it gives an improved user experience. So anyway, someday it will just be a memory, and a long lesson in what can go wrong with the security of a plugin, back from when it was necessary.

Okay. And this crazy, but I'm sure we ran across a story like this in the past. There's something that Palo Alto Networks, a piece of malware that they've named ComboJack. They named it ComboJack because it hijacks the system clipboard, watching for a valid cryptocurrency address to be pasted into the clipboard, and then replaces that address with one of its own on the fly. So that's the "jack" part, the hijack part. The "combo" part is that, believe it or not, it can differentiate and detect the individual address formats for Bitcoin, Litecoin, Ethereum, Monero, Qiwi, Yandex Money, and two versions of WebMoney, both denominated in U.S. dollars and in rubles.

So if you're infected by this thing, and you are wanting to send somebody money; right? They email you their cryptocurrency address, or there's a "donate to" cryptocurrency address on a web page, and you're feeling beneficent so you say, okay, fine. So of course you're not going to transcribe it by hand, so you mark it with your cursor. You highlight it, and you hit Ctrl-C to copy it, or whatever you do, right-click and get a context menu and say "copy." Well, that puts it on the clipboard.

This thing, ComboJack, is watching. It sees the clipboard suddenly containing a cryptocurrency address which it recognizes. It parses it and recognizes it because it turns out they're all a little bit different, enough different that they can be disambiguated. And it has its own accounts in all of those cryptocurrencies. So it immediately replaces it with its own. Now you go to your whatever it is, wallet or wherever you go to send some of your cryptocurrency to that cryptocurrency, never really paying attention to the fact that - because, I mean, everyone glosses over those crypto-looking things. You're not memorizing this thing that you just copied off the web page. So then you paste it into your cryptocurrency payment app and say, here, send them some money.

Well, apparently this scheme is working. It isn't easy to get it into your system, or at least it's a long and involved exploit chain. But it's an exploit chain that we've actually seen before. You receive an email containing a PDF. You open the PDF. Oh, and this particular PDF says it's the scan of a lost passport. So that seems like a...

Leo: Oh, I've been looking for that.

Steve: Yeah.

Leo: Send me that email, yeah.

Steve: Seems like a bit of a stretch. It's like, wait a minute, I didn't lose a passport. I'm just going to delete the email. But it's more like the scan of the traffic ticket that you received last week.

Leo: That's more recently, yeah, yeah.

Steve: The e-ticket or something. That seems more likely to get somebody.

Leo: Your tickets are here. Your flight to Paris...

Steve: Right, that's right.

Leo: Yeah, there you go.

Steve: Yeah.

Leo: I'm opening that one.

Steve: So you open the PDF. It opens an RTF file that it contains, a Rich Text Format file, which in turn contains an embedded HTA object which attempts to exploit a known DirectX vulnerability which then executes a series of power shell commands to download and execute a password-protected, self-extracting SFX file which then installs ComboJack.

Leo: Holy cow.

Steve: I know. I mean, that's - unfortunately, I mean, this is how circuitous this route is to get this thing installed in your machine, but it's happening. It's been found on people's machines. They're thinking, what lost passport? And then before they know it they just can't help themselves. They need to take a look at it to see whose lost passport it is. And then, bang. If they attempt to send cryptocurrency to anybody, it ends up going to the bad guys. So seems like a hard way to make money. But cryptocurrency is the thing now.

Okay. Two more things, then we'll do our last break, and then talk about MemCrashed, as it's being called. So Leo, I was 10 years old in 1965.

Leo: Ten years old.

Steve: Having been born in 1955.

Leo: Just a kid.

Steve: And I guess you were, what, eight, probably, I think.

Leo: '56, so.

Steve: So what happened is that the phrase "Danger Will Robinson" entered...

[Clip] Danger, danger.

Steve: I mean, it's incredible when you think about it. I mean, "Danger Will Robinson," that's just a thing now. I mean, it's a meme. And the other thing I realized was "Does not compute." I don't think that "Lost in Space" was the first time we saw the robot...

[Clip] Does not compute.

Steve: ...waving its rubber, its accordion arms around. But certainly "Does not compute" became another meme.

Leo: Is that where that started? That's amazing. I bet you're right.

Steve: I think so, yeah. And so...

[Clip] I was programmed for more important work.

Steve: Probably asked him for the recipe for a souffl or something.

Leo: Yes, probably.

[Clip] Warning, warning, warning.

Steve: Oh, lord, yes.

Leo: My childhood's coming right back to me.

Steve: So I'm mentioning this because on Friday April 13th, Friday the 13th, April, next month, Netflix releases a 10-episode, what looks like a fantastic reboot of "Lost in Space." There was, what, maybe - how long ago was it, the feature-length movie?

Leo: Yeah, that's right. Was it...

Steve: It wasn't that good. I mean, I watched it, of course. But it was, eh, it was okay, but not great. This looks wonderful. There's a three-minute trailer for Netflix's forthcoming "Lost in Space" series. I will urge any sci-fi lovers in the audience to go find it. I've got the link here in the show notes. It's just shy of three minutes long. You will know that you're going to want to be, I mean, I will be metering those 10 episodes out over the weekend. Luckily...

[Clip] Silence, you mental midget.

Steve: Luckily, Lorrie loves this kind of stuff as much as I do, so we will be sitting side by side, glued to the screen, probably in two five-hour bursts, I expect.

Leo: Chunks, big chunks.

Steve: Friday night and Saturday night.

[Clip] Power on.

[Clip] It's working great.

Leo: I have a bunch of those.

Steve: Oh, and Leo, I remember at 10 years old being terrified. My sister, who is also two years younger, we sort of had our heads buried in our pillows, peering out over them, as like the salad monster...

Leo: Monsters, yes.

Steve: Exactly. The salad monster comes out from behind a rock and like, oh, my god. Anyway, there are no salad monsters...

[Clip] My micromechanism thanks you. My computer tapes thank you. And I thank you.

Leo: It is not - is it a comedy? Because that was kind of a comedy. This looks pretty serious.

Steve: No, no, no. No, no, no.

Leo: Good-looking robot. Man, that's a nice robot.

Steve: Oh, it's a beautiful robot. Maybe you should - it's only three minutes. Let's go ahead. Even though our listeners, those who are only listening will only hear the audio, you'll get a sense for it.

Leo: It's dramatic.

Steve: Let's run it into the podcast.

[Begin "Lost in Space" trailer]

Leo: So this is the ship. The Robinson family is on the ship, and they are now stranded on a planet, and there's the little Will Robinson, exploring the planet for the

first time.

Steve: But, I mean, beautiful special effects.

Leo: Yeah, really, it looks [crosstalk].

Steve: This is world-class sci-fi.

Leo: There's Penny.

[Trailer plays]

Leo: Oh, so the robot isn't their robot. It's an alien robot.

Steve: I don't know.

Leo: We don't know. We'll find out, won't we.

[Clip] Danger Will Robinson.

Leo: Oh, yeah, baby. Oh, my goodness. Oh, there goes their ship.

[Trailer plays]

Leo: So the premise of this originally was it was kind of the Swiss Family Robinsons.

Steve: Yes.

Leo: But it was sci-fi. They were on a planet. So where's Dr. Smith?

Steve: So Dr. Smith is...

Leo: So we got Mom, we got Dad, we got Penny, we got Will. Oh, is that Dr. Smith? There's another guy.

Steve: There was like a younger male.

Leo: Oh, there's Penny. There's what's-her-name. She's the new Dr. Smith?

Steve: Yes. We have a female Dr. Smith.

Leo: Oh, and I like her.

Steve: Yeah, whose agenda we're not quite sure about.

Leo: Oh.

Steve: Just as the old Dr. Smith. We really didn't know what his deal was.

Leo: Yeah, we're not sure. He might have been the first gay guy on TV, actually.

Steve: And I think he was - he wanted to sabotage the mission, but he got trapped onboard.

Leo: He was a bad guy, yeah.

Steve: Yeah. He was a bad guy, but I think he got trapped onboard and wasn't at all happy. And he was also really annoying.

Leo: He was kind of prissy and [vocalizing]. But he was kind of the comedy relief; right?

Steve: Well, he was always screwing things up. He was a fly in the ointment.

Leo: Yeah.

[End trailer]

Steve: And it was like, just kill him already. But no, they never did.

[Clip] Never fear, Smith is here.

Leo: Oh ho.

Steve: Oh, god. Anyway, so a few weeks from now, oh, boy, it looks like we have 10

hours of wonderful fun.

Leo: That's going to be a lot of fun, yeah. The movie was not good, but this might - but they know there's a burden because there's a lot of people our age who have fond memories.

Steve: Oh, flush those. I don't care about those. This looks wonderful.

Leo: "Lost in Space," the reboot.

Steve: I just want it to be good so we get more, in the same way that we get more "Stranger Things" and we get more...

Leo: We were disappointed by "Altered Carbon," so this is a chance for Netflix to make good on that.

Steve: Yeah, this looks like a healthy family sci-fi drama. So Rick James in Toronto, Ontario, Canada send me a note mostly, he said - he said: "Security Now! ASA vulnerability follow-up." Remember that the ASA/VPN, that was the Cisco vulnerability that was initially not - it didn't look like it affected as many people as it looked like it was going to, but later it turns out that many more of the subsystems had this problem.

So anyway, Rick wrote: "As I'm sure you and your listeners are aware, Cisco requires its customers to have a support contract to get ASA/VPN software updates. I wonder if this is a circumstance where Cisco NEEDS [all caps] to release a patch like Microsoft did last year to Windows XP," meaning that it's so bad that you should be forced to have a maintenance upgrade. And I think Rick is exactly right. This makes sense. This is clearly a bad vulnerability. He says: "Are software/hardware vendors only responsible to fix their security issues if you're a current customer? I thought this might be an interesting discussion since we can't talk to Cisco about it."

And then the point of this, well, the thing that brought it to my attention, or the reason I'm bringing it to everyone's attention since we've sort of discussed this already, is: "PS: I had a 2TB drive 'die' in my Drobo Mini last week, and guess what. I ran a Level 4 [meaning SpinRite] on it; and, voila, all is good. SpinRite saved the drive." He says: "Love the show and your efforts," and then he says, "cough, bootable Mac SpinRite, cough." And so, yes, that's, as we know, that's on the way as soon as I get SQRL behind me. And that's close to happening, too. So thanks, Rick, for reminding our listeners that, even though you have a Drobo, you can still have a drive go bad. SpinRite can bring it back for you.

Leo: All right, Steve. Let's [crosstalk].

Steve: Okay. So a little bit of closing the loop with our listeners. Regarding the bill that was threatening to pass in Georgia to outlaw anybody who accessed a computer without authorization, Joel Odom said: "The bill in Georgia that you mentioned on Security Now! is effectively dead, thanks to the hard work of EFF and the security community such as

those of us at Georgia Tech. Thanks for bringing it to my attention via the podcast." And it turns out Joel is a Georgia Tech computer security researcher. So good news that that didn't get off the ground because that was just - we can't have that precedent. That would really be bad.

And so Ned Griffin said: "Hey @SGgrc, enabling Quad9 DNS on my router, but the few discussions I've looked are pointing people to changing in their computer setting. I have control over all PCs on my network, so am I right to assume the change at the router is okay?" I know we've sort of confused things in the last couple weeks because we've talked about all the variations on that. So I wanted to just clarify for Ned and anybody else, yes, the normal default for Windows machines, and actually computers in general, iOS devices and Macs and Linux boxes, everybody uses DHCP, meaning that they ask the router, which is this DHCP server, for the DNS address.

If you then set the router so that it provides the Quad9 DNS IPs, all the systems in your network will just synchronously get it next time they boot. And at least in Windows, if you inspect the IP addresses, you can if you want to bring up a console window and do ipconfig, I-P-C-O-N-F-I-G, and you might need to say /all. I don't remember, if you don't say /all, if it shows you DNS addresses or DNS IPs. But definitely if you say "ipconfig /all," that will give you a dump of the current IP configuration, including what IP addresses for DNS your machine is currently using. That's just a quick way of verifying that your system is getting them from the local network's router, and that is the default, so it almost certainly is unless you've done something to manually override it.

Jared Komoroski said: "@SGgrc Your description of how Bluehost could securely use the last four [meaning four characters] of a password to verify account ownership is exactly correct. They store two hashes at password creation. I confirmed with some Bluehost quality assurance engineers that I know. Keep up the good work." So I guessed correctly. It made sense that that's the way you would do that. It's kind of the only way you can do that is to capture it at the time that it's being hashed.

So, and I forgot to mention that this looked like - I saw the screenshot that I had referred to. I looked at it again after last week's podcast. It looked like somebody was online in a text conversation, like on a web page, asking for the last four of the password, just for ad hoc identity verification. So that's what the application was, was somebody saying, hey, to make sure you are you, just give me the last four characters of your password.

Which, you know, it's not super secure. As I said, it does weaken the rest of the password because it's easier to guess the last four, very much like you're able to guess the two halves separately. But still, as a simple way of asking somebody something that they would know, that somebody else wouldn't know, and that you wouldn't have to separately write down, like, the name of your favorite pet or your first pet or your favorite teacher in high school, who knows. That's a nice way to do it.

And finally, two people on pronunciation of German or Germanic names. Martin Badke said: "Speaking long ago with a German interpreter, she explained, 'In German, when two vowels go walking, the second does the talking.'"

Leo: Oh, that's handy. Good to know.

Steve: As a teaching phrase. And then he says: "D-I-E-B-O-L-D is said with the 'E,' thus Diebold." And he says: "Consider Einstein and Siemens." And that's interesting because

it's E-I-N, ein, so there you're getting the second vowel, the "I." And with Siemens, S-I-E-M, you're getting the second vowel, the "E." And then also adding to that, MidwestGuru said: "Diebold" - and I'm now working to pronounce it correctly - "is a Germanic name. When 'E' and 'I' are together, pronounce the second one." He also used the example of Einstein and Dietrich, where again D-I-E-T-R-I-C-H, Dietrich. So anyway, thank you, everybody, for your thoughts and feedback.

The picture I have in the show notes is of yesterday's re-record-breaking attack, showing basically a timeline of prior attacks. One, and I can't quite read it, looks like maybe it's 2001, can't see it. But anyway, it's 24GB, that it was like, whoa, 24GB, oh, what? And then a few years later, 100GB, Leo, oh, my god, 100GB. And then a few years later, 309, oh, three times more. More than three times more than 100, 309. A few years later, now we're up to about, looks like about 2016, 650GB. Okay, wow. Now, yesterday, yesterday, 1.7, okay. So I could say it as 1700 gigabits per second, or 1.7 terabits per second is where we are, an Internet-melting DDoS attack.

As I mentioned at the top of the show, this is the result of a devastating service being exposed publicly. What's really curious is that this had been noted a couple years ago in some of the Black Hat/DefCon-style conferences. It might have been one of the two of those. Someone just sort of noted in passing, and it may have been that no one thought anyone was crazy enough to have this publicly exposed.

But there's a service on Linux platforms called memcache, or because it's a service, memcacheD for daemon, meaning that it's running in the background. It is a very simple, very fast cache, the idea being that, because you might have a very complex SQL database query, after performing a complex lengthy SQL database query, you might want to do some short-term caching. So it could be in RAM, or it could be on the hard drive, depending. But the idea being that you've performed some extensive query. You may need it again. You don't want to go back and reissue the query. So you use sort of this intermediate cache where you use a key or a tag and the data.

So it's very simple. You can't do any fancy queries. You can just say, "Here's a blob. Store it under this index." And so this memcache says okay, and it is accessible over both TCP - and, yes, I did say both - both TCP and UDP. Well, we know what that means, if it's publicly exposed. The problem with UDP is there is no provision for verifying the source IP of a query.

As we've often discussed with TCP, there is the famous three-way handshake: a SYN for synchronized goes in one direction, a SYN with an ACK acknowledging the receipt of the SYN goes in the other direction, and then finally an ACK acknowledging the receipt of the second SYN makes the third interchange of the three-way handshake. An interesting side effect is that both endpoints have to tell the truth about their IP addresses in order to receive each other's acknowledgments. Otherwise it doesn't happen. So TCP cannot be spoofed.

UDP is used for lightweight Internet traffic where you simply say "give me something," like give me an IP address for DNS, and DNS is UDP. And DNS has been used in so-called amplification and reflection attacks. It's amplification because a tiny query can generate a large response. So what that means is that an attacker is able to send little DNS queries to a DNS server, which will respond with much larger replies, thus amplification. But because it's UDP, nothing prevents the instigator of the attack, the person sending the UDP packet, from lying about their IP. They put the IP of the intended victim, the target, in the UDP packet so that the DNS server that receives it believes that guy was asking for a DNS address. And so the larger response goes to the victim of the attack. Thus the attack is reflected off of the DNS server and amplified, a reflected amplification

attack using UDP.

Well, there have been a bunch of different types of UDP services exposed and used over time. They all pale in comparison to this one. DNS kind of has to be exposed to be publicly used, but a DNS server normally is not used by the public. It's normally used by, for example, an ISP's own clients. So it's possible, even with DNS, to lock it down so that it will not respond to random public queries from the Internet because that's where an attacker would be trying to bounce their bogus query off the DNS server in order to attack somebody else. So you can lock down DNS.

This memcache normally would never be publicly exposed. I mean, you wouldn't want it to be publicly exposed. The idea is that it is used as an intermediate cache for things that an active system is doing. Okay. So the first attack, the first big attack that got everybody's attention was about four days ago, and it was an attack on GitHub, not because probably anybody was mad at GitHub, but just they wanted to attack somebody big. And so this was a 1.35Tb attack on GitHub.

In a post in their engineering blog, GitHub said: "The attack originated from over a thousand different autonomous systems, across tens of thousands of unique endpoints. It was an amplification attack using the memcached-based approach that peaked at 1.35 tbps via 126.9 million packets per second." So these are large packets. They're full-size, 1,400-byte packets, because this is this memcache trying to answer somebody, it thinks somebody's valid query for data that it has cached.

Leo: You know what impressed me was how quickly it was mitigated.

Steve: Yes.

Leo: That blew me away.

Steve: Yes, yes.

Leo: And that's Akamai. That's Akamai's DDoS protection.

Steve: Yes. They have strong DDoS protection. And then yesterday we saw an even bigger one that broke that record. That was the biggest ever seen, 1.35 tbps, four days ago. Now we're at 1.7. So, okay. What is this? What's going on?

So there is a site where this software lives, MemCached.org, M-E-M-C-A-C-H-E-D dot org. And they say, introducing themselves, "What is memcached? Free and open source, high-performance, distributed memory object caching system, generic in nature, but intended for use in speeding up dynamic web applications by alleviating database load. Memcached is an in-memory key value store for small chunks of arbitrary data - strings, objects, whatever - from results of database calls, API calls, or page rendering."

Leo: A lot of PHP sites use it. My old WordPress.org PHP-based site used it.

Steve: Well, Leo, get this. YouTube, Reddit, Facebook, Twitter, Wikipedia, Microsoft Azure, IBM Bluemix, Amazon Web Services.

Leo: Oh, boy.

Steve: I mean, yes, it is a highly used, very popular tool because, as you said, I mean, it's there, and it allows, you know, there's very often a slow process to pull something together which may be re-requested. And so what happens is you're able to set a certain size of the cache. It is an MRU, a Most Recently Used cache. So if you keep asking for something over and over, that stays at the front of the list. Something that doesn't get re-requested ends up getting pushed off the end, getting pushed out of the cache in favor of more recently requested things, and those things tend to stay there. So, I mean, if you've got lots of RAM, and you can commit a chunk of RAM to this in-memory cache, then it can speed up your system.

The problem is the server is normally meant to be bound to the localhost IP. Oh, and it's running on port 11211. So it's a well-known port. That's where the service lives. And it's supposed to be bound to port 11211 on the localhost IP, so only on 127.0.0.1. But in some instances where you have a cluster of systems, maybe you do want to make it more widely available. So it is unbound from that particular IP, and it operates on the interface that the system is using. Unfortunately, some of those interfaces, those network interfaces are public. And a recent query of our old friend Shodan reports 87,811 open memcached servers.

The final picture in today's show notes shows the result of a search for the product memcached returned by Shodan on February 26th showing 25,000, the majority of them, 25,000 of them in the U.S.; just shy of 20,000, 19,647 in China; 4,000-some in France; 3,500, little more than that in Japan; almost 3,400 in Hong Kong, and so on. So there are 87,000 of these on the Internet. And what happens is the bad guys are able, because it's an open server, you're able to fill them remotely also.

So the remote attackers are creating single huge blobs so that a single query is able to return like a megabyte of data. They're able to return massive queries. So not only are they open - UDP thus can be spoofed - but they will accept for caching from the public anything that you want to store there subject to cache expiration. But if it's a large blob, a megabyte blob, and then you are constantly asking for it over and over, you're going to keep it from expiring out of the cache, that is, you the attacker. The attacker is going to keep it from expiring out of the cache by continuing to ask for it while they're attacking somebody. So this is not good, Leo.

Leo: Yeah.

Steve: This is not good.

Leo: So this is not an exploit in the sense that there's a flaw in the code. This is actually how it's designed to work.

Steve: Yes. Yes. But it was never designed to be public facing. That's the problem. By no means would you want to offer your RAM on your server to the public.

Leo: So these are misconfigured memcached servers.

Steve: Eighty-seven-plus thousand of them, yes. Misconfigured. And notice that your own system's performance will drop if the bad guys use your memcache RAM for their own purposes because your valid pages and so forth won't be cached any longer. They'll just get pushed out by this blob which is being used to attack somebody else. So it's not in your interest to have them exposed as using your bandwidth, which you're probably paying something for. And it's using a lot of bandwidth. But it's not clear how we're going to fix this. Once again, notice that the only way these attacks can occur is if an ISP is allowing packets not bearing an IP on their network to egress out onto the public Internet. Once again...

Leo: There you go. There you go.

Steve: We've discussed this. It's the lack of egress filtering. There is absolutely no reason why UDP packets should be lying about their source IP as they leave an ISP's network. Simply blocking those packets would cause them to be dropped, and then that network could not be used for attacking. Now, of course, the argument is, well, yeah, but everybody else's could. It's like, yes, but we have to start somewhere. And unfortunately no one is preventing IP address spoofing. It's just anybody pretty much can do it. I've seen some reports of some ISPs doing it, but obviously not all.

So, boy, we have a massive new capability. And it's not clear, I mean, the problem is also there's no authentication. This is a "get a packet, send a blob, get a packet, send a blob." No authentication on this protocol. Again, the idea being it was never meant to be public. It was meant to be an internal use tag and data store for quick storage and, I mean, the whole point of it always is to be fast. It can run over TCP, but it also answers to UDP. And it does so with devastating effect. So, wow. I have a feeling, I mean...

Leo: The mitigation is first and foremost people should just fix their freaking memcache servers.

Steve: Yes, yes, yes. Oh, now, here's the good news also. The IP of the inbound attack, that's the actual IP of the memcached server. That is, the bad guys can spoof their IPs, but the attack traffic...

Leo: Right. It doesn't, of course.

Steve: ...doesn't, exactly.

Leo: It's the server, yeah.

Steve: So, yeah, so it's the IP of the server. So it is possible to identify and potentially to go knocking on someone's door saying, hey, by the way, do you realize...

Leo: Could you fix your freaking servers, you morons?

Steve: Yeah, could you please bolt down your firewall, block port 11211, because you're attacking people all over the Internet. And it would behoove people to do that, too. So it's in their best interests.

Leo: Yeah. But also...

Steve: On the other hand...

Leo: Yeah, I mean, it sounds like also their Internet service providers should stop allowing outbound traffic. Actually, no, because the outbound will appear to be from the memcached server. So they can't do that; right? The outbound would be from a legit IP address.

Steve: Yeah. Let's see. I think you could argue that there is no reason for memcache...

Leo: Outbound UDP packets.

Steve: Yeah, exactly. There's no reason for any memcache traffic to transit outside in the world.

Leo: Right, right.

Steve: Yeah. Wow.

Leo: Very interesting stuff. Now, what do you make of this fast mitigation? I mean, that's one thing we've gotten better at.

Steve: Yes, yes, yes, yes.

Leo: Is fixing and surviving DDoS attacks. What, they just throw a lot of bandwidth at it; right? I mean...

Steve: Well, remember that they know that the traffic is going to be inbound from port 11211.

Leo: They block that port, got it.

Steve: Exactly. So if they're a huge network, and they've got automated border controls, they can see a massive incoming traffic coming from port 11211 and just say, okay, block that. Shut it down.

Leo: Nope, nope, nope.

Steve: But still, no small company is going to be able to do that. I mean, and this is very burdensome. That's a massive attack.

Leo: No kidding.

Steve: And now that this news is out, there's nothing to prevent all the botnets, all the IoT malware, I mean, you can't even - it's not a remote execution attack, so you can't infect something and then close the back door behind you. These are open servers that anybody who wants to can find and use.

Leo: Wow, wow, wow.

Steve: Until this problem is mitigated. So that's not going to happen anytime soon.

Leo: Steve, another fascinating episode. I love, you know, of course all the things you do are great. But I always love hearing the mechanics of these kinds of very big news story attacks, and why it's a problem, and the mitigation. How simple it would be for people to fix it if they just would sit up and pay attention...

Steve: Yeah.

Leo: ...to what they're doing. As always, very informative. That's why we want to all make sure and tune in every Tuesday for Security Now!.

Steve: Why they keep me around, Leo.

Leo: 1:30 p.m. Pacific, 4:30 Eastern. We spring forward this weekend in California. So as we enter daylight saving time, that means we're going to record what will appear to be an hour - oh, no, math is so hard.

Steve: Danger, Will Robinson.

Leo: Our clocks go forward, but UTC does not. So it will appear that we are recording an hour earlier. All I know is we're going to UTC -7, which means 18:30 UTC. You can do the math.

Steve: And then something about losing an hour of sleep. I'm not sure where that happens.

Leo: We do that, as well. I hate that.

Steve: But go to sleep earlier and then you won't.

Leo: I hate that. Tune in if you want to watch live to TWiT.tv/live. And you can also, by the way, join the chatroom, always a lot of fun in there, irc.twit.tv. They're talking about what's happening on the live stream, so the conversation will make more sense if you're watching the live stream. You can watch on-demand, of course. Steve has full beautiful transcripts, along with the audio version of the show, at his site, GRC.com.

And while you're there, it might behoove you to pick up a copy of SpinRite, the world's best hard drive recovery and maintenance utility, and all the freebies Steve offers. And learn about SQRL because it's coming soon. And, oh, there's so much stuff there. It's actually a treasure. It's one of those places, the web used to be full of this, where you would find a site and spend hours just reading all the stuff there. It's like that, GRC.com.

We have audio and video of the show. I don't know why you'd want to watch it, but you can. Oh, I know why. For the trailers, the movie trailers. There is always a great - no, TV trailers. There's always a great TV trailer on every show. Go to TWiT.tv/sn for Security Now!, or subscribe on your favorite podcast application. We're there. We're everywhere. This show's been going on for a long dang time, 653 episodes.

Steve: Coming up on the, well, not quite yet, but the end of - we're in year 12 now.

Leo: Amazing, amazing, amazing. Thank you, Steve. Thank you, everybody, for watching and listening. And we'll see you next week on Security Now!.

Steve: Thanks, Leo.