SECURITY NOW!

Transcript of Episode #792

"Slipstream" NAT Firewall Bypass

Description: This week we look at the dilemma of Let's Encrypt's coming root expiration, new Chrome and Apple zero-day vulnerabilities, some new high-profile ransomware victims, China's Tianfu Cup pwning competition, the retirement of a PC industry insider, the continuing Great Encryption Dilemma, police monitoring of consumers' video, more ongoing pain for WordPress, a note about a sci-fi book event one week from now, and Samy Kamkar's tricky Slipstream attack and its mitigations.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-792.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-792-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. There's lots to talk about. Let's Encrypt is facing a crisis on Android. Steve will explain what's going on. We're getting ready for Patch Tuesday, 113 new exploits. Steve will give us a little heads-up on that, but we'll have our analysis next week. And then we're going to talk about Slipstream. This is something everybody with a router will want to pay attention to, a NAT firewall bypass. We'll tell you how to fix it next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 792, recorded Tuesday, November 10th, 2020: NAT Firewall Bypass.

It's time for Security Now!, the show where we cover your privacy, your security, how computers work, all of this stuff you need to know, with this guy right here. He's our Explainer in Chief, Steve Gibson from GRC.com. Hi, Steve.

Steve Gibson: Yo, Leo. Great to be with you again for our 792nd episode.

Leo: Wow. Eventually those computers behind you will solve that problem.

Steve: They're working on it. You can see them. They're just chugging away.

Leo: Chugging away.

Steve: They're computing pi at the rate of one digit per year, I think. They're not very fast. So our title is something which I didn't have a chance to get to last week, but it was probably the most tweeted, like breathlessly: "Hey, Steve, you've got to talk about this, you've got to talk about this on the podcast." Well, anyway, so it's the topic. It's

Slipstream, which is our old friend Samy Kamkar, who's a hacker we've referred to from time to time. He's done a bunch of Black Hat and Defcon presentations. He came up with a way of bypassing the NAT firewall technology that we all take for granted and which is such a boon to those of us who want to be able to use the Internet, yet not have the Internet use us. So that's our topic, the Slipstream NAT Firewall Bypass.

But there was a bunch of stuff that has happened in the intervening week, since we were all last together. We're going to take a look at something interesting that's happening. And I would love to know what's going on behind the scenes because it sounds like the Let's Encrypt folks suddenly realized that the root certificate that they had cross-signed in order to get themselves bootstrapped is expiring before they were ready for it to. And like there's still a lot of - we'll talk about this - a lot of mostly Android OSes that never learned about the new replacement Let's Encrypt root cert. And it's also interesting, too, because in retrospect this puts a different little spin on what we talked about, actually it was the topic of last week's podcast, Chrome's decision to do their own root certs because they kind of have to. Anyway, we've got that.

We also have some Chrome and Apple zero-day vulnerabilities to discuss. I'm going to touch on some new high-profile ransomware victims because that's the threat that just keeps on giving. We have China's Tianfu Cup, which it's actually the third one of these which has happened. That's their own kind of equivalent of the Pwn2Own competition. That happened just this past weekend. We also have the retirement of a well-known PC insider, and his parting comments were sort of interesting.

We've got the continuing Great Encryption Dilemma which was brought back to fore by something that the EU just did. Also a somewhat troublesome experiment about police monitoring of consumer video, which we'll talk about. More ongoing pain for WordPress. Also I just wanted to drop a note about a sci-fi book event, which I'm sure John is aware of, occurring exactly one week from now, and I'm very aware of it. And then we will conclude by talking about Samy Kamkar's Slipstream attack and how fortunately it was not named after a vulnonym.

Leo: That gift keeps on giving. I follow them on Twitter, man. Every day there's something. Vulnonym. We'll talk about that later.

Steve: And of course we have one of our classic Pictures of the Week, which just, you know, you just have to say, what? Also, Leo, there's a padlock in the middle of it. It's small.

Leo: Oh, I didn't even see that. Oh, my god.

Steve: Because you wouldn't want that to be opened by mistake.

Leo: No, wouldn't want somebody to open that.

Steve: No, got to keep that sucker locked, yeah.

Leo: And we go back to Steve and our Picture of the Week.

Steve: For those who can't see the picture, we have another of those curious locked gates in the middle of an empty field. And it's like...

Leo: What?

Steve: There's got to be a story here. Like maybe there used to be a fence that they took down? Or, I mean, it doesn't look brand new, so it doesn't look like they haven't yet - like they put the gate in, but not the fence. And also there's a very well-used-looking padlock. The original image was very high-resolution so I cut it way down so that it wouldn't bloat the file too much. But, I mean, it looks like it's been there forever. And of course you'd want that padlock there because you wouldn't want anyone to open the gate.

Leo: Maybe there's a road under those leaves or something. But still it would be trivial to drive around.

Steve: Well, yeah, look, there's nothing on either side. I mean, it's...

Leo: People are still building those. Still building those gates protecting nothing.

Steve: Yeah. Probably somebody had a contract to, like, install the gate there. And they're like, gosh darn it, I'm going to get paid for this gate.

Leo: Whatever you say.

Steve: Whether they want it or not. Yeah, exactly. So something interesting has happened with Let's Encrypt. On Friday they blogged: "Standing On Our Own Two Feet." And the issue that's come up is so interesting and fundamental that I decided to share what they've said, with some inline editorializing, of course, about the situation they find themselves in.

So they wrote: "When a new Certificate Authority comes on the scene" - which I'll note does not happen very often - they said: "...it faces a conundrum. In order to be useful to people, it needs its root certificate to be trusted by a wide variety of operating systems and browsers. However, it can take years for the OSes and browsers to accept the new root certificate, and even longer for people to upgrade their devices to the newer versions that include that change. The common solution: A new CA will often ask an existing, trusted Certificate Authority for a cross-signature, to quickly get it into being trusted by lots of devices."

And, you know, we talked about this in the beginning of Let's Encrypt. And we looked at the certificate that they were producing was cross-signed by IdenTrust, I think it was. Anyway, so they said: "Five years ago, when Let's Encrypt launched, that's exactly what we did. We got a cross-signature from IdenTrust. Their DST Root X3 had been around for a long time, and all the major software platforms trusted it already: Windows, Firefox, macOS, Android, iOS, and a variety of Linux distributions. That cross-signature allowed us," they wrote, "to start issuing certificates right away, and have them be used by a lot of people. Without IdenTrust, Let's Encrypt may have never happened, and we are grateful to them for their partnership. Meanwhile, we issued our own root certificate, ISRG Root X1, and applied for it to be trusted by the major software platforms.

"Now, those software platforms have trusted our root certificate" - meaning the Let's Encrypt root certificate - "for years. And the DST Root X3 root certificate that we relied on to get us off the ground is going to expire on September 1st, 2021." They said: "Fortunately, we're ready to stand on our own and rely solely on our own root certificate. However," they wrote, "this does introduce some compatibility woes. Some software that has not been updated since 2016," they said, "approximately when our root was accepted to many root programs, still doesn't trust our root certificate, ISRG Root X1. Most notably, this includes versions of Android prior to 7.1.1. That means those older versions of Android will no longer trust certificates issued by Let's Encrypt."

They said: "Android has a longstanding and well-known issue with operating system updates." Yeah, a frequent topic of this podcast. "There are lots of Android devices in the world running out-of-date operating systems. The causes are complex and hard to fix. For each phone, the core Android operating system is commonly modified by both the manufacturer and a mobile carrier before an end user receives it. When there's an update to Android, both the manufacturer and the mobile carrier have to incorporate those changes into their customized version before sending it out. Often manufacturers decide that it's not worth the effort. The result is bad for the people who buy these devices. Many are stuck on operating systems that are years out of date.

"Google no longer provides version numbers on its Distribution Dashboard, but you can still get some data by downloading Android Studio. Here's what the numbers looked like as of September 2020." And I have in the show notes an interesting table, showing from the bottom which are the newest versions, moving back in time as we go up. And so, for example - and they're also identified by API level or Android version.

So right now, as of September 2020, this year, 8.2% of all Android devices, Google says, are running Android 10. If you go up a tier to include 9.0, which was Android Pi, now the total install base increases, that is, of Android 10 or 9, 9 or 10, to 39.5%. One more further up to add 8.1 to 9 and 10. That brings you to Oreo 8.1. Now we're at 53.5%. Move back again and look at everything since Oreo, the first, 8.0. Now we've got 60.8%. And if we drop one notch back to Nougat at 7.1 - so we've got 7.1, 8, 8.1, 9, and 10. And this is the time where Let's Encrypt's root appeared. We are at 66.2%, which is to say two-thirds of all current Android devices are from 7.1 on, that is, two-thirds are aware of the actual Let's Encrypt root.

But that leaves fully one-third of all current Android devices naive to this change in the root CA. And they're working right now across all Let's Encrypt sites only by virtue of the fact that Let's Encrypt has this cross-signing relationship with IdenTrust, which goes away on September 1st of next year, meaning that those devices will no longer, believe it or not, will no longer be able to connect to any Let's Encrypt sites. And we know how incredibly popular Let's Encrypt has become. So, oops. Bit of a problem there.

So they said: "Currently, 66.2% of Android devices are running version 7.1 or above. The remaining 33.8% of Android devices will eventually start getting certificate errors" - oh, that's true. I thought they were going to say upgrade, but that's, you know, it's never going to happen - "will eventually start getting certificate errors when users visit sites that have a Let's Encrypt certificate. In our communications with large integrators, we have found that this represents" - okay, they're saying around 1 to 5% of traffic to their sites. Okay. So those would be people using Let's Encrypt certs are saying that 1 to 5% of people who visit them have these older versions of Android.

On the other hand, there's a lot of people who have, like, one-third of all Android devices, that's a lot of devices, who wouldn't be able to bring up any Let's Encrypt-based websites. So they said: "Hopefully these numbers will be lower by the time DST Root X3" - that's the cross-signed root, the IdenTrust root - "expires next year, but they acknowledge the changes may not be very significant."

What can we do about this? They said: "Well, while we'd love to improve the Android update situation, there's not much we can do there. We also can't afford to buy the world a new phone. Can we get another cross-signature?" They said: "We've explored this option and it seems unlikely." And I'll just stop here for a moment to say, remember that not only is Let's Encrypt incredibly popular with sites that want encryption, and now you could argue these days need encryption, but the bad guys have been having a field day. So whereas when this original agreement was made with IdenTrust for the cross-signature, IdenTrust might have said, yeah, fine, why not? That sounds great. Well, nobody wants to cross-sign Let's Encrypt's cert now because essentially they're putting their reputation on the line for what we now know has been a crazy, like, insane abuse of this no-charge ACME bot-based self-certificate issuance.

So then they say, "Can we get another cross-signature? We've explored this option, and it seems unlikely." It's like, uh-huh, yeah, not surprisingly. Oh, and they said: "It's a big risk for a CA to cross-sign another CA's certificate, since they become responsible for everything that CA does." Which is to say, yeah, all the certs that that Let's Encrypt in this case Certificate Authority issues. They said: "That also means the recipient of the cross-signature has to follow all the procedures laid out by the cross-signing CA."

They said: "It's important for us to be able to stand on our own. Also, the Android update problem doesn't seem to be going away. If we commit ourselves to supporting old Android versions, we would commit ourselves to seeking cross-signatures from other CAs indefinitely." In other words, like just sort of give up on having their own, actual their own CA, and just always be obtaining CAs that haven't expired yet.

So they said: "It's quite a bind. We're committed to everybody on the planet having secure and privacy-respecting communications. And we know that the people most affected by the Android update problem are those we most want to help - people who may not be able to buy a new phone every four years. Unfortunately," they said, "we don't expect the Android usage numbers to change much prior to ISRG Root X1's expiration." Actually, that's a typo there. They meant the expiration of the other cross-signing.

They said: "But raising awareness of this change now, we hope to help our community to find the best path forward. As of January 11, 2021" - so okay, this coming January 11 - "we're planning to make a change to our API so that ACME clients will, by default, serve a certificate chain that leads to ISRG Root X1." Okay, now, which is to say their actual root. So they have been planning to change their certificate chain so that it chains up to their own sole ISRG Root X1. They recognize that that's going to cause a problem because of this immediate inability to get then to any of these Android, one-third of all Android devices that don't have ISRG Root X1 in them and never will.

So, they said: "However, it will also be possible to serve an alternate certificate chain for the same certificate that leads to DST Root X3" - that original cross-signed one, which will at least be lasting until September 1st, they said - "and offers broader compatibility." Uh-huh. One-third of Android devices. They said: "This is implemented via the ACME alternate link relation. This is supported by Certbot from version 1.6.0 onwards. If you use a different ACME client, please check your client's documentation to see if the alternate link relation is supported."

So this is a takeaway for our listeners who may be using Let's Encrypt certs. You want to make sure that you are running a late version of Certbot. And there's no reason not to just go ahead and select right today now, so that you don't forget. Well, I imagine we'll be talking about this next year because this is going to cause some problems for the world. But you want to be chaining up to the DST Root X3. I mean, why not? It's just as good. It's much more universally trusted. And you then obtain one-third of Android users who you would otherwise be losing on January 11th of next year.

They said: "There will be site owners who receive complaints from users." I don't think the users will be able to complain because they won't be able to connect. "And we are empathetic to that being not ideal." They said: "We're working hard to alert site owners so you can plan and prepare. We encourage site owners to deploy a temporary fix," they said, "switching to the alternate certificate root chain to keep your site working while you evaluate what you need for a long-term solution, whether you need to run a banner asking your Android users on older OSes to install Firefox" - and I thought, ah, isn't that interesting because this is exactly what we were talking about last week; right? Firefox has its own root CA. It already knows, all versions of Firefox have long known about the Let's Encrypt root and have incorporated it.

Anyway, they said: "...asking your Android users on older OSes" - that is, older Android OSes - "to install Firefox, stop supporting older Android versions, drop back to HTTP" - that's not going to fly - "for older Android versions, or switch to a CA that is installed on those older versions." Ouch. As in, you know, go get a certificate from someone else, not Let's Encrypt. Certainly they don't want that to be your choice.

They said: "If you get Let's Encrypt certificates through your hosting provider, your hosting provider may be serving the DST Root X3 until September 2021" - you hope, that would be good - "or they may decide to switch to the certificate chain that leads to ISRG Root X1 after January 11, 2021. Please contact them if you have any questions." In other words, make sure they're serving the older existing root. They said: "If you're on an older version of Android, we recommend you install Firefox Mobile, which supports Android 5.0 and above as of the time of writing."

They said: "Why does installing Firefox help? For an Android phone's built-in browser, the list of trusted root certificates comes from the operating system, which is out of date on those older phones. However, Firefox is currently unique," they wrote, "among browsers. It ships with its own list of trusted root certificates. So anyone who installs the latest Firefox version gets the benefit of an up-to-date list of trusted certificate authorities, even if their operating system is out of date."

They finished with: "We appreciate your understanding and support both now and over the years as we continue to grow as a CA, making sure people everywhere have access to encryption." And now we know why Chrome has decided to adopt their own root platform, their own root store. Chrome certainly is the default browser on all of these Android phones, and this allows - because Chrome, even though the OS will not be updating itself, at least the browser is. And so this allows Google to push a Chrome anytime between now and the end of next August, which will be bringing its own root store.

Now, this is not a complete solution for Android because of course you're still going to have an obsolete OS-level store. On the other hand, I would imagine Let's Encrypt is only being used, theoretically it could be used for other CA purposes, but almost entirely by web servers, which are only going to be serving to web browsers. And so as long as Chrome switches to their own root store, certainly Chrome will be updating itself. This solves the problem for the Let's Encrypt folks, at least on these older Android platforms, and so long as someone's not using something else. I mean, what would it be on Android other than Chrome and Firefox?

Leo: Oh, there's others. Plenty of browsers out there. So is this...

Steve: Popular, you think?

Leo: Yeah, yeah. I mean, if you're on Samsung, you use the Samsung browser, the Internet browser. I don't know which cert store they use. And there are other, yeah, there are plenty of browsers. But Firefox is probably the most obvious choice. It's like Android's not so different in the Windows world. There's dozens of weird off-brand browsers out there.

Steve: And non-Chromium based, because I would have had all Chromium browsers.

Leo: Well, is that the case? They're all using Google's cert store?

Steve: Well, we don't know for sure. And we talked about this last week. For example, would Edge on Windows use the Chromium certs, the new Chromium...

Leo: Yeah, because Edge is available on Android. I wonder, though. That's interesting. There's Brave, but that's Chromium-based, as well.

Steve: Anyway, a really, really odd, yeah, really odd consequence of being on an older OS platform where, you know...

Leo: Well, it's not the only one.

Steve: What has turned out to be a very popular CA is trying to...

Leo: Yeah. I use Let's Encrypt certs for a lot of stuff.

Steve: Yeah. I mean, they work. They're great. They're free.

Leo: Dolphin has been around a long time. I'm trying to see if Dolphin is Chromium. So you think anything based on the Chromium engine is going to use the same certs. I'm not sure that's the case. It's an interesting question.

Steve: Yeah.

Leo: Whose cert store does Chromium use? I don't know.

Steve: Well, right now, I mean, it's the underlying OS platform cert store.

Leo: Oh, it comes from Android.

Steve: What we know from - yeah.

Leo: So but if a browser comes with its own cert store, it would override that.

Steve: Correct. And I know, for example, Firefox does, and it always has. And what Google has said is that they are establishing their own root program just to have control. And a nice side effect is that then, if it were Chrome running on Android, then it would be bringing that along, and that one-third of Android devices wouldn't be having a problem.

Leo: Yeah.

Steve: Anyway, really, really sort of interesting whoopsie.

Leo: Yeah, kind of nasty, yeah.

Steve: You have to wonder, like, did it suddenly occur to them that, wait a minute, there's still a substantial population of Android devices, like one-third of them are not going to get updated.

Leo: Oh, yeah. Mostly outside the U.S., mostly probably by people on very inexpensive Android devices. So they're not super sophisticated. I don't think getting the word out, "Use Firefox," to them would be easy at all.

Steve: Right.

Leo: But they're already on horrifically insecure platforms. So, you know...

Steve: That's true. That's true.

Leo: Their life is miserable anyway.

Steve: Well, and it's not a matter of like their security dropping.

Leo: No, this doesn't affect security. They can't go visit sites.

Steve: Complete inability to connect to any site with a Let's Encrypt cert.

Leo: Right, yeah.

Steve: And, I mean, it's going to happen. That's a drop dead at the end of next August.

Leo: Right.

Steve: It's over.

Leo: Yeah.

Steve: So really interesting. While we were recording last week's podcast, unbeknownst to us, Google was busy releasing another emergency update to Chrome for Windows, Mac, and Linux. Which brought it up to v86, main v86, but ending in 4240.183. And we don't know much about the zero-day flaw that was found being actively used in the wild except that we know that it's CVE-2020-16009 is a problem in the V8 engine, Chrome's JavaScript engine, which is being used to enable remote code execution. So it was a zero-day, was discovered in use, and they immediately pushed out an update.

As I always do, I went to check my Chrome, and it turned out that, the 183 version of last week, last Tuesday, was already obsolete for me since, when I went to take a peek, it took that opportunity to move itself from 183 to 193. Which is now where we are, although there's no information, no more details about that one yet. So just a note that Chrome is moving forward.

Oh, and I'll also note that Android smartphone users should also be sure to poke their Chromes to verify that they are now running .185 or later to close a different Android-specific zero-day that Google found being exploited in the wild. There was a heap buffer overflow vulnerability in that Chrome, in the previous Chrome, for Android UI component. That was CVE-2020-16010. It was being exploited to allow attackers to bypass and escape from Chrome's security sandbox on Android devices and then run their code on the underlying OS.

So at this point Google's internal - they call it the Threat Analysis Group, TAG. And it's kind of cool because that gives them the acronym TAG Team. The Threat Analysis Group TAG Team has discovered not only this zero-day, the Android zero-day, the previous two zero-days, bringing their total to three zero-days in Chrome in just the past two weeks. Thus it appears that Chrome is finally receiving the attention from the attacker community that was once the province of IE. IE used to be attacked like this. Now these are all having been discovered in use in the wild at the time that they were found and fixed. So as we know, once upon a time IE was the big target. Now it's Chrome.

And speaking of targets, I've got a quick run-through, some high-profile ransomware targets. The famous toy company Mattel was a victim. We learned of it last Wednesday. They're a publicly traded company, so they're required to file a 10Q quarterly report. They acknowledged having been hit by ransomware at the end of July. They say that no data was stolen, and that they were able to restore themselves quickly.

Compal, which is the world's second largest laptop manufacturer, they're a Taiwanese-based firm. And they actually build the laptops for Apple, Acer, Lenovo, Dell, Toshiba, HP, and Fujitsu. So they're a biggie. They suffered a ransomware attack just this past weekend. They were a victim of the DoppelPaymer ransomware gang. Apparently about 30% of their computer fleet was compromised. They said it did not get into their production line, so none of the builds of laptops for companies was affected, and that they were 100% back up by yesterday. So no news on the details of how they got zapped. But we do know that it was this DoppelPaymer gang.

Capcom, which is the well-known Japanese game developer, was hit by the Ragnar Locker ransomware in an attack, and around a terabyte of the company's data was exfiltrated. Of course they're the well-known producer of Street Fighter, Resident Evil, Devil May Cry, Monster Hunter, and Mega Man game franchises. The bad guys got into their networks in the U.S., Japan, and Canada.

And finally, I got a kick out of Threatpost's headline for this one. They said: "Campari Site Suffers a Ransomware Hangover." And of course given that Campari is a famous producer of alcoholic beverages, including the brands SKYY, Grand Marnier, and Wild Turkey, a hangover seemed appropriate. Anyway, they restored their servers after also being hit by the Ragnar Locker ransomware and received the attacker's demand of $15 million in bitcoin. The attackers left a notice: "We have breached your security perimeter and accessed every server of the company's network in different countries across all your international offices."

And then the note goes on to detail the types of data compromised, including accounting files, bank statements, employee personal information, and more. The note said that they had obtained a total of 2TB of data and said: "If no deal is made, then all your data will be published and/or sold through an auction to any third parties." And I thought interestingly, as proof of theft, the group posted a copy of the contract between Wild Turkey and Matthew McConaughey. Which is to say, yes, in fact, we did actually get your data.

Leo: All right, all right, all right.

Steve: Here's the contract that Matthew McConaughey signed.

Leo: That's hysterical.

Steve: Yeah. So four more recent attacks. The gang behind the Ryuk ransomware alone, just Ryuk, reportedly averages 20 attacks like this every seven days, every week.

Leo: Wow.

Steve: I managed to pick up a bit of intelligence about Ryuk, the Ryuk gang's recent successes. The average payment received by this Russian-speaking gang is on the average 48 bitcoins, so that's currently - and by the way, bitcoin is back up to 15K, $15,000 per coin. And yes, we're shedding a tear, you and I, Leo, over our lost bitcoins.

Leo: Well, yours is lost forever. Mine is just locked away behind a password I can't remember. I'm not sure which is worse.

Steve: Yeah, I would love to get mine because back then I bet I know the password I used.

Leo: And you had 50-plus; right?

Steve: I formatted it and installed Windows over it. So it's definitely wiped.

Leo: Three quarters of a million dollars. But I'm not rubbing it in.

Steve: Okay. Ouch.

Leo: Ouch.

Steve: So yes, actually the average payment received by the Russian-speaking is 48 bitcoins, just about exactly what I formatted.

Leo: Ah, shoot. Gosh.

Steve: $720,000 at the moment.

Leo: Yikes.

Steve: And since 2018 they have netted at least $150 million. The word is they're exceedingly tough during negotiations, rarely showing any leniency or compassion. And the largest confirmed payment they are known to have received was, get this, 2,200 bitcoins, currently valued at $33 million. $33 million. So, boy. I like to sleep at night. I adhere to the old adage, "Crime doesn't pay." And I'm sure most of us are not tempted by the lure of the dark side. But neither is it difficult to imagine why ransomware, why the ransomware business is booming these days, as it clearly is.

Leo: I bet where they are, the chances of them getting arrested are minimal.

Steve: Oh, I'm sure Putin is jumping up and down, clapping, saying yay, you go get those...

Leo: For all we know, it's the GRU. It could be, who knows.

Steve: Those Yankees. Meanwhile, and this was interesting, there's a little bit of back story to this. Apple's move to 14.2, I heard you guys talking about it on iOS Today, early today. iOS users, of which I am one, may have noticed that our iOS devices were recently bumped from 14.1 to 14.2. As I said, there's a story behind that. The very short version is that this was done to close three zero-day vulnerabilities that were disclosed, or discovered rather, in use in attacks against iOS.

Leo: Ooh.

Steve: Yes. The much longer and more interesting version is that, according to Shane Huntley, the director of Google's Threat Analysis Group - again, the TAG Team - the three iOS zero-days are "related" to those recent three Chrome zero-days, and the Windows zero-day that we covered last week. So that means it was a big, well-coordinated, multiplatform campaign. Since Google and Apple have clamped down on any details, we don't know whether the zero-days were being used against selected targets or sprayed. But all iOS users will be wanting to upgrade to iOS 14.2 regardless. And again, I've had to ask my phone if it had any news for me, and it said, oh, yeah, thanks for asking. I've got an update. Would you like to download and install? I said, yeah, thank you. I want 14.2.

These same three vulnerabilities have also been closed in the most recent iPadOS and watchOS updates, and they've also been backported to older generation iPhones as iOS 12.4.9. What little we know from Ben Hawkes' Project Zero team is that the three iOS zero-days are - and we've got three stardate-looking CVE numbers. I won't bother with that. But first one, a remote code execution issue in the iOS FontParser component that lets attackers run code remotely on iOS devices. Second one, a privilege escalation vulnerability in the iOS kernel that lets attackers run malicious code with kernel-level privileges. And, third, a memory leak in the kernel that allows attackers to retrieve content from an iOS device's kernel memory.

All three bugs are believed to have been used together to form a highly - and I will underscore that word - sophisticated exploit chain. This would have allowed attackers to compromise iPhone devices remotely. And think for a moment about how difficult that is to pull off on a locked-down iOS device. I mean, to engineer the attacks with Windows, Linux, or Android, it's so much easier. All the code is just there to be poked at and prodded. But not on iOS. As we know, Apple has their devices so locked down, encrypted, and hack-proof that it takes some serious effort just to get a peek under the covers on an iOS device, let alone perform any sort of deep reverse engineering that's of the kind required to find a problem, and then turn it into anything like a reliable exploit. Almost all vulnerabilities are far easier to find and would be vastly easier to exploit once found.

So this feels like a world-class piece of work. For that reason I would bet that this was never being widely sprayed around the Internet because they did not want it to get - whoever they are did not want it to be seen and found and closed. While it was there and secret, it would have been incredibly valuable for enabling highly targeted information compromise on iOS devices. And I wouldn't be surprised if this was a nation-state-level attack that has now been shut down and closed.

Leo: Interesting. Interesting, yeah. On we go with the show, Steve Gibson.

Steve: So introducing the Tianfu Cup, and I checked my pronunciation on this Chinese word since I thought, okay, let's, I mean, it's not like that's a hard one to pronounce, but Tianfu, T-I-A-N-F-U. It's China's version of the West's Pwn2Own hacker competition. It was created two years ago, in 2018, following the Chinese government's regulation which barred their security researchers from participating in international hacking competitions over national security concerns. Our listeners may recall that we talked about the absence of the Chinese hackers, who are among the most talented and most successful in the previous Pwn2Own and other hacking competitions. It was like, oh, shoot, we don't have those guys this year. They're being kept within their own borders. But they're also being given their own competition.

So this was, this past weekend was China's third Tianfu Cup, a two-day event. Contestants from 15 different teams participated to deploy and demonstrate their discoveries of original vulnerabilities to break into widely used software and mobile devices. Each team was allotted five minutes and three attempts per category. The requirement was to use web browsers to navigate to a remote URL, presumably where they would have staged a server attack on the device, or to use local software to obtain control of the browser or the underlying operating system. The targets were software from Adobe, Apple, Google, Microsoft, Mozilla, and Samsung, all of which were successfully pwned utilizing previously unknown exploits.

The event's organizer said that: "Many mature and hard targets have been pwned during this year's contest. Eleven out of 16 targets were cracked, with 23 successful demos." So that was, what, an average of a little over two cracks per. And so the 11 that were cracked, there was Adobe's PDF Reader got cracked. The Apple iPhone 11 Pro running iOS 14 and Safari. That's interesting. The ASUS RT-AX86U router, I think that's the one I have. CentOS 8. Docker Community Edition. Chrome got cracked. Microsoft Windows 10 2004. Mozilla Firefox. Samsung Galaxy S20 running Android 10. TP-Link's TL-WDR7660 router. And VMware's ESXi hypervisor. All in an average of two cracks per.

The big winner was, not surprisingly, Qihoo 360's Enterprise Security and Government Vulnerability Research Institute. They often are finding things that we're reporting. They came out on top, winning just shy of three quarters of a million dollars, $744,500 in U.S. dollars. And they were followed by Ant-Financial Light-Year Security Lab at a little over a quarter million - yeah, I know, funny name - $258,000. And then a security researcher just named Pang came in at basically 100,000 - 99,500.

So the good news is they successfully pwned, at least an average of two times, all of those 11 devices. But patches for all the demonstrated bugs have been released. The exploits are being responsibly disclosed. Patches are coming, which we could expect to see in the coming days, or as soon as the various publishers of those devices are able to get to them. So as always, this kind of competition does get people to look harder at these various systems than they would otherwise if they didn't know that there was a nice carrot, a golden carrot that they might win.

Leo: That Pang. He's always coming in, taking the $100,000, every time.

Steve: He's a pang in the butt, Leo.

Leo: I love it. Just named Pang.

Steve: That's Pang.

Leo: Some guy named Pang.

Steve: Pang took away $100,000. There goes Pang again.

Leo: Nice work if you can get it. That's great.

Steve: So speaking of Patch Tuesday, Leo, today and in days following, many Windows machines will have the somewhat mixed blessing of receiving Microsoft's latest monthly insult of updates and new breakages. Sadly, as we've been seeing all year, updating promptly appears to be of increasing urgency every month, as the stakes in this game seem to have risen during 2020. So I expect that our next podcast, next week, will have a readout on some outcomes of this month's Windows 10 continuing update adventure.

But something I recently encountered was apropos of this. Woody Leonhard, a longtime valued participant and commentator in the PC industry, announced on Sunday his retirement from our industry. For those unfamiliar with Woody, two short bits from the Internet noted: "Woody Leonhard is a columnist at Computerworld and author of dozens of Windows books, including 'Windows 10 All-in-One for Dummies.'" And elsewhere it was written that "Woody Leonhard has covered Windows Dummies foibles and fantasies since the days of Windows XP. With more than a million regular readers, he's Senior Contributing Editor at InfoWorld and Senior Editor at Windows Secrets, where he weighs in daily on all things Windows."

Anyway, given his deep background, I mean, he's been in the industry forever. I thought that his parting characterization was interesting, if only to further assure us that we're not all crazy. When he announced his retirement, he wrote: "Life's changed in extraordinary ways since my first 'meatspace' book 'Windows 3.1 Programming for Mere Mortals' appeared 28 years ago." He said: "Windows has evolved from a rickety infrastructure built on top of a wobbly operating system to a wobbly operating system in its own right."

He said: "I don't miss the original bug-ridden incarnations of Windows. But I do miss the fire and vision that drove the unqualified success of Windows XP and Windows 7. And I'll continue to rail against the flaws that are introduced and sometimes re-introduced with every round of updates." He finished: "Microsoft has a long history of Windows patching issues. Some things never change, eh?"

Leo: What are you going to write in five years, when you write your retirement? It probably will be not much different.

Steve: No, exactly.

Leo: Yeah, some things never change.

Steve: Yes. Maybe I'll add something like, "Fortunately, there were some alternative OSes that it was a pleasure to hop over to."

Leo: Yes, maybe. Yeah.

Steve: Anyway, so a tip of the hat to Woody.

Leo: Great man. Great man.

Steve: Enjoy your retirement. The rest of us are all having too much fun to quit.

Leo: Right.

Steve: I think that's what this is, Leo. I think this is fun.

Leo: Yeah.

Steve: It is. It is.

Leo: Check. Just check. Just look. Make sure.

Steve: Are we having fun yet?

Leo: I'm having fun. I don't know about you, but I love doing this.

Steve: Yeah, I am, too. I am, too.

Leo: But you do all the work. I just sit here and eat popcorn while you give them the facts.

Steve: We had the Great Encryption Dilemma. The Council of the European Union last week published a short, five-page draft resolution with the title, and this was a new term, they said: "Draft Council Resolution on Encryption - Security through encryption and security despite encryption." So we know where that's going.

I read through the resolution, and there's nothing new there. But since the Great Encryption Dilemma remains outstanding and unresolved, I wanted to just stick a pin in this to note that the issue does remain alive and well, and also that I suspect it always will. And I wanted to take this occasion to be a little more clear and definitive about this than I have been previously. Cryptographers, and this podcast's audience, know with absolute clarity that this is a problem without a solution. Such things exist, and this is one of them. Bureaucrats, who are not cryptographers, are unable to accept the simple math of this fact. Once unbreakable encryption was created, that was the end of it. Game over. Enciphering algorithms without known weaknesses now exist. They cannot ever be made not to exist. Once something is encrypted with them, the only known way to reverse the encryption is with a key. Period.

Sure, a deliberately weakened encryption system could easily be created. We know how to do that. But that doesn't mean that the existing systems, the ones without any known weaknesses, can be uncreated. They cannot be uncreated. Governments and law enforcement may not be happy about what's been created, but it has been already. So the cat's out of the bag, the horses have left the stable, the chickens have flown the coop, the train has left the station, and the ship has sailed. This is done.

So hopefully this will forever remain a stalled issue, with academia and industry patiently explaining, over and over, as many times as necessary, to any government or law enforcement agency who asks, whenever it comes up, as it certainly will, that there's no safe way to add a deliberate backdoor into existing encryption. And even if there were, and this is the point you always make, Leo, that would not spontaneously uncreate any of the existing uncrackable encryption technologies that anyone could still easily and freely use. So that's just it. Done.

Leo: Yeah. If they outlaw encryption, only outlaws will have encryption.

Steve: That's the only thing that they can do. And exactly. If they were to outlaw it, only the bad guys would be using it.

Leo: Or math processors.

Steve: And the other thing, too, you know, if they want to have weakened encryption, are they willing to use it themselves? No.

Leo: No, of course not.

Steve: No. They want us to have a backdoor; but oh, no, they need to have it because, you know, they're in charge.

Leo: We're Congress. We're safe. No, no.

Steve: Okay. So this one is a little creepy. In Jackson, Mississippi, a small trial has been initiated and is being conducted for a month and a half, about 45 days, to explore the feasibility of allowing private citizens to have their video doorbells participate in police dragnet monitoring. The rationale is, while on the one hand municipalities might install video cameras pointing down all four directions of every intersection, which is by the way exactly what I've noticed being done here in Southern California, that doesn't provide as much granular video coverage as might also be afforded if all of the video doorbells in residential neighborhoods were also tied into a much larger surveillance network. We've talked about pervasive video monitoring before, and it's a little creepy.

But the EFF feels somewhat more strongly about this. They noted that handing over control of live streams to law enforcement may not only allow the covert recording of a willing participant's comings and goings, but also neighbors, which of course could happen. The EFF wrote: "The footage from your front door includes you coming and going from your house, your neighbors taking out the trash, and the dog walkers and delivery people who do their jobs in your street. In Jackson, this footage can now be live-streamed directly onto a dozen monitors scrutinized by police around the clock. Even if you refuse to allow your footage to be used that way, your neighbor's camera pointed at your house may still be transmitted directly to the police."

And in sort of an interesting coincidence, this past August Jackson city officials voted to preemptively ban police forces from using facial recognition technology to identify potential suspects on city streets. And although this is not that, it's getting close. And the month before, in September, an analysis leaked from the FBI highlighted how smart doorbells could also be turned against law enforcement as live feeds could warn suspected criminals of police presence, alert them to incoming visits from police, and might show suspects where officers are, which could pose a safety risk to law enforcement conducting property raids. So, yeah. This is increasingly feeling like a brave new world that we are in.

WordPress, as I promised at the top of the show, once again in the news for their Ultimate Member plugin, or maybe it's Ultimate Dismember. If you haven't yet been convinced by listening to me the last few weeks to sequester any WordPress instance so that its takeover cannot harm you further, here's another few reasons to consider either sequestering it or maybe evicting it completely.

Three separate supercritical flaws exist within another highly popular WordPress add-on known as Ultimate Member. They have CVSS severity ratings of 10, 10, and 9.9, each out of 10, of course. And the Ultimate Member plugin is installed on more than 100,000 WordPress sites. Each of the three critical security bugs allows for privilege elevation leading to full control over a WordPress site. The plugin allows, like the reason you have it is it allows web admins to add user profiles and membership areas to their WordPress sites.

And according to Wordfence researchers, the security guys, the flaws make it possible for both authenticated and unauthenticated attackers to elevate their privileges during registration to allow them admin status. Oops. And of course once an attacker has admin status on a WordPress site, they've effectively taken over the entire site and can perform any action they like, from taking the site offline to further infecting the site with additional malware.

So I'm not going to bother delving into all the details about each of the three vulnerabilities because I think that a broader point needs to be made. We've seen that the hacker community tends to focus on one category or another from time to time. For a while, earlier this year, RDP was under attack. Earlier than that, it was the router botnets attacking the authentication, the web authentication of routers. And tomorrow it'll be something else.

But the recent evidence suggests that WordPress plugins have been enjoying a period, up until recently, of relative quiet and under examination by that nefarious community. It feels like that community has recently awakened to just how much low-hanging fruit has been growing while their attention has been directed elsewhere. Last week a security vulnerability in the Welcart e-Commerce plugin was found to be opening WordPress sites to code injection. This led to payment skimmers being installed, crashing of the site, or information retrieval via SQL injection.

Last month, two high-severity vulnerabilities were disclosed in Post Grid, another WordPress plugin with more than 60,000 installations. It opened the door to site takeovers. And in September a high-severity flaw in the Email Subscribers & Newsletters plugin made by Icegram was found to affect more than 100,000 WordPress sites. In August, a plugin that adds quizzes and surveys to WordPress patched two critical vulnerabilities which could be exploited by remote unauthenticated attackers to launch a variety of attacks including full site takeover. Also that month, in August, the Newsletter WordPress plugin, with more than 300,000 installations, was discovered to have a pair of vulnerabilities leading to code execution and site takeover.

And before that, in July, researchers warned of a critical vulnerability in a WordPress plugin called Comments - wpDiscuz, which is installed on more than 70,000 websites. The flaw gave unauthenticated attackers the ability to upload arbitrary files, including PHP, and ultimately execute remote code on vulnerable website servers. I said before that WordPress is demonstrably a PHP-coded disaster, and that the tantalizing WordPress plugin ecosystem, which is I'm sure a large part of WordPress's allure, is also a hot mess. It's impractical to tell people not to use it. I get that. That is, not to use WordPress.

But don't, I would say to people, don't run a WordPress instance on your Drobo. I sort of chuckle when I see that in its menu of options. It's like, oh, yeah, let's run WordPress on our Drobo and install a bunch of add-ons. What could possibly go wrong? Or, for that matter, on any machine that has access to anything else. That's when I talk about sequestration. Depending upon how many tasty-looking goodies you add to your WordPress installation over time, there is a high likelihood of local site compromise sooner or later. That means that containment is the best you can hope for.

So please consider it. If you have WordPress running somewhere, somehow arrange a sandbox. Contain it. Put it on its own server. Hook it up to a separate port on your router that you're able to firewall so that, if anything gets loose in it, it can't get loose on your network. Do something. Don't install it side-by-side on a server with a bunch of other stuff there. That's just asking for trouble.

Okay. Off my soapbox. I mentioned one little bit of sci-fi miscellany: "The Saints of Salvation" is the title of Peter Hamilton's final third of his...

Leo: Is it out?

Steve: Next Tuesday.

Leo: Oh, good. I can finally read them. I've been holding off because I hate it when you've got two books of the trilogy...

Steve: And Leo, I was thinking of you. I've been waiting with great impatience. You know, like all of Peter's work, it is a truly remarkable - I can speak of the first two of the trilogy, which I've read, which I know John has read. It is a remarkable work of science fiction. And I read a lot of science fiction. In a world, ours, filled with lazy and largely derivative fiction, you know, I mean, it's engaging, it's entertaining, it's distracting. Peter somehow always arranges to create entire, fully realized, believable worlds and characters. And this "Salvation" trilogy is another one of those. He's done so again. It's annoying to wait long periods between installations of his multibook series. And when, in this case, when the second of these three, when the second book dropped, I reread the first one because it had been so long...

Leo: Right, to catch up, yeah.

Steve: ...I'd kind of forgotten it, to refresh my memory of what had happened. The good news is I didn't finish the second one that long ago. So I am just like, I'm on the edge of my seat. I mean, it's Lorrie's going to wonder what happened to me because it's like, "Okay, honey, I just have to read this now."

Leo: He's a machine because the first one came out October 2018, the second one October 2019. Here we are November 2020, I mean, it's boom, boom, boom. One year a book. He must just be - unlike somebody like, say, George R. R. Martin. He must really - he's just consistent. That's really awesome.

Steve: Oh, and Leo, they are so good.

Leo: Well, I have the first two on Audible. But I haven't listened because I don't want to - and I can start now.

Steve: You can start now.

Leo: I'm starting now.

Steve: Because you will not be finished by next Tuesday when you can get the third. And, oh, this is - and I mentioned this before. I didn't, you know, the thing he does is he runs several timelines at the same time.

Leo: Yeah, yeah.

Steve: There's the beginning of the whole adventure. And then he's also running way in the future. And so at first when you encounter the name of a planet that you've never heard of, it's like, what? You know, what happened to where we were just the last paragraph? But now that I'm - I guess I'll say I'm okay with the way he did it. It makes more sense to be showing us how all this trouble we're in began, and also the way far future that resulted from all of that trouble. So it's a little jarring. But once you understand that he's, like, starting you at the beginning and also the far, far future, then it kind of makes sense. And so I think it was the right thing to do. And I'll just say to our listeners, boy, there was "Pandora's Star," which was the first of two books.

Leo: Loved that. Loved that.

Steve: Oh, my god, I think it's probably my favorite. We were introduced to the Commonwealth with wormholes running trains through them, between widely spaced habitable planets. The one standalone book is "Fallen Dragon."

Leo: Wonderful, yup.

Steve: Which is excellent, with a surprise wonderful ending. And it was "Judas Unchained" was the second book of the - "Pandora's Star" and "Judas Unchained," that was the two books. And then the Salvation trilogy, as it's called.

Leo: The new one, book three, is "The Saints of Salvation."

Steve: Correct.

Leo: And that comes out November 17th. I have preordered it on Audible.

Steve: Oh, baby.

Leo: Then I'll have all three.

Steve: Yes, you can start listening because it is, oh, it is really completely new. This is not the Commonwealth universe. It's back to an Earth that we recognize, and then something happens.

Leo: I can't wait. "Salvation Lost," "Salvation," and "The Saints of Salvation." Peter F. Hamilton, the Salvation Sequence Series. Can't wait.

Steve: He did it again.

Leo: Oh, you know, this is - because I wanted to read this so bad. I even bought the books. But then I thought, I don't want to be disappointed when I get to the end.

Steve: No, you're right. And it's just so annoying to wait a year.

Leo: Yeah, yeah, yeah. Good.

Steve: Now you don't have to because you waited two years.

Leo: Yes, that's right. I distracted myself with other lesser books.

Steve: So Slipstream is a perfect name for this bit of cleverness. And let's all be thankful that Samy Kamkar, the security researcher who invented this, or discovered it, or engineered it, had the freedom to name this whatever he wished, rather than needing to dip into vulnonym to receive a name. Oh, my lord. I'll confess to being morbidly curious yesterday.

Leo: Me, too, yeah.

Steve: Yeah. So I went over, and I took a look at CERT's feed at that moment. I was greeted with Putative Loon, Pungent Pronghorn...

Leo: Oh, that's the vulnerability I want.

Steve: Oh, yeah, the Pungent Pronghorn. We also had Discordant Screamer, which I thought was interesting. And also Feckless Mongrel.

Leo: You Feckless Mongrel, you.

Steve: Yeah, and it's not clear to me that being told I've been a victim of the Feckless Mongrel attack demonstrates the achievement of CERT's intentions with regard to naming.

Leo: No, no, it's terrible.

Steve: So as you said, Leo, I think they chose a bad word list.

Leo: Yeah, yeah.

Steve: And they need to figure out what to do about that. But what did Samy come up with? His NAT Slipstreaming page states: "NAT Slipstreaming allows an attacker to remotely access any TCP/UDP service bound to a victim machine, bypassing the victim's NAT firewall," he says, "arbitrary firewall pinhole control, just by the victim visiting a website." Okay. So think about that for a minute. What that means is that you go to a website, and external malicious servers can obtain access to any machine and port on your LAN. That's bad.

So we've come to treat our NAT routers, as we know, as smart firewalls, which block all unsolicited incoming traffic by default, like bad guys trying to get into one of our machines where we don't want them to be. They do this for us automatically by operating a system of stateful packet inspection, where any incoming packets must be replies to recent outgoing packets; right? So this super elegant and simple scheme has served us almost without exception so far. I'll quote from Samy's summary, even though his terminology is a bit opaque and confusing. But don't worry, I'll explain with a really perfect example, and it'll all become very clear.

So Samy said: "NAT Slipstreaming exploits the user's browsers, in conjunction with the Application Level Gateway connection tracking mechanism built into NATs, routers, and firewalls, by chaining internal IP extraction via timing attack or WebRTC, automated remote MTU and IP fragmentation discovery, TCP packet size massaging, TURN authentication misuse, precise packet boundary control, and protocol confusion through browser abuse. As it's the NAT or firewall that opens the destination port, this bypasses any browser-based port restrictions."

Okay. So all that was confusing. And that's kind of in the weeds of what this is really about. So it's important, but it doesn't really matter. Anyway, he continues: "This attack takes advantage of arbitrary control of the data portion of some TCP and UDP packets without including HTTP or other headers. The attack performs this new packet injection technique across all major modern and older browsers, and is a modernized version of my original," he wrote, "NAT pinning technique from 2010, which was presented at DEFCON 18 and Black Hat 2010. Additionally, new techniques for local IP address discovery are included."

And he finishes: "This attack requires the NAT firewall to support ALG, Application Level Gateways, which are mandatory for protocols that can use multiple ports. That is to say, control channel and data channel, such as SIP, H.323, which are the VoIP protocols; FTP, our old friend; IRC DCC, et cetera."

Okay. So that's how we introduced this problem. To understand what's going on, let's talk about Application Layer Gateways as they have been sort of generically named. They exist because not all Internet protocols are as simple as HTTP, where we make an outbound query over a connection and receive a returning reply over the same connection. The best, most well-known example, certainly for us old-timers, is FTP, the File Transfer Protocol. With the original FTP protocol, which was designed before firewalls existed, the user's FTP client would reach out to a new FTP server to initiate a connection at that server's port 21. They would exchange logon username and password prompts and answers over the so-called "control channel," and upon agreeing on the transfer of data, that transfer would not occur over the current connection to the server's port 21, but from its port 20, the so-called "data channel." And most significantly, the remote FTP server would be the initiator of that data channel connection.

So we connect to the remote FTP server on its port 21. It subsequently connects back to us from its port 20 to a port that we designate. So what this means is that an FTP client would need to open a high-numbered listening port at its end. Remember that, as a client of an operating system, applications running on that OS are restricted from opening low-numbered ports below 1024. Those are reserved for privileged OS processes like that FTP server running on the remote server. So over the original connection to the remote server's port 21, the FTP client would say: My OS has given me port whatever, XYZ, where I'm now listening for your FTP server, your incoming return data channel connection. So please open a TCP connection to me at that port, and I will answer.

Okay. So in summary, the client sets up a high-numbered listening port, then initiates an outbound connection to a remote FTP service port 21. And among other things, it requests the remote server to call back to it at that high numbered port. So the bad news is that the operation of NAT that we've talked about often on this podcast is totally hostile to this wacky old active FTP protocol. That was recognized early on by the first implementers of NAT because it broke FTP, which back then was still seeing some use. I mean, it's still a viable protocol today. But as we've talked about before, not from your browser.

Okay. NAT broke it. So this was solved, or at least resolved, with a somewhat horrific kludge. NAT routers would notice when an outbound connection was being made to a server at port 21. They would then begin sniffing the outbound traffic over the FTP control channel, looking for the client's command to open the reverse data channel. And remember, that command would have the port that the client's OS had given the client to listen for a received connection. The router, sniffing that packet, would then on the fly patch in a WAN port on the router for the remote server to instead connect and query, and the NAT router would establish a NAT mapping from that WAN port back to the client's specified listening port.

So if you think about that for a moment, you'll notice that this is effectively deliberately and necessarily penetrating the NAT firewall for this connection instance, this back connection from the remote server. Since those early pre-firewall days, a number of other protocols have been designed which are similarly NAT hostile. And the need for a means of opening incoming ports has been generalized under sort of the umbrella of Application Layer Gateway since the control side of this resides, not at the packet level, but as with our FTP example, within the application's data layer, the FTP protocol, or the whatever, the SIP protocol, or the BitTorrent protocol, or RTSP protocol, whatever.

Wikipedia explains this whole problem this way. They said: "Application-level gateway - also known as ALG, application layer gateway, application gateway, application proxy, or application-level proxy - is a security component that augments a firewall or NAT, employed in a computer network. It allows customized NAT traversal filters to be plugged into the gateway to support address and port translation for certain application layer control/data protocols such as FTP, BitTorrent, SIP, RTSP, file transfer in instant messaging applications, et cetera.

"In order for these protocols to work through NAT or a firewall, either the application has to know about an address/port number combination that allows incoming packets, or the NAT has to monitor the control traffic and open up port mappings, a so-called 'firewall pinhole,' dynamically as required. Legitimate application data can thus be passed through the security checks of the firewall or NAT that would have otherwise restricted the traffic for not meeting its limited filter criteria." So a nice, succinct description of the whole thing.

So most clearly stated, the problem that Samy figured out, or the hack, the problem is that Application Layer Gateways attempt to be completely transparent to the application protocols they are proxying for. They're sitting there in our routers, enabled by default, hidden, powerful, and automatic. And because they're automatic and trusting, they can be readily spoofed by any inside agent pretending to need their help. "Spoofing" in this context means tricking the user's border NAT router into opening a packet return path through to any internal IP and port on the LAN of the attacker's choice. This subjects any services, any servers existing anywhere on a LAN to remote external access and abuse.

The second part of Samy's work was to clearly demonstrate that JavaScript code running in any of today's web browsers, even in a malicious malvertisement on an unwitting website, is all that's necessary to launch just such a spoofing attack. In other words, our web browsers themselves can serve as unwitting Application Layer Gateway spoofing agents.

So during his research, Samy explored all of the various ALGs. He looked at Linux's Netfilter ALG support. He reverse-engineered router firmware and finally settled upon the abuse of the SIP, the Session Initiation Protocol, as like a perfect target, widely supported. It has lots of uses. It's sort of a generic protocol that Application Layer Gateways support if they're going to do any support for SIP.

Samy wrote: "While we found some FTP functions, we're more interested in ports that we can use. Modern browsers prevent outbound HTTPS connections to a number of restricted ports, including FTP. So abusing the FTP Application Layer Gateway is likely a no-go." In other words, those are low-numbered ports, and browsers are already blocked. They're already smart enough not to allow code running in the browser to reach out to port 21 and do anything. So you can't use it. He said: "In 2010, when I first demonstrated NAT pinning, I used port 6667" - which of course was IRC - "via the DCC CHAT/FILE messages." He said: "Quickly, browser vendors blocked port 6667."

So the same thing has happened this time. In quick reaction to Samy's revelations, about two weeks old, all web browser vendors immediately announced their plans to block the TCP SIP ports 5060 and 5061, which are used in Samy's demonstrated attacks, by adding them to the browsers' existing restricted lists. So already they won't let you use FTP ports. You can't aim any browser traffic at 6667, thanks to what he did 10 years ago. And now, shortly, you won't be able to aim any traffic at ports 5060 and 5061.

Chromium's developer Adam Rice said: "As a workaround for the Slipstream NAT bypass attack, we will be blocking HTTP and HTTPS connections to the SIP ports 5060 and 5061. This will mean that connections to servers on those ports will fail." Once those ports have been added to the restricted ports list, Rice expects some impact to be observed by browser users. For example, connections to servers on those ports, like if you tried to go to http://example.com:5060 or https://example.com:5061, will no longer work. And that's probably not a big problem.

But if test servers happen to be spinning up instances on those particular high-numbered ports, that would be a problem in the future. The development teams behind Firefox, Safari, and Blink, which is the Chromium rendering engine, have all indicated their intent to implement these mitigations needed to block these NAT slipstreaming attacks.

And the takeaway for our listeners is that it's probably worth going to the trouble of turning these ALG supports off in your routers, your border routers, if you have them. I'm using pfSense at this location. Actually I'm using pfSense at both locations, so nothing incoming would work. But I do know that the ASUS router has a tab where there's a list of, I don't know, seven or eight of those things. I remember I turned off IPSec, thinking that I didn't need it. But I'm using one of those, as I mentioned once before, one of the little Verizon, they're not called nano...

Leo: Microcells or something, yeah.

Steve: Yeah, the little...

Leo: Femtocells is the technical term.

Steve: Femto, yeah, femto, exactly.

Leo: But Verizon has its own name.

Steve: Right. And we're in a really bad reception area, so I'm using a femtocell for us. And it requires IPSec. So I was like really puzzled. It was like, wait, how did I break that? It was like, oh, and then I remembered that I had turned it off. But there's typically a list of those things, like you may have a checkmark for SIP. You may have a checkmark for IRC. My point is, if you're not using those things, this is another example of turn stuff off that you're not using.

And so I know that there's a bunch of things that I could turn off in my ASUS router. And I turned them all off. Then it was like, then my femtocell broke, so I had to turn on IPSec. But the rest of them are still off because I don't need those other things. And this is an example of just not having unneeded services available, just protecting you, because today we have this. Who knows what we're going to have tomorrow. Our browsers will shortly be fixed. In fact, I didn't think of it, but maybe that's what happened from the 183 to the 193 change that I mentioned from last week to this week. Maybe those ports are already being preemptively blocked by Chrome because we know that they would be quick to do that.

So that's what the Slipstream attack is. It is a clever way of abusing, essentially using your browser to pretend to be a client of one of these Application Layer Gateways, all of which had the ability, by spoofing what's going on, to trick the router into opening a return path back through the router, which the browser would be able to aim at anything on your network. So it's certainly possible that bad guys could use it to get up to some mischief.

Leo: I'm just busily trying to find out if my Ubiquiti supports Application Layer Gateway, and which ones - I know SIP is in there, but that's not turned on, obviously. So most of that stuff you wouldn't turn on. DCC for IRC, H.323 for VoIP, FTP, it seems like.

Steve: Well, exactly. But the problem is most of them are on by default from the factory because they don't want the installation of their router to break something.

Leo: Right, right.

Steve: And so for the naive user, okay, that's the right thing. For our audience, you know, go pry that stuff out with a pitchfork.

Leo: Yeah. I'm just trying to figure out where that would be. But I will find it. And chaos...

Steve: And same thing on that ASUS router. You've got to dig in a few...

Leo: Yeah, I'm sure it's hidden away, yeah.

Steve: There are so many bells and whistles on the routers these days.

Leo: Yeah. The beauty of the Ubiquiti is I can do it from here. I can log into my router at home from here and see what's going on. So I'm just - I'm looking at it. Probably advanced features or something like that. Oh, yeah, Advanced Gateway Settings. Bet you that's it.

Steve: ALG. Gee, yeah.

Leo: Well, anyway, we've got to finish the show.

Steve: Cool.

Leo: I'll do that after. I'll do that later. Hey, that's it for this episode.

Steve: As they say, "That's the show."

Leo: That's the show, baby. And that's the guy, the guy behind the curtain. He's Mr. Steve Gibson. You want to know more, you go to GRC.com. That's his home on the Internet, the Gibson Research Corporation. That's where you can find SpinRite, the world's best hard drive maintenance and recovery utility. He's working hard on 6.1. 6.0 is available. You get a free upgrade.

Steve: Am.

Leo: M?

Steve: Am. I am.

Leo: Oh, I thought there was some code, M.

Steve: Version M.

Leo: M. Yes, he is. Yes, he is working hard. 6.1 will come soon, but you will have a free copy if you buy 6.0 now. Steve also has this show there. In fact he has unique versions of the show, a 16Kb audio version for the bandwidth-impaired; beautifully written transcripts by Elaine Farris so you can read along as you listen; and of course 64Kb audio. GRC.com.

We have audio and video at our site, TWiT.tv/sn. Of course, the easiest thing to do would be subscribe in your favorite podcast app. You'll get it automatically, the minute it's available. We do the show every Tuesday afternoon, about 1:30 Pacific. It varies, depending on the shows before it. But about 1:30 Pacific, 4:30 Eastern time, 20:30 UTC.

And the reason I mention our live production time is because you can watch us do it live, if you're in a big hurry. The stream's at TWiT.tv/live. There are audio and video streams you can choose from there. People who watch live often like to chat live, which would be irc.twit.tv. Yes, it's the good old IRC. No DCC required. Don't worry. I don't think; right? We don't need DCC. ScooterX, correct me if I'm wrong. We also have on-demand versions. I mentioned that. Oh, and we have a great forum. Steve's got his forums. They're fantastic. We have ours at www.twit.community. Okay. Whew. That being done, I think it's time to say so long, Steve.

Steve: Mission accomplished.

Leo: Mission accomplished.

Steve: Until next Tuesday with the release of the final book in the trilogy.

Leo: I know what you're going to be doing Tuesday night. We'll see you next time on Security Now!.

Steve: Tonight I'll be finishing watching "Queen's Gambit."

Leo: Oh, yeah, what a good show that is, yeah.

Steve: Ooh.