Certificate Transparency

Description: Leo and I discuss the week's major security events, focusing on this month's crucially important Microsoft MEGA Patch Tuesday updates which, if exploited, will allow for wholesale remote client and server code execution and takeover. They then take a first pass look at the new "Certificate Transparency" standard and initiative being launched by Google and currently supported by DigiCert and others.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-481.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-481-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We've got a big show for you. His planned topic, transparency around certificates, an interesting way to find out if the certificates your site is using are real, if they're from your site, and what other certificates might be out there from somebody pretending to be your site. We'll also talk about Microsoft's Patch Tuesday. This may have gone unnoticed, but one of the patches may be one of the worst vulnerabilities ever. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 481, recorded November 11th, 2014: Certificate Transparency.

It's time for Security Now!, the show that covers your security and privacy online with the guy, the man, the Explainer in Chief, Mr. Steven "Tiberius" Gibson. Hello, Steven. Good to see you.

Steve Gibson: Hey, Leo.

Leo: I'm going to say this right upfront because they ask this every time in the chatroom. What are those blinky lights over your left shoulder?

Steve: Okay. Those are clones of the very first minicomputer, one of the very first minicomputers I ever programmed. The venerable DEC, the Digital Equipment Corporation, PDP-8.

Leo: Yeah.

Steve: And some years ago, the very end of life of that minicomputer, which itself was like a sort of a small refrigerator-size thing, I mean, the original one was just - not even integrated circuits. This predated the availability of ICs. So this was the so-called "discrete" components, transistors, resistors, and capacitors, mostly transistors and resistors. There wasn't much need for capacitors back then, in the signal path, anyway. And so it's a 12-bit minicomputer that has three bits for the operation code, meaning that sort of nominally eight different instructions, except that the math has more power in the individual bits than the rest of the instructions.

So at the very end of life they made a chip. They put the entire minicomputer into a single chip. And some guy found some of those available, like on the surplus market - and I've got mail coming in here, distracting me, there - and made a kit that a number of people built. And of course one wasn't enough for me. I had to have three. So there's three PDP-8s.

Oh, and I forgot to say that what the blinking lights are is the whole point of the old-style minicomputers was that they had front panels. They had so-called lights and switches, blinking lights and switches, sometimes known as the "blinkin' lights panels." So I wrote some software to show that off, that is, because this is the way computers back then looked like when they were actually doing something was that they sort of had a sort of a sporadic changing. They weren't just like blinking, like randomly in a constant pattern. And so I attempted to write some software that made them look like they were actually busy. So nothing's actually happening back there at all. But it's fun.

Leo: What's the clock speed on those?

Steve: Boy. Ah. I should know that. That's...

Leo: It's not very fast.

Steve: But, yeah.

Leo: But you can't compare it to a kind of a standard microprocessor, either.

Steve: Oh, lord, no. I know that some of the cycle times on the core memory, because that's what they used, was like 1200 microseconds, so 1.2 milliseconds. And a millisecond would be a thousand instructions per second. So that's right around - because there were also some cycle times of 800 microseconds, which would be a little faster than a millisecond. So right around the average, right around the neighborhood of a thousand instructions per second. So when - and these, for example, did not have any hardware multiply or division. And multiplication and division are famously and sort of paradoxically difficult things. I mean, they're time-consuming to do. They're iterative. And essentially you take, in the case of, for example, a 12-bit computer, if you multiply two 12-bit values, you get a 24-bit result. So the act of multiplying involves lots of shifting and bit testing and summing. And you need to do double-precision math where

you have to take the carry from the output of the first half and then move it into the second half.

The point is that just doing something simple, like we would think of simple, like a multiply, which contemporary chips just all do instantly, you know, they've got dedicated hardware where you just give it the two numbers, and it says here's your answer, that takes a long time. And when every single one of those processes is a thousandth of a second, before long you've used up a second, just to do something simple.

So those, I mean, what I miss about those days is you had very little memory. 4K words, 4,000 words was sort of like the starting point. And since it was 12 bits, that is 4K. You know, 10 bits is 1K. So 12 bits is 4K. So then, I mean, so that was the normal, default amount of memory is 4K words. And they had a BASIC interpreter running, you know, interpreting the BASIC language in 4K. Well, I wish somebody would say to me, "Steve, it's really important for us to have one," because I would just love to do that. But no one's asking for that anymore.

Leo: Apparently not.

Steve: No.

Leo: Why not? You'd think they'd be...

Steve: That's just my strange nature. The best I could do now is program everything in assembly language which, yeah, which everybody already thinks is a little...

Leo: You're already a crackpot. Don't push it any farther.

Steve: I'm out there.

Leo: I didn't mean to sidetrack you. But about every eight episodes you should probably just mention that.

Steve: Yes, perfect. Give me a reminder when we need to just explain what all that is back there going on.

Leo: And a couple of times during the 10-year tenure of the show - I like that, the 10-year tenure - you have mentioned that these kits are available and - because the guy has to silk screen them in a batch and blah blah blah. But so currently not available, these PDP-8 clones.

Steve: Right. They have come and gone. Sometimes you'll see somebody parting with one on eBay, you know, with a tear in his eye. And, I mean, and I have some of the original PDP-8s and 11s and, like, reel-to-reel - they had a beautiful little tape drive called DECtape, which used cute little reels. And it used a mylar technology where it was

mylar on both sides containing the magnetic material, rather than traditional tape that was the oxide on one side of the mylar. And those have not stood up well over time, whereas DECtape has. Anyway, someday my plan is to bring all that stuff back to life. I just, you know, if no one's going to ask me to do it now, I'll give myself the project when I'm 70.

Leo: We do have quite a bit to talk about today. You're going to do what you call a propeller hat episode.

Steve: Yeah. There's - when I - I settled down to think, okay, what do I want to talk about this week? And I didn't want to do another Q&A because we've had two in a row. It was a relatively quiet security period during the last week.

I do want to talk about something really fun that happened for me. I snuck off to Las Vegas and presented SQRL for the very first presentation of its life ever, at a security summit in Las Vegas, and encountered the lead architect of SQRL's main competition, the FIDO project that we've talked about.

Leo: Oh, you talked to them. Oh, good.

Steve: And blew his mind.

Leo: Oh, boy. Good. Even better.

Steve: It really, really made the whole thing way worthwhile for me.

Leo: Awesome.

Steve: Oh, I really enjoyed - and the conference was hosted by DigiCert. But anyway, I'll talk about that in a second. The topic for today is something that's been in the air for a little over a year. And this is another of Google's many initiatives, you know, largely which - most of which I support. I mean, as we know, I chafe a little bit when they decide that some of the things they want to do, or I don't feel like they're being completely forthcoming, you know, the CRLSet controversy of course comes to mind. And then I was a little annoyed at them arbitrarily pushing forward all of the expiration of SHA-1 certs just because they wanted to. But there are other things, like we talked about SPDY. They're working on protocols to smooth and enhance and speed up our experience using the web. And obviously they have a huge commitment to that themselves.

Today's topic is something known as Certificate Transparency. And there's a website, certificate-transparency.org, which is like the formal place where this lives. There's an RFC 69 - I can't remember what it is. I have it written down. We'll get to it later. But the point is this is Google's initiative, which is underway. There is, as I said, there's an RFC submitted last July, July of 2013, to augment and fix the problems with the current certificate hierarchy. The whole certificate authority trust, you know, root trust problem that we talk about all the time.

So there are, to really get into the implementation, I haven't figured out how to explain it in an audio podcast without lots of diagrams. But to give everyone a sense of it, that we can do. And that's really enough for now because I'm not sure if it's going to happen. I mean, to listen to Google, they're going to make it happen because they're going to require it for EV certs starting early next year, like I read February of 2015, which is not long from now.

But anyway, so that's today's topic. We'll talk about that. And we have to talk about this is the Second Tuesday of the Month. And as I was saying to Leo before he hit the Record button just now, I was just sort of rummaging around through the Patch Tuesday, and the third one just poked me in the eye. It's like, oh, my goodness. I'll be making a trip to my datacenter this afternoon.

Leo: Whoa.

Steve: Like, wow. And so will every single person running a web server with IIS. And even Windows users who are not having servers exposed have to do this. So we've got a big Patch Tuesday. We've got, oh, and a surprising event in the Net Neutrality struggle occurred yesterday that I'll chat with Leo about. I want to talk a little bit about my trip to Las Vegas and what happened. And an important Belkin firmware router update. So another great podcast, I think.

Leo: Can't wait. All right. We're going to put you back to work, Steve. You've got your eye drops in. You're ready to go. I saw that.

Steve: Yeah. It's just they were a little bit dry, and I thought, oh, a little eye drops.

Leo: Yeah. Well, it's a good time to do that.

Steve: Now they're a little wet.

Leo: Well, you're back from Vegas, and there's nothing drier. I'm going to tell you, I go to Vegas, and I come back with a sore throat every single time.

Steve: Oh, my goodness. I know, I was chapped.

Leo: Yeah. Chapped. Chapped, I tell you. So let's do the security news first, and then we'll go on with the certificate transparency stuff.

Steve: Yeah, of course. So, okay. So it is Patch Tuesday, the Second Tuesday of the Month. And as I said, I was - oh, and we already knew that it was going to be large. Microsoft sends out sort of a, I don't know why, but sort of a get-ready email a week before. They're don't tell you what you're getting ready for, but just sort of an overview of, it's like, here it comes, don't forget, Tuesday. So we knew a few days ago that this was going to be a biggie.

And it is, I mean, just independent of severity, in terms of quantity, this leaves nobody wanting. Sixteen individual patch bundles where we often have three. This time 16. And in scanning through the CVE list, I counted 33 known vulnerabilities that were encompassed within these 16 bundles. Except then I looked, and the one for IE - of course there's always one for IE - it alone had 17 privately reported vulnerabilities that it was fixing. So then I'm thinking, okay, maybe I didn't count correctly, but - because it seems like that would lead us to more than 33. But, you know, typically the bundles all handle multiple vulnerabilities rather than just one. But, you know, this one IE bundle had 17, so.

Anyway, so the third one down was MS14-066. And it's like, okay. And I'm reading what it says. "This security update resolves a privately reported vulnerability in the Microsoft Secure Channel" - Schannel is their acronym for it, or abbreviation, Schannel - "security package in Windows." And it's worth noting that Schannel is in all Windows forever. I mean, it was an original component of NT. It's where all of the security stuff resides. So all of the SSL, TLS, certificate handling, all of that's in Schannel. I've programmed it many times, for example, all of that certificate stuff, the certificate fingerprinting stuff that I did on GRC's server, that's all Schannel code and so forth.

And the point is, it's in both the client and the server. It's in Windows 7 and 8 and, okay, there is no 9, but 10. And also in, you know, XP, in old XP machines, in Server 2003, 2008, 2012 and so forth. So it's everywhere. It is an intrinsic component of Windows. So, "This security update resolves a privately reported vulnerability in the Microsoft Secure Channel (Schannel) security package in Windows. The vulnerability could allow remote code execution, if an attacker sends specially crafted packets to a Windows server."

So first of all, it's like, whoa, what? I mean, all Windows servers on the 'Net, by definition, have ports exposed. That's how you get to them. So there's no hiding for a Windows server. It's out there. That's what it has to be. Then, you know, and we've talked about vulnerabilities, Microsoft's whole vulnerability architecture through the years. So they had the Exploitability Index. And sometimes these things are like, well, yeah, if you stood on - if somehow you could stand on your right foot, raised upon your big toe, during a full moon, and spin slowly counterclockwise while chanting, then - oh, and threw a ping pong ball through a small hoop 10 yards away, you'd have a chance of exploiting it. Not here. This is the worst exploitability they offer, Index 1, which they call "more likely to be exploited."

Then we've also talked about how they always have sort of these cheesy mitigating factors where, oh, a bad guy would need to lure you to a website, and you'd have to have scripting enabled and click on something and then agree to down - blah blah blah blah. Okay. Mitigating factors for this, quote, "Microsoft has not identified any mitigating factors for this vulnerability." In other words, there's nothing you can do.

Leo: That sounds good. It's bad. There are no mitigating factors. Oh, that's good. No, that's bad.

Steve: Yeah. You know, like, for example, we'll often talk about how, oh, go into the registry, and you can turn, you know, add this setting to disable this obscure thing that you don't actually need and nobody actually uses, but it's on all the time anyway, so you can turn it off and then forget about it, and you're fine. No. There's nothing...

Leo: Oh, that's not good.

Steve: ...like nothing that we know of that you can do to mitigate this. And then they also have workarounds where, like, well, you can turn this off, and then turn - blah blah blah, et cetera. Against, workarounds: "Microsoft has not identified any workarounds for this vulnerability." So what this says is that, I mean, and what I'm a little surprised about is, since I saw this, this morning, I'm seeing Twitter begin to catch up. People are looking through this, as I did, and going, whoa, hold on, what? Because unfortunately this doesn't have a fancy name. This isn't Shellshock or Humpty-Dumpty or anything. No one gave it a cutesy name. But this is like, to understand this, from what this looks like, this is as bad as it gets for a Microsoft, like for a major Internet server vulnerability just sitting out here.

Leo: I'm kind of surprised that they're not making a bigger deal out of this.

Steve: Well, they don't want to. Oh, and the other thing that I loved was that they do make a point of saying, oh, and we've added four shiny new TLS cipher suites. So it's like, oh. Well, thanks. But you've also made potentially every Windows server on the Internet is, like, going to be vulnerable as soon as the bad guys figure out what it is, what magic packet you can send to every Windows server on the 'Net today to take it over remotely. So they have an FAQ, and they say...

Leo: That's terrible. Okay, so, okay. So I understand that what they're thinking is, well, look, this isn't in the wild because this was privately found.

Steve: Correct, privately reported, yes.

Leo: What we would like to do is get everybody patched as quietly as possible.

Steve: Yes.

Leo: And then we'll mention it.

Steve: Just slip it in and get it in. Maybe nobody will notice. Maybe they'll say, oh, look, we have four new TLS cipher suites. Isn't that nice. Oh, and it's fixing some little problem over here.

Leo: Pay no attention to the little problem over there. Cipher suites.

Steve: So under their FAQ they ask themselves the question, because they really have no choice: What might an attacker use the vulnerability to do? Answer: An attacker who successfully exploited this vulnerability could run arbitrary code on a target server. How could an attacker exploit this vulnerability? An attacker could attempt to exploit this

vulnerability by sending specially crafted packets to a Windows server. Oh, but would anyone ever think to do that?

Leo: Why would anyone? Why?

Steve: What systems are primarily at risk from the vulnerability? Then they said: Server and workstation systems that are running an affected version of Schannel are primarily at risk. Okay. Wait a minute. There are no server and workstation systems that are not running an affected version of Schannel at this time.

Leo: So, okay. Let me just capsulize this. There is a vulnerability that affects all versions of Windows Server. It's IIS; right?

Steve: All versions of Windows.

Leo: Of Windows.

Steve: Yes. Both workstations and servers.

Leo: So every version out there of Windows.

Steve: Yes.

Leo: The vulnerability can be triggered by a remote attack, sending misconfigured packets.

Steve: Correct.

Leo: Of course that means that the system has to be online. But all servers obviously are.

Steve: Right.

Leo: Could it be sent over port 80?

Steve: No. It's Schannel. It is in the secure suite. So it's got to be - it's some sort of a TLS or SSL deliberate malformed communication.

Leo: Could an IIS server that's not running TLS, doesn't have an SSL cert, still be

vulnerable?

Steve: Probably not. My guess is that what they have probably - somebody found a buffer overrun in Microsoft's HTTPS, in their SSL/TLS protocol. It's that bad.

Leo: So if I - just so everybody understands, if I'm running Windows behind a NAT server, if I'm not running Windows bare to the Internet, and I'm not running SSL or TLS of some kind, I'm not vulnerable.

Steve: No. Here's the problem.

Leo: Oh.

Steve: Where they said "What systems are primarily at risk," and they said "Server and workstation systems that are running an affected version of Schannel." What may be possible is that any end user who connects to an HTTPS server, in doing that, if that server knew about this and was malicious, it might still bite you.

Leo: So a server that is running IIS...

Steve: No, no, no, a server running - any secure server.

Leo: No, but you can compromise it this way easily, that secure server.

Steve: Yes.

Leo: And then it could go on to compromise anybody connecting to it via SSL.

Steve: Or even worse. Once this gets out in the wild, once the hackers know what this is, any web server that end users connect to over a secure channel could reverse attack the end user.

Leo: But it would have to be compromised.

Steve: Well, no, or malicious, yeah.

Leo: But let's say I'm a company, I'm a company, big company, running Outlook web server so people can use their Outlook mail on the web.

Steve: Yeah, okay, good.

Leo: My server, my server gets compromised because I'm running IIS, and I am out in the public. That compromise could then put malicious code on there that would infect every single person who logged into the Outlook webmail.

Steve: Correct.

Leo: One after the other, boom boom boom.

Steve: Yes. Or you could also be Jimmy's Evil Blog.

Leo: Right.

Steve: That, like, or Jimmy's Secure and Evil Blog, offering TLS connections for people who want to read the blog, and it's actually infecting the workstation of everyone who visits.

Leo: So the first thing as a hacker I would do is go after VPNs. I'd go after...

Steve: Yes. Well, yes. Like servers are clearly the pot of gold because they've got...

Leo: But they have to be running IIS or Windows. They have to be running on a Windows machine.

Steve: Correct.

Leo: An Apache-based server, which is still a majority, fortunately. These are not vulnerable.

Steve: Yes. And Nginx, they're all going to be safe, too. So you're fine.

Leo: Yeah, we're Nginx, absolutely, and Apache, yeah.

Steve: Yup, yup. So it's only the...

Leo: But if you're running - but you're not, by the way, because you're running Windows Server.

Steve: Right. And so is eBay, and so is Amazon. Or, no, no, no, not Amazon. eBay is. I think PayPal is. I mean, there are major sites which are on IIS platforms. I don't know what the percentage is now, but I'm thinking, what, maybe...

Leo: Ten, 20, 30?

Steve: ...20 or 30?

Leo: Yeah.

Steve: Yeah.

Leo: It's not the majority, but it's a significant portion.

Steve: Yeah, and everyone knows who they are because the server identifies itself in protocol...

Leo: Right. If you go anywhere that says ASP or, you know...

Steve: Yup.

Leo: ...active server pages, that's it. So they can be compromised. They of course can be compromised to this particular attack, and then they'll pass it along to all other Windows machines. But once you compromise a server, you might as well just put the whole kit and caboodle on there.

Steve: Well, yes. And what, I mean, depending upon the site that someone manages to penetrate, I mean, we haven't had an exposed server protocol vulnerability, well, except, you know, I mean, well, okay. Heartbleed was that. But that was like, as we know, a low statistical likelihood of compromise.

Leo: And there was a mitigation. You could just turn off SSL. I mean SSH.

Steve: Correct.

Leo: And then you're fine.

Steve: Right.

Leo: There's no mitigation for this, you're saying. You can't switch off a service.

Steve: Right. This is probably in the underlying Schannel library core. It's when your secure server accepts a connection over port 443, SSL/TLS, there's something that a bad guy can do to, like, send a huge mother lode packet in containing their own code, and that server will execute their code. I mean, it's like the worst thing that can happen. And it doesn't sound like...

Leo: Are we not doing a disservice by publicizing this? I mean, shouldn't we just all keep this quiet until Microsoft can patch everything?

Steve: Well, the good news is, I mean, I have to tell our listeners that anybody, certainly we have, we know we have listeners who are running...

Leo: If you read the document, you're going to know.

Steve: Yeah, we know - yes, exactly. Anyone who reads this is going to have their eyes bug out when they come to the third thing on the list. And all, we certainly have listeners who are running, who are in corporations running IIS servers.

Leo: And they should know.

Steve: So they need to know right now. You know, the moment I disconnect I'm making a little road trip over to my datacenter because I haven't been over there for a long time, and it's time to do a little maintenance. But, yeah, I'm not leaving this thing hanging for a minute.

Leo: This is 2992611.

Steve: Yes. The MS14-066.

Leo: And it's as bad as it gets.

Steve: It is as bad as it gets for a remotely exploitable, hanging out there, flapping in the breeze, open port, pot of gold. Because what hackers want is servers. They want servers. And, but end users need to pay attention, too, because it may very well be that, by connecting to a malicious server, you could expose your workstation, and that's not good, either.

Leo: What is it, on a server, is there a Windows Update just like there is on the desktop client?

Steve: Yeah, exactly the same procedure. Well, I mean, my annoyance with Microsoft is these are really all the same thing. You know, Server 2003 is Windows 7. Windows 2008 is Vista. 2012 is - I got that backwards. But anyway, yes. I mean, the server platform exactly corresponds to a workstation platform. They just configure it differently. They just, you know, same foundation, same code, same everything. So there it is. Vulnerability in Schannel could allow remote code executions.

Leo: We have not seen any exploits in the wild as of yet.

Steve: Right. This just, you know, when they publish the code you know - it's not like this is not going to get attention. The bad guys are going to be tearing this thing apart. We can hope that Microsoft did what they could to obfuscate the change they made, like rearranging the code so it's much more difficult to figure out what changed. But they always do that. And we've seen now the pattern. We know that patches are reverse-engineered to find out what got changed. So certainly no one should rely on the idea that this is not in the wild yet. It's just a matter of days now. And I imagine we're going to - we'll be talking about this for the next few weeks, that something's going to happen here.

Leo: They don't mention XP in this list because they're no longer supporting it.

Steve: Right. But I'm sure it's vulnerable.

Leo: Well, if Server 2003 is vulnerable, I think that means that XP is vulnerable.

Steve: Goes all the way back, yeah.

Leo: So this might...

Steve: I think XP; I think 2003. I think it was Windows 2000 and XP. I don't remember how they were paired. Because there was a Server 2000. I think that was with XP.

Leo: Okay.

Steve: Because, see, the fact that Server 2003 is supported...

Leo: Well, it's still updated. It's not XP. Even if it's...

Steve: I think that corresponds...

Leo: Even if the core of it was XP, it's still supported directly.

Steve: Right.

Leo: Nevertheless, I imagine...

Steve: I think that 2003 and Vista were paired.

Leo: Maybe that's it, okay.

Steve: I think those were the two. And '08 and 7, and then 2012, Server 2012 and Windows 8. So anyway, so amid this gloom it is significant that we got four, I mean, the silver lining is we did get four shiny new TLS cipher suites, and they are nice. Two of them use RSA for key agreement. Two of them use Diffie-Hellman. So we know that that gives us - it's DHE, Ephemeral Diffie-Hellman. We've talked about that. That gives us perfect forward secrecy. So we have two new, of the four, two give us perfect forward secrecy, which is now what everyone wants. And they are GCM. They're Galois Counter Mode ciphers.

Remember GCM, actually that's the same cryptographic mode that I ended up choosing for SQRL. It's a hybrid authenticated encryption which does both encryption and authentication at the same time, rather than needing to separately encrypt and authenticate. And as we talked about recently, SSL got it backwards, where unfortunately the authentication happens first, and then the encryption occurs, which means that - and actually it should have been the other way around because that means you decrypt, then you authenticate, and that exposes SSL to all kinds of vulnerabilities.

So anyway, this is four new cipher suites which I'll also be bringing up this afternoon because they're just beautiful. And very secure, SHA-256 and 384 for the hashes. So I'm glad to have them. So this is a nice update. Unfortunately, it's fixing a real problem that is going to force reboots of all these Windows machines after - oh, and it does require a reboot, by the way. You're not getting away without because this is core code in the kernel.

Leo: Yeah, wow.

Steve: And in addition to that...

Leo: So XP is - Server 2003 is XP. 2000 is 2000 Pro. So just to get that...

Steve: Oh, good. So they did, they are going back and fixing 2003?

Leo: 2003 they are fixing. But they have been fixing.

Steve: Right. Oh, and they'll fix XP as long as you say that you use that embedded thing, hack. So...

Leo: Right. They're still doing that?

Steve: Yeah. So XP, I'm still getting updates on XP because I said, oh, yeah, I'm embedded.

Leo: And you will want this update. Not that most people run server on a plain XP machine. But maybe they do.

Steve: No, but remember, clients are vulnerable, too.

Leo: Clients are vulnerable, too. So if you surf to a compromised machine, then you will be compromised.

Steve: Right, exactly, yeah. You have to go to a compromised machine, but nobody wants to do that.

Leo: Right.

Steve: In addition, we had, I mean, this is like almost, it's almost an anticlimax, 17 vulnerabilities fixed in IE. XML Code Services were patched. Microsoft Office had three privately reported vulnerabilities fixed. There's also a problem which we don't often see in TCP/IP. If a bad guy got, somehow got code running on your machine, this is not something that you have to worry about a lot, but they said in TCP/IP IOCTL processing: "This security update resolves a publically reported" - oh, publicly reported, so that raises the bar, that means that everybody knows about it - "vulnerability in TCP/IP that occurs during the input/output control." That's the OCTL processing. "This vulnerability could allow elevation of privilege if an attacker logs onto a system and runs a specially crafted application."

And again, it's like, okay, how is that going to happen? That's not something you need to worry about. "An attacker who successfully exploited this vulnerability could run arbitrary code in the context of another process." Well, that's not good. "If this process runs with administrator privileges, an attacker could then install programs; view, change, or delete data; or create new accounts with full user rights." So this is one of those things where it's difficult to see how they would gain that foothold. But if they did, then all hell breaks loose.

But still, the requirements for this are such that we're not running around with our pants on fire the way we are for this server, this first vulnerability we talked about, MS14-066. But Windows Audio Service also is being fixed. .NET framework, SharePoint Foundation, Remote Desktop Protocol, IIS has a restriction feature bypass, Active Directory Federation Services, even the Japanese IME, the Input Method Editor had a vulnerability fixed. And not to be forgotten is the Windows kernel mode driver has some problem where, if a specially crafted TrueType font file was put out on a Windows share, and a bad guy got you to enumerate it, he could take you over.

So, I mean, this thing is just - it's got something for everybody. But one really, really

important update that everyone needs to handle immediately, like when you stop listening to this.

Leo: Nice. Okay.

Steve: So yesterday Obama, President Obama, to address him formally, I guess, announced that he's going to ask the FCC to reclassify ISPs as telecommunications carriers under something known as Title II. And this of course has rekindled the whole debate. Your producer, Jason, asked me for Brett Glass's email address yesterday, since Brett, as we know, we had Brett on the podcast to talk about this [SN-457], sort of has the ISP's viewpoint of...

Leo: Yes, an ISP's viewpoint. If we got Dane Jasper of Sonic, that might be another viewpoint.

Steve: Yeah.

Leo: But I know Brett feels very strongly. So we thought we'd get him on TWiT.

Steve: Good.

Leo: And we're trying to get Nilay Patel to debate him, which would be fiery.

Steve: And this would be Sunday? Is this going to be Sunday next week?

Leo: Yes. So Brett has confirmed. We will have Brett on. And maybe we'll get Dane Jasper, if we can't get Nilay. But I want to get both sides of this because I think there are two sides. The Title II regulation is a fairly hardcore regulation. It regulates them as a public utility.

Steve: Right.

Leo: And a lot of ISPs do not think that's a good idea.

Steve: Right.

Leo: A lot of the debate is, well, who do you want to regulate the Internet? Do you want the U.S. government to regulate the Internet? That seems like a bad idea. But certainly these guys aren't going to regulate themselves. So this is a tough, tough challenge.

Steve: It is. I mean, I'm, as an independent serial entrepreneur who's created and operated small companies for my entire life, I want as much freedom as I can have. But I do recognize that the free enterprise capitalistic system has a flaw. There is a tendency, and we see it happen, for large companies to use their inherent power to become larger still. That is, and what that does is it creates an unstable system. From an engineering standpoint, negative feedback is something that creates homeostasis, where as something begins to drift off, if you have feedback that corrects it, then that's good. That tends to create a stable system.

The problem is that capitalism is inherently, in a free market, is inherently unstable. It doesn't tend to self-correct. Large companies tend to get larger because of network effects, as it's known in economics. And so much as I dislike the idea of "government," in quotes, regulating, I recognize the need for some control. And I'm unhappy with the idea that there is actually no choice, I have no choice in my source of bandwidth. And it turns out most Americans don't. Most of us are, first of all, unhappy with our bandwidth provider, and have absolutely no useful market choice.

Leo: And of course I'm sure Brett will point this out, the reason you have no choice is because the government intervened in the first place...

Steve: Yeah.

Leo: ...to create monopolies for cable and DSL.

Steve: Yeah.

Leo: So it's a mess. And I think...

Steve: It is.

Leo: I don't know what the answer is. You know, President Obama campaigned on the promise of supporting Net Neutrality. He has been kind of weak, particularly in appointing Tom Wheeler, who many feel is just too compromised.

Steve: A lobbyist for the cable industry.

Leo: Yeah. And he's the chairman...

Steve: Or an ex-lobbyist.

Leo: ...of the FCC. However, I have to say Mr. Wheeler has been coming up with some pretty good ideas and creative ideas for trying to solve this conundrum. He has not just said, oh, trust the cable companies, they'll be find. So he's trying to do - it

seems like he's trying to do his best. But we were waiting for something from the president, and this is exactly the, I think, most Internet advocates, this is exactly what they wanted to hear. I have misgivings, though, and I think any reasonable person does.

Steve: Remember, too, that, as I understand it - and, I mean, I'm not obsessing about this. I haven't been following it super closely. But as I understand it, the FCC did impose some regulations. And Verizon sued.

Leo: That's right.

Steve: And the Supreme Court agreed that the FCC did not have the authority to regulate ISPs in that fashion. So basically this is a response to that. By reclassifying ISPs as common carriers, now there's no question that they fall under the FCC's regulation.

Leo: Yeah. This was the roadmap the court gave the FCC. They said, look, you don't have the - either you get Congress to mandate this - because Congress is ultimately the body that tells the FCC what it can and cannot do. Either you get Congress to give you these powers, or you're going to have to go some other route like Title II.

Steve: Right.

Leo: And we'll see. Title II, as Brett has pointed out, has regulations that would be considered onerous for any Internet service provider because...

Steve: Well, and it's old.

Leo: It's aimed at Ma Bell. It's aimed at...

Steve: Right. It's old, and it's antiquated, and a lot of it doesn't make sense. But it's the only thing we've got at the moment.

Leo: It'll be very interesting. And we're going to have a great debate on Sunday.

Steve: Wonderful.

Leo: I'm looking forward to it, yeah.

Steve: Okay. So I quietly departed for Las Vegas on Thursday. I never, just as sort of general security protocol, I never preannounce when I'm going to be away because that just seems, you know, unwise. But I always talk about trips when I come back. And I

was asked in the summer, I think in August, by DigiCert, my certificate provider, with whom everyone knows I am super happy. And of course I've worked, during the whole CRLSet thing - they participate in the certificate authority organizing committee and groups. And they had asked if I wanted to do a blog posting or a submission there. And I said, no, I pretty much said my piece.

But anyway, when they were holding a security summit, their 2014 security summit with all of their major customers and other interested parties, they said, "Hey, Steve, would you like to speak?" And I said, "Well, the only thing I really have to talk about is the project that I've been working on for, then, about nine months. And that's SQRL." And they said, "We'd love to have you talk about SQRL." And I said, "Well, then, I'll happily show up."

So on Friday I gave a presentation to the entire conference, basically a 45-minute front-to-back talk, a presentation about SQRL, how it works, what it does, the whole thing. And I don't know - there was a camera in the back recording it. And, I mean, I just - I've been so busy since I got back I haven't had a chance, but I will shoot a note off saying, hey, you know, is that presentation available? Because I did a very nice, 45-minute, I left nothing out, covered everything. It's the nicest formal presentation I've put together so far. So maybe I can get a copy of it.

But afterwards, in the first break after my presentation, there were a number of people who were interested came up, and we were chatting. And one of them was Brad Hill, who I had never met. But I certainly knew his name because his name is all over all of the FIDO docs. FIDO is the big sort of historically moving slowly, sort of glacially forward authentication alternative.

Leo: This is the one that Yubico key supports is FIDO; right?

Steve: Well, they support...

Leo: FIDO 2.

Steve: Well, yeah. There's two specs. There's UAF, which is the full one, the Universal Authentication Framework. And then what Stina sort of - she sort of peeled off the part that she wanted, frankly.

Leo: Ah, okay.

Steve: And that's called U2F, Universal 2 Factor. And that's what she did with Google and what we talked about a few weeks ago was that announcement. And so it's sort of FIDO Junior. And it operates very differently. But it borrows some of the technology. Anyway, what really delighted me was that Brad Hill, who is a very nice guy, who was at PayPal during all of this - PayPal made him available to the FIDO project. So he was like one of the lead architects of FIDO. He's now very recently switched over to Facebook. So he's now a security engineer, his business card says, at Facebook. He said that what he just saw was the most well-thought-out authentication system he has ever seen, in what I presented.

Leo: SQRL.

Steve: In SQRL.

Leo: Awesome.

Steve: And it was funny because there's one, you know, the key concept behind SQRL, the thing that hit me that morning, just about a year ago, I think it was in November of 2013 [SN-424], and this just, you know, I'd been reading through elliptic curve crypto and looking at Dan Bernstein's stuff. And there's a place on Dan's page where he's talking about his ED 25519, the 25519 elliptic curve, where in sort of an old Q&A he says, you know, how do you create a private key? And then his answer is you don't.

And that's what's so special about the crypto that I chose is that anything can be the private key. And that's the fundamental basis for SQRL because that "anything" is when you take the user's master key and use it to key a hash of the domain name. Then the result of that is the private key. And that's what's so cool because, for example, what FIDO does, and this is unfortunately what Stina's solution is, is they choose keys at random. And but so now you've got a random private key and the matching public key and nowhere to store it. So what they do is they encrypt it and give it to the server to store. So you're authenticating to the server, but the first thing you have to do is ask it back for the private key it's been holding for you. And then you decrypt it and use that to sign the nonce that it gives you to prove that you have the private key that it just gave you. So it's like, okay.

Anyway, so the point of this is that I sort of gloss over that. Obviously in a short presentation I don't have time to dip into everything. So I have a diagram that I showed that is like, it's the diagram on the first page of the SQRL pages at GRC, where I show that process, where you take the master key and hash it with the domain name. That's the private key. Then you run it through an API call to produce the public key. And then you also use the private key to sign the URL. And the public key is your identity token, and the signature is your proof that you have the private key. And I just sort of - and then I move on.

And so Brad was looking at that and, you know, really knows crypto, but wasn't aware of this particular property of the Bernstein 25519 elliptic curve. So he came up to me, when we were talking about this, and he said, you know, he said, remember that famous - oh, actually he told me, he says, you know, you really need to change your presentation because you just sort of skipped over that step. And he said, remember that famous joke where Einstein or someone is on a chalkboard, and they're working out a proof of a theorem, and they get stuck, they paint themselves into a logical corner, and so in the series of equations there's like a little puff of smoke, and it says, "And then a miracle happens."

And so his point was that he recognized, because he understands this stuff, that apparently something was missing. But it's this particular choice of crypto that solves that problem, and that's the magic of SQRL. And of course now he understands that. And so then he was having to - he was digesting that, realizing that I'd just obsoleted the way FIDO worked because it was a kluge, and SQRL wasn't.

Leo: Ooh.

Steve: So anyway, it was a great...

Leo: How did he react to that?

Steve: Well, I didn't rub his face in it. But, you know...

Leo: Did he kind of understand that?

Steve: Oh, yeah, he definitely understood it. He said to me, I mean, he said, "That means there's no need to have the server store the private key on behalf of the user anymore." I said, "Right."

Leo: Right. And he said, oh.

Steve: Yeah, we don't - I don't have to - we don't do that with SQRL.

Leo: Oh.

Steve: It's much better, actually.

Leo: Interesting. Wow.

Steve: So, yeah. Anyway, but he's a super nice guy. And I think he was glad for the meeting, and I certainly was because...

Leo: Good, good, good, yeah.

Steve: ...somebody who really knows this stuff was able to watch the presentation and then say, "This is the most well-designed authentication system I've ever seen." And this is the guy who just spent five years doing that. So, wow.

Leo: Well, I hope people pay attention to that.

Steve: Ah, well, we'll - it's going to succeed if it should. I hope it does. So I have one last security thing. And this is important. For everyone who has a Belkin N750 dual band router, I mention this, it's this specific router, but it's been around for three years. It went on the market in 2011. So it's Belkin's N750 dual band router. There is a very easily

exploited vulnerability that allows an attacker to gain root on the router through exploiting the guest network, which there's a problem with. The guest network has a web page, has a little web server where you authenticate, and a specially crafted POST - verb, you know, HTTP has GET and POST for submitting only a URL query or that plus a block of data. And POST is typically used, for example, for submitting the contents of forms. If you maliciously format the POST submission, you can access the telnet server and get access to the telnet, basically the command prompt with full root access on the router.

Leo: Oh, crap.

Steve: So they have a firmware update. The firmware that everyone currently has ends in 1.10.16m as in Michael. What you want is 1.10.17, sorry, 1.10.17m as in Michael. So just a heads-up. I'm sure within the sound of my voice, with our whatever we have, about a hundred thousand listeners, there are people with Belkin N750 dual band routers. As a workaround, you can immediately shut down the guest mode, if you want, if you're concerned. But this allows wardriving to really be...

Leo: Wardriving, yeah.

Steve: ...wardriving. It really means war.

Leo: Wow.

Steve: Yeah. So, and in the show notes I have a link to the guys that found it and a very simple proof of concept. I mean, it's just - it's trivial to implement. Oh, and I forgot to mention, it's been put into Metasploit. So, and it's an automated remote takeover module in Metasploit, so also very available. So anybody with this Belkin N750, update your firmware. Get .17m from Belkin.

Leo: You know, and we don't say this enough because so many of these router manufacturers never update their firmware.

Steve: Yeah, they're just not good about it.

Leo: Well, it's such a commodity device, they...

Steve: Yes, you're right. It's - you're right.

Leo: So thank you, Belkin, for fixing it.

Steve: Yeah, you're right. I'm glad you said that, Leo. That's a very good point. It's easy for us, and, I mean, we just sort of take it for granted. But you're right. Consumers - there's probably not a lot of demand because consumers typically see it as like a set-it-

and-forget box.

Leo: It's a box, yeah.

Steve: Yeah.

Leo: So, yeah. I mean, and there's not a lot of incentive. I'm sure there's no money in fixing it.

Steve: No.

Leo: But they really should fix it. So, good.

Steve: I was exchanging tweets with somebody last week, and I don't remember now what the topic was, but it was related to security. And he said some - I guess in the tweet, in his tweet he said that, oh, by the say, SpinRite just came to the rescue for me. And he sent me a link to his blog post from last month, so a couple weeks ago, October 2014. And the topic was, his subject was "Universal fix for Windows KSOD." And I thought, what? We know about the BSOD, the Blue Screen of Death. I thought, what's a KSOD?

So he said: "Ever had your Windows installation inexplicably die, leaving your computer unusable without a fix? I have - more times than I'd like to count. The last time this happened was yesterday when Windows 7 would only boot into a black screen with a movable cursor, also known as the blacK [with a capital K] Screen of Death." So, you know, "B" was already taken for Blue Screen of Death, so he used the "K" at the end of black, so KSOD for blacK Screen of Death. He said: "It was a serious case, considering none of the safe modes or repair function in the Windows boot options would work. Each option would universally end in either a KSOD [as he calls it] or the classic BSOD after hanging on [and then he has] aswRvrt.sys during safe boot. After exhaustively eliminating all possible 'regular' fixes that were available on the Internet, I decided it was time for the big guns: Steve Gibson's SpinRite."

He said: "Prior to trying SpinRite, I first tried Kaspersky's Rescue Disc 10, which was entirely useless for my case. After booting from the rescue USB dongle, I would always get a 'Missing Operating System' error in the boot screen." So it sounds like Kaspersky was needing a little more there than was present. And of course SpinRite needs nothing. It operates, it brings its own little OS along with it. So anyway, he said: "Not reassuring. I have long been a fan of Steve Gibson's Security Now! podcast, which is why I knew of the tool. I knew the tool would be one of the few things that might do the trick, so I gave it a shot.

"After about an hour running SpinRite 6, and a few reboots later, my Windows 7 installation was working perfectly, as if nothing had ever happened. SpinRite saved the day." And then there's a bunch of stuff that I'm skipping over because then he said TL;DR at the end: "SpinRite saved my machine from a perpetual and otherwise unbeatable KSOD scenario. And my guess is that, if you are having KSOD problems, then SpinRite is one of the few things that might help you, too." So I don't have his name, unfortunately, in front of me, but thank you. I'm sure he's listening to this right now. So

I appreciated our Twitter exchange, and thanks for sharing the link.

Leo: Yup.

Steve: And there it is.

Leo: Universal Fix.

Steve: It's a very, very cool screenshot...

Leo: And there's the DynaStat system.

Steve: ...showing the DynaStat.

Leo: Yeah. Nontrivial to get a DynaStat screenshot because you're running DOS.

Steve: Exactly.

Leo: I don't know how you'd get that. That's funny. That's awesome. Very nice. So he didn't even send this to you. This was just - you saw the blog post.

Steve: Yeah, well, he sent me the link.

Leo: Oh, he did.

Steve: And so that, yeah, so I was able to find it that way.

Leo: Great. And it's Reposter.net. Let us continue on with Mr. Gibson and a little bit of certificate transparency.

Steve: Okay. So everyone knows who's been following the podcast for any period of time how the current certificate authority-anchored system operates. The idea is that certificate authorities verify the identity to differing degrees of entities, individuals, companies, organizations of any kind, who want to have a - want to assert their identity on the Internet. So the idea is that they prove who they are to the certificate authority. They provide their server's public key to the CA, the certificate authority. The certificate authority signs that public key and gives it back to them.

Then, whenever someone connects to them, they send the client, the user's web browser typically, this signed public key. The public key itself is the key to them establishing a secure connection with each other because their private key, the matching private key,

never leaves their control. And them having the private key that matches the public key that they've just provided is the way a secure tunnel is bootstrapped such that nobody listening in the middle can intercept. And the fact that it's signed by the certificate, this public key, which is what's going to be used for communication, the fact that it's signed means that somebody else, a third party, whom the client trusts to have performed due diligence, is asserting, yes, this is really Amazon or Facebook or eBay or PayPal or GRC or whomever you're establishing a secure connection with. So that's the model.

Now, we've talked about the many ways this fails. Good as it is, there are problems with it. For example, we're trusting, I mean, hundreds, many, like four or 500 different authorities with this performance of due diligence. And the problem is, whereas all of GRC's certificates are only signed by my chosen certificate authority, DigiCert, it is entirely possible for, and I'll just pick on them because they're my typical whipping boy, it's entirely possible for the Hong Kong Post Office to sign a fraudulent certificate asserting that it is GRC. And if the Hong Kong Post Office, which is a CA, a certificate authority, is trusted by the user's browser, then a different website can impersonate GRC, if they're also able to get DNS to cooperate because somebody's web browser has to believe they're going to GRC, but be given the wrong IP address or be intercepted and rerouted to a fraudulent server that's offering a certificate signed by an untrustworthy certificate authority.

So that's one of the problems. Then of course there's - so that's like the malicious CA problem. But then there's the compromised certificate authority. And remember we talked about years ago the famous DigiNotar problem. DigiNotar was a - were they - I think they were a Dutch certificate authority that had a breach, had hundreds of certificates created somehow in their system. Somebody got in and was able to get all kinds of certs signed. They knew about it and didn't tell anybody, hoping that they could get away with it.

Well, the fact that they found out about it and kept quiet ended them. They were bankrupt a few months later because it's one thing, you know, anyone can make a mistake. But if you're a CA, you have got to immediately acknowledge when you find out there's a problem, and take responsibility for it. And the fact that they didn't meant that no one could ever and would ever trust them again. All the browsers immediately suspended their support. And if you don't have browsers trusting you, remember, this whole thing is that the browser verifies the certificate by trusting the signer of it. And if the browsers retract their trust of the certificate authority, that certificate authority is out of business. No one, I mean, they can't assert their reputation, even if they haven't completely wrecked their reputation. So there's that problem.

And then so we have the malicious CA. We've got the compromised CA. And then just the mistake where, like, there was another instance more recently where a certificate authority had issued a certificate that had CA rights. That is, normally the intermediate certificates are themselves unable to be a certificate authority. But that's just a bit flag in the privileges, in the characteristics of the certificate. And one CA did issue an infinitely powerful certificate that was able to sign any certs of any kind. And so that mistake was corrected.

So the point is that there are a number of things, a number of ways that this existing hierarchy with certificates can break. And because of Google's size, they are almost always at least among the websites that are compromised. That is, they're discovering bogus certificates that they didn't issue that other people issued. And, I mean, one of the things that is nice about Chrome, the Google browser, is that they've built technology in where Chrome knows much more about the - because Chrome is from Google, Chrome knows about the validity of Google's own certificates and immediately sends up all kinds

of warnings if a certificate that appears to be valid for Google.com or any of Google's properties is used on the Chrome browser that's actually not from Google. So they're in a special place, having both making certificates and their own browser that can be aware of all the certificates that should be considered legitimate. But none of the rest of us have our own browsers that everybody's using, so that doesn't work for us.

Still, Google has spiders, bots, which are out rummaging around the Internet. That's what they do. They're basically going to all websites and servers, following links, sucking down and indexing the entire Internet. And so it occurred to them at some point, it's like, hey, you know, every time we connect to a server following a secure link, and that's happening historically more and more, we get the certificate. That's what happens. The first stage of establishing a secure connection is the secure server sends the client, that in this case is a Google bot, the certificate.

So they began to think, you know, why don't we start collecting these, all of these certs? Because, I mean, these are all the certs on the Internet, all the certificates. And when you think about it, that doesn't exist anywhere. There's no central repository of all the certificates. But imagine if there were. Imagine if every certificate issued existed in some kind of database that could be queried. I would love to query it. Hey, what are the GRC.com certificates? Let me have them. Because, boy, don't I want to know if there are any that I don't know about. And in fact everybody wants to know that.

Leo: It couldn't be used to validate because some might be bogus.

Steve: Ah, well. And so, yes. So the first phase of this was this notion of collecting them all. And, well, okay. So we used Google's bots to create this database. And then Google started saying, okay, what we want is - somehow they sort of made this conversion to this idea of, okay, we will put all the certificates in that we encounter. Now, we'd like to be able to allow people to query the database for properties that they care about and find out if, you know, what certs we have encountered that are there. We'd also like to encourage certificate authorities to submit certificates that they're in the process of issuing.

And so this is sort of - it's going to kind of have to bootstrap itself. But the idea would be that ultimately this database would run in parallel to the existing CA verification. First of all - and they call it a log, they call it the Certificate Transparency Log. It's not a log in the traditional sense, as I'll explain in a second. They called it a log because, like a log, a correct log, it has what they call an "append-only property." That is, by its design, all you can ever do is append to it. So it's a chronology of all the certificates that have been found so far. And the structure of it is such that it can only be appended to. And it has a series of security properties in its design that allow it - that don't require it to be trusted.

And this is where it starts to get a little mind-bending from a security standpoint. But the idea would be that, if it exists, and its design is correct - and I should say it does exist, and the design is correct, so these are sort of - I'm sort of trying to lay down a framework for understanding this. And it could be trusted - and again, that problem has been solved, and I'll get to that in a second. Then web browser clients could rely on it as an additional source of information, that is, people who owned domain properties could check the log to verify that only the properties, the domain properties that they own are present, that is, there are no fake versions of their certificates. And if they found them, then they would go through the normal process for remediating. They would notify all browser vendors, and tell the CA who issued that cert that this was bogus, and get them to revoke it, and tell the browsers to revoke it and so forth.

So the idea is that you, first of all, to get this thing bootstrapped, browsers will only trust certificates that they receive from servers if those certificates are members of the certificate transparency database. And, okay, so that's a requirement, that is, the certificate must be in the database in order for the browser to trust it. And then the database must be queryable by the owners of the certificates to allow bad certificates to be found because only if the certificate, only if a bad certificate is in the database will a browser trust it.

So what's useful about this is that, by creating a sort of a repository of all certificates - we don't have that now. Right now the browser receives a certificate from the web server and checks the chain of trust back, the validity of the chain of trust back to someone, some single certificate authority that signed it. But that does allow, within a controlled environment, like in Iran, just to pick on someone, for them to control a CA, and for them to issue certificates for Google.com, for example, which appear to be valid, only because the chain of trust is local. It's never being, you know, having a bright light shined on it. It's never being exposed to the light of day. It's the protocol doesn't have global scope. This system gives certificates global visibility which Google calls "certificate transparency."

So, okay. Now, how does this work? One way, the way to start visualizing this, the first way to think about it, which is not the way it's actually structured, but we have to start somewhere, is as a chain, a linear chain of hashes of certificates. So you start with the first certificate. And you take the SHA-256 hash. Now you take the second certificate. You append the hash of the first one to it, and you hash them. Now you have a second hash, which you append to the third certificate and hash them. And now you have a third hash, which you append to the fourth certificate and hash them. And you have the fifth hash, and so on.

So what you have is a chain of hashes of certificates which creates a dependency chain. That is, when you get done hashing all the certificates, you know, one by one in this chain, if you think about it, due to the way hashes work, the resulting single hash is a value which you only get if every certificate - you only get as a result of hashing all of the certificates sequentially. And now, moving forward from there, any new certificates that come along, it's easy to evolve this, that is, from a given starting point of having all that work done in the past. You hash any new certificates sequentially, adding them to the front of the chain, and the resulting final hash is the head of this chain.

And so essentially, thanks to this magic cryptographic processing of the way hashing works, you have a - essentially you're able to state the proof that all, every single previous certificate has been seen if you get this final value. And incrementally, what this means is that, that is, it's provable that you had this append-only property because any particular version of this log, and this log is this chain of hashes, is the full proper superset of any previous version, that is, because you can always look at the - any previous version is entirely representable just by the final hash that was its head. And you append to it by hashing additional certificates.

Now, if that was the actual model, this would not be very practical and very cumbersome. So it turns out that instead of a linear chain, we can use a tree. And if we use a tree structure, suddenly this thing becomes incredibly cool. There's a data structure known as a Merkle, M-E-R-K-L-E, a Merkle hash tree. And a Merkle hash tree is exactly what I described, but using a binary tree structure rather than a linear hash chain.

So, and I've never, I don't think I've really ever talked about trees. One of the things I have always planned to do on the podcast was do a fundamental data structures series of

podcasts to talk, you know, because we've done other fundamental things like CPUs and the Internet and other stuff. And I thought data structures, the fundamental underpinnings of data structures would be really great. But the world's just gone crazy with security problems, and we've never - we have very little chance to talk about that kind of stuff.

But a binary tree is if you took two things, like two hashes of certificates, and hashed them together, and then took two other hashes of certificates and hashed them together, then you took those two hashes that you'd done and hashed them together into a top one. So you can sort of see how there's a top node which splits into two, and it splits into two, and so on. So it turns out that - okay. And what I just described, that is a Merkle hash tree. All it is, is a series of joinings from the bottom up of two hashes into a third, or to make a third, and then somewhere else you take two hashes to make a third, and then you hash those together to make a fourth and so on.

So this binary tree has the very cool property that, and this is where it's difficult, I mean, I'm already sort of into where it's difficult to do this on an audio podcast without any audiovisual tools. But the idea is that where the size of the log grew as a linear function, if it was just that first linear example, if you do a - if you use a binary tree, it grows with the log of the number of certificates. That is, it grows incredibly more slowly. So that, for example, if you double the size or you double the number of certificates that are in the tree, the tree is able to grow only one - it only requires one additional level of growth. So it is a far more efficient means for storing these hashes.

And if anyone's curious, look up "Merkle hash trees" because essentially what this means is that every certificate that has ever been found - and I should explain, these logs exist now. There are about seven or eight of them that are being maintained. Google has about four. DigiCert, my certificate authority, was working with Google on this early on. They're running one. And there are a couple other people who are. And it is Google's intention, last I saw, to in Chrome, starting in just a few months, in February 2015, to require certificates to be present and proven in this certificate transparency log to have EV status. So this is another thing Google is doing to work on dramatically strengthening the certificate authority system.

And it's easy to get lost in the weeds here because it turns out that there are, when you have this tree structure, you can demonstrate that very few nodes need to be known in order to provide a proof that a given certificate is present in the tree. And that's the key. If we had this single linear list of chained hashes to prove that a single given certificate was in the tree, we would have to have the hash in front of it, that is, the hash that was there before it was added, and its hash and all the certificates since, to prove that the value we end up with is the value at the end of the chain.

But by breaking this down into a binary tree, the number of hashes that we need is just a handful because the binary tree structure is so efficient that it is very economical to provide a proof that any given certificate in the world has been represented in this tree. And that's what's so cool about this is the proof is very efficient and very lean. And that allows all the certs to be dumped in. And what will be happening in the future is that certificate authorities will be submitting, voluntarily submitting their certificates to this certificate transparency log, which is in this tree structure, in order to essentially preannounce their certificate as being in the world. And so we'll no longer have a situation where a certificate authority can go rogue and have its rogue-issued certificates blindly accepted by browsers and having us maybe discover that that exists. Once this system is in place and running and enforced, within a matter of hours, typically, any maliciously issued certificates will be found.

So that's the ambitious project that Google embarked on about a year ago, RFC 6962, if anyone's interested. And in the little abstract, first paragraph, it says: "This document," that is the certificate transparency document, "describes an experimental protocol for publicly logging the existence of TLS certificates as they are issued or observed, in a manner that allows anyone to audit certificate authority activity and notice the issuance of suspect certificates, as well as to audit the certificate logs themselves. The intent is that eventually clients would refuse to honor certificates that do not appear in a log, effectively forcing certificate authorities to add all issued certificates to the logs." So, ambitious, but it may happen. And a very, you know, a substantial and significant addition to the Internet's security infrastructure.

Leo: Interesting.

Steve: Yeah.

Leo: All right. Thanks for...

Steve: Certificate transparency.

Leo: Transparency. And by which we don't mean disappearing, we just mean transparency about the provenance.

Steve: Visibility, yes, visibility.

Leo: Yeah, visibility.

Steve: Yeah, exactly, making it all, making what's happening transparent to everyone.

Leo: Don't know what this means, but the chatroom just pointed out that Open Whisper Systems has just tweeted that RedPhone was removed from the Play store today. The tweet says we don't know why, but we've reached out to Google Support for more information. That was one of the security products created by Open Whisper Systems, along with the TextSecure text messenger that we had recommended.

Steve: Right, right. And I should mention that these Merkle hash trees, they're in LimeWire and Bitcoin and Gnutella. I mean, the Merkle hash tree itself is not an invention of Google. It's a very handy way for managing peer-to-peer filesharing, in order to verify that sets of files that have been received are valid by taking their hashes and then receiving a chunk of the tree and by evolving these trees over time. The Merkle hash tree itself is a really interesting data structure in fundamental computer science. It was invented back in 1975, I think. And it has been around and used for many different purposes ever since.

Leo: Interesting, yeah. Steve is at GRC.com. That's where you'll find SpinRite, the world's finest hard drive maintenance and recovery utility. We highly recommend it. While you're there, though, you might want to check out some of the freebies Steve offers at GRC.com, including this show. 16Kb audio is there of the show. That's the smallest audio file we offer. Plus the full text transcript written by Elaine Farris. Thank you, Elaine. And all sorts of other stuff, including information, more information about his SQRL protocol. If you want to get involved in that, there's forums. There's a lot going on.

If we do a Q&A next week, your questions will come to Steve in one of two ways. He does tweet replies, so @SGgrc is his handle. And you can also use his website. That's probably the best way.

Steve: Yeah.

Leo: GRC.com/feedback, and your feedback will be received by Steve that way. We have full quality audio and video of the show at our own site, TWiT.tv/sn. We also make sure that it's available everywhere you get your finer podcasts. If you want to watch live, you can do so at TWiT.tv every Tuesday, 1:00 p.m. Pacific, 4:00 p.m. Eastern time, 2100 UTC at TWiT.tv. Thank you, Steve.

Steve: Okay, my friend. I will be watching Brett Glass talk about Net Neutrality on this coming Sunday. And we'll orient for a Q&A next week.

Leo: And while we don't ever reveal Steve's itinerary, I do think he might be heading over to Level 3 at some point later today.

Steve: Yes, everybody. Update your Windows systems. I'll be doing that. You'll notice that GRC disappears from the 'Net briefly here in about an hour or so.

Leo: Briefly while it reboots. Thanks, Steve. We'll see you next time...

Steve: Bye.

Leo: ...on Security Now!. Bye-bye.