Listener Feedback #160

Description: Steve and Leo discuss the week's major security events and discuss questions and comments from listeners of previous episodes. They tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world 'application notes' for any of the security technologies and issues we have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-389.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-389-lq.mp3

SHOW TEASE: It's time for Security Now!, a really big show for you. We're going to talk about a new vulnerability in almost every router, on almost every computer, on almost every television set and toaster and microwave oven that allows bad guys to work their way with your network. The problem with UPnP, coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 389, recorded January 30th, 2013: Unplug UPnP.

It's time for Security Now!, the show that covers you and your loved ones in an invisible shield, a barrier against the malware, the viruses, the privacy invasions the Internet is prone to. And we all owe a big gratitude of thanks, a heaping gratitude of thanks to Mr. Steve Gibson.

Steve Gibson: A heaping helping.

Leo: A turducken stuffed with three kinds of gratefulness to Steve Gibson. He is the Explainer in Chief and our security guru. Hello, Steve.

Steve: Hey, Leo. Well, we're here for Episode 389, which is a Q&A, our 160th Q&A. And we have big news this week. Really interesting news. It's been funny because there's been a Twitter storm of people saying, oh, my god, do you know about this? This was something that just came out yesterday. Our old friend HD Moore - who is the famous original author and curator of the Metasploit Framework, which we've spoken of many times in the last eight years, and HD pops up on the radar from time to time. He finished, about a month ago, a five-and-a-half-month scan of the entire Internet for

exploitable Universal Plug and Play router ports.

Leo: Oh, dear.

Steve: And he found 81 million.

Leo: [Laughing] That's not individual routers, though; right?

Steve: Yes. 81 individual routers.

Leo: 81 million individual exploitable routers.

Steve: 2.2 percent of the entire public IPv4 address space has open exploitable Universal Plug and Play routers on them right now. It's a disaster.

Leo: And when you say "exploitable," is it theoretically exploitable, or there are known exploits that can take advantage of these...

Steve: Known exploits.

Leo: Oh, crap.

Steve: We're going to go into it in detail at the top of the show. And I have no choice but to quickly add scanning for that to ShieldsUP!. So that'll be the first change I make to ShieldsUP! in years.

Leo: Oh, marvelous.

Steve: I mean, that's what ShieldsUP! is for...

Leo: That's fantastic.

Steve: ...is to allow people to check themselves. So I'm going to do that.

Leo: I'm surprised it's so low, if you say it's only 2 percent. I would have thought that virtually all routers - isn't UPnP on by default on routers?

Steve: No, but this is it. It should never be on the public side. It should never be exposed to the Internet.

Leo: Oh, you're saying that you could, from the outside, use UPnP.

Steve: No. Any hacker, Leo, can take over people's private networks. 81 million of them.

Leo: That doesn't sound good.

Steve: No, this is really bad.

Leo: That doesn't sound good, yeah, yeah.

Steve: And we've talked about this. We've known about this. But what was funny was I saw some - there were some dialogues happening in Twitter where people kept the @SGgrc in their dialogue. And they were saying, yeah, you know, Gibson told us about this two years ago. It's like, well, yeah. And I wrote, of course, Unplug n' Pray, famously, 11 years ago when Microsoft first screwed up with their initial UPnP server. So anyway, you wanted to talk to our listeners first, but then we'll get into all this. And we've got lots of interesting - I did 11 questions today because a couple are just kind of fluffy. It was like, oh, well, they're fun, so...

Leo: You really want to give value for dollar, Steve.

Steve: Ah, we're going to give them 90 minutes no matter what.

Leo: Not time, value. Just value...

[Talking simultaneously]

Steve: Oh, and by the way, it looks like our listeners all just went to ShieldsUP!. But don't go there yet. I have not...

Leo: Oh, Steve.

Steve: I have not written that yet.

Leo: Have you not learned?

Steve: I didn't know.

Leo: Have you not learned? We brought down, before the show - are you going to

mention this underwater cable site on the show?

Steve: I think we should because people will...

Leo: We almost brought it down, we might as well continue to keep it down.

Steve: Absolutely. You tell them.

Leo: Before we do that, let's talk a little bit about our friends at Audible.com, a great place to go. Well, I'm going on a midnight flight, for instance, and that means...

Steve: To Georgia?

Leo: Pardon me?

Steve: No. Not to Georgia.

Leo: Midnight flight to Georgia. Well, actually, as a matter of fact, I am. I'm going to Atlanta, and I'm then changing planes for New Orleans. So you're right, I am on a midnight flight to Georgia. At midnight tonight.

Steve: That would be when, yes. Thank you for doing the podcast first, Leo, because at the rate we're going, this is going to be a high-value production.

Leo: All right, Steverino. Fire when ready.

Steve: So, lord knows this is not any news to our listeners. How many times have we talked about the fundamental problems with the design of the Universal Plug and Play protocol? The whole concept - and this is it, I mean, it arose from the typical tension which exists between ease of use and security. Standard, one of our fundamental lemmas of security is that there is going to be competing interests, I guess, between something that's easy to use and also secure. So, for example, if in a state-of-the-art home environment you've got everything on your network - and, my god, anything you buy now comes with WiFi built in - the televisions, the toaster ovens, you name it, it's got WiFi for some reason. And in order for these things to self-configure, of course they have DHCP, Dynamic Host Configuration Protocol, which allows them to ask for an IP from your router.

But they also universally now have Universal Plug and Play, which is a system that allows those devices to be discovered, which is the actual terminology used, "discovery," on the 'Net. It's possible for, for example, Windows or a Mac or a Linux machine to send out a discovery packet onto the 'Net. Actually on a local network you can use a broadcast packet, which we have talked about in the past, where it's essentially addressed to

everybody. And so the query is who has UPnP? And those devices that have Universal Plug and Play will respond to that query. And then the computer starts to talk to them one at a time, asking them about their capabilities, what sort of interfaces do they have and so forth.

So the problem with that is that, well, there are several problems. First of all, computers respond, too. So if malware got into one computer, it could send out a query for all the other computers which are currently on and use the Universal Plug and Play to put holes in their firewalls and access them directly because that's one of the main capabilities of Universal Plug and Play is router configuration. For example, BitTorrent is a well-known deliberate Universal Plug and Play client. So when you run BitTorrent on a machine in your network, it tracks down your router through UPnP, and it says, hi there. What is your external IP address? Because that's not something which is known inside the network. Inside the network you only have private IP addresses.

But BitTorrent needs to know your current public external IP address so that it can tell other BitTorrents out on the Internet how to contact you. So it asks the router, what's your public IP? Thank you very much. Now, please give me an open port. And so the router configuration is changed through this Universal Plug and Play dialogue to allow BitTorrent to receive unsolicited incoming traffic to the IP that the router has just disclosed to it and route it to the machine running BitTorrent so that it's able to knit itself into the big BitTorrent network in the cloud. So there's a typical example of the way this should work.

At no point in time has it ever been suggested that any of this should be functioning on the public interface, that is, on the outside facing surface of the router. All of this is - Universal Plug and Play only makes sense, it's only designed for inside your home, inside your office, inside your LAN, your local area network, never on the WAN, on the wide area network, on the Internet.

It turns out that through just not caring, I mean, the only way to explain this is that no one cared, a huge number of routers do have Universal Plug and Play exposed, that is, this same interface that I was just describing, very powerful, on their public side, on the public-facing interface, such that, if a hacker knew the router's IP address, that hacker could send Universal Plug and Play queries to the router with the same level of control and power as happens inside because there's no password. There's no security. There's no logon. And here again, I mean, it would be nice if there were. But if there were, then Universal Plug and Play wouldn't just work by itself, by magic. And when it's in your home network, it's because there's no security, there's no password, no authentication at all, it is authentication free, that's what allows all of this just to work automatically.

And of course we know, we've talked about it before, the Xbox, for example, game console famously needs UPnP enabled on the router. For a gaming network knitting themselves together, it needs to open ports through the router to itself so that incoming traffic can get to it. So, well, and Skype, same thing. One of Skype's early successes was that it was able to connect to Skype instances behind NAT routers. And there's a bunch of ways, sort of flaky and not really reliable, but they kind of mostly work for arranging, if you don't have Universal Plug and Play, to connect two endpoints behind routers.

But because it doesn't always work, the Skype system also had to have so-called "relay nodes" which were outside, where, if two endpoints behind routers could not get to each other, then they would relay through this other point. And that's why you and I, Leo, have famously selectively opened some ports that we use with our Skype clients who give us a non-relay connection because you get a much better Skype result. What happened was, as Universal Plug and Play became more popular, Skype added that

technology. So now Skype is a Universal Plug and Play user, and it's able to solve this problem by itself by opening ports and finding out from the router, what's my public IP, and then sending that to Skype Central so that somebody who wants to connect to you knows where to send the packets and which port the router opened for that stuff to come in.

Leo: So that's a good thing. I mean, it's a convenience.

Steve: Massively convenient. But...

Leo: And it even required, with some, like, gaming boxes like the Xbox, we've talked about this before, I think they invented, Microsoft invented this for the Xbox because, if you want to Xbox live and play games against other people, you need to open those ports.

Steve: Yeah, yeah. So, but the key is no authentication. Anybody can access it. And if this stayed constrained to your private network, that would be enough concern. Now, listeners who've been paying attention will know that we have talked, I think it was maybe a couple years ago that it came to light that some routers had Universal Plug and Play enabled on their public interfaces. And I remember clearly saying to our listeners, oh, my god, make sure that's not you. Log into your router, see whether there's a way to disable UPnP on the WAN. The problem is there typically is not because it's not supposed to be on in the first place. It's a mistake. It's an oversight.

So HD Moore somehow, something, who knows why, what the back story is, he isn't saying, or hasn't said, but starting June 1st of 2012 he launched a project to start scanning the entire public Internet, the IPv4 address space, so four billion IPs, to start scanning it to find out what was out there on UPnP. And he was scanning at a speed, and with enough servers, I don't know how many or how fast or anything, but he was able to touch every IP on the public Internet about weekly. This went on for five and a half months, from June 1st, and concluded around mid-November of late last year, 2012. He found 81 million Universal Plug and Play exposed routers. Now, the number should be zero. And there's 81 million. So what that says is that 81 million routers will respond to a hacker's Universal Plug and Play query and say, yeah, I'm here, what do you need?

Leo: How can I help you?

Steve: How can I help you today?

Leo: What can I do to make your day better?

Steve: My configuration is your configuration.

Leo: Oh, my god.

Steve: The response...

Leo: But this is different from - I just want to understand this. This is different because, if you have UPnP turned on in your router, this does not mean you're opening yourself up to the outside world. It means you're opening up yourself to things inside your network who are reconfiguring your router.

Steve: Correct.

Leo: So how does it happen that the router opens UPnP to the outside world? Is this WAN administration? What is this saying?

Steve: There is no - there is absolutely, I mean, think about it, think what this means. This is your router is promiscuously responding...

Leo: Hey, big boy.

Steve: ...to unsolicited incoming nonsense...

Leo: Wanna sniff my ports?

Steve: ...from the public - sniff my ports - from the public Internet and saying, hi there. What would you like to change?

Leo: So this is a flaw in the fundamental design of the router.

Steve: Yes.

Leo: Do we know the manufacturers that do this?

Steve: Oh, yes. All 1,700 of them.

Leo: It's a bug. Is this a bug, or is this a feature?

Steve: Oh, I'm sorry, it's 1,500 vendors and 6,900 products.

Leo: I'm guessing Cisco and Linksys are one.

Steve: Well, yeah, gee, Leo, remember, was it last week or the week before we talked

about how many of them hadn't even bothered to fix the WPS flaw?

Leo: Right.

Steve: Okay. So I'll just share the executive summary from HD Moore's paper. I tweeted a short bit.ly link to the PDF a few hours ago, so it's in my Twitter stream. It's bit.ly/upnpflaws, all lower case. So it's bit.ly/upnpflaws. And so his executive summary, that's not long, starts out:

"Universal Plug and Play (UPnP) is a protocol standard that allows easy communication between computers and network-enabled devices. This protocol is enabled by default on millions of systems, including routers, printers, media servers, IP cameras, smart TVs, home automation systems, network storage servers. UPnP support is enabled by default on Microsoft Windows, Mac OS X, and many distributions of Linux." Because, Leo, why wouldn't you want it?

Leo: But it's not an issue if your router doesn't put this to the outside world. I mean, if your...

Steve: Correct.

Leo: Yes, if your TV, if your smart TV is plugged in directly to your cable modem or your Mac or your PC...

Steve: Well, unless you have both. Because then a hacker in Russia can log into your smart TV directly.

Leo: Right. But if you had a router that didn't do this, you'd be okay; right?

Steve: Correct. And now you understand why top priority for me is adding detection of this to GRC's ShieldsUP! system so that everyone will be able to immediately determine whether they're exposed. Anyway, so continuing:

"The UPnP protocol," he says, "suffers from a number of basic security problems, many of which have been highlighted over the last 12 years. Authentication is rarely implemented by device manufacturers" - I didn't even know there was any - "privileged capabilities are often exposed to untrusted networks, and common programming flaws plague common UPnP software implementations. These issues are endemic across UPnP-enabled applications and network devices.

"The statistics in this paper were derived from five and a half months of active scanning. UPnP discovery requests were sent to every routable IPv4 address approximately once a week from June 1 through November 17, 2012. This process identified over 81 million unique IP addresses that responded to a standard UPnP discovery request." And remember, that number should be zero. It was 81 million. "Further probes determined that approximately 17 million of these systems also exposed the UPnP Simple Object Access Protocol (SOAP) service to the world. This level of exposure far exceeded the

expectations of the researchers.

"This paper quantifies the exposure of UPnP-enabled systems to the Internet at large, classifies these systems by vendor, identifies specific products, and describes a number of new vulnerabilities that were identified in common UPnP implementations. Over 1,500 vendors and 6,900 products were identified that are vulnerable to least one of the security flaws outlined in this paper. Over 23 million systems were vulnerable to a single remote code execution flaw that was discovered during the course of this research." 23 million remote code execution in their routers.

"Rapid7 worked with CERT to notify the open source projects and device manufacturers vulnerable to the issues described in this paper. Unfortunately, the realities of the consumer electronics industry will leave most systems vulnerable for the indefinite future. For this reason, Rapid7 strongly recommends disabling UPnP on all Internet-facing systems and replacing systems that do not provide the ability to disable this protocol.

"Rapid7 has provided a number of tools to help identify UPnP-enabled systems, including the free ScanNow for UPnP, modules for the open source Metasploit Framework, and updates to the Nexpose vulnerability management platform."

So to summarize, 2.2 percent of the total IPv4 address space has exposed Universal Plug and Play services, which is 81 million unique IPs. About 20 percent of those expose what's called the SOAP API. What happens is, when you send a UDP packet to the router, the so-called "discovery" packet, it responds with an identification string telling it what version of Linux it's running, what version of which UPnP library it's running. I mean, it tells you everything you need to know to then send the next exploit packet to it. But it also tells you where it's running its HTTP web service because that's the second part of Universal Plug and Play.

The first is you use UDP to find the device. Then it tells you which port its little web server is listening on. So isn't that handy. So the attacker receives that information and then makes a port 80 or actually port whatever, it's over the HTTP protocol, but not a port 80. Most manufacturers have their devices randomly choose a port so that it's not obvious where it's going to be all the time. But all you have to do is ask it, where is your HTTP service, your UPnP HTTP service, and the router tells you.

So then you connect to that. And over a well-understood, well-known API called SOAP, which is sort of an XML-ish API, you're able to dump out the router's configuration, make whatever changes you like, and send them back. And the router will, just as it does if you're an Xbox in the network, or if you're Skype in the network, or if you're BitTorrent in the network, it doesn't care. It makes whatever changes in its configuration have been asked for. Meaning, essentially, that it takes the router out of the way for anyone who wants access to your network internally, and all the devices on it are then available.

So in their analysis they found that four development kits accounted for nearly three quarters, 73 percent, of all the discovered Universal Plug and Play instances. So from their scan and the fingerprints that they were able to obtain, mostly from the thing just saying this is what I have, in the first response package just lays it all out for you, they determined where the software came from that they were talking to. And there are - unfortunately this is very heterogeneous, I'm sorry, very homogeneous, not very heterogeneous, meaning very few kits have been used across most of these. And they're buggy as hell.

So they found - there's one kit called the MiniUPnP, which 332 different products use. 69

percent of those still have v1.0, which has multiple exploitable flaws. A different library, which actually came originally from Intel and then became the Portable UPnP SDK, then it went open source and is called "libupnp," 23 million fingerprints out of the 81 million that replied match a version of UPnP that exposes the system to remote code execution.

And then, get this, Leo: Only one UDP packet is all it takes to exploit any of the eight newly discovered libupnp vulnerabilities, which as I said affect 23 million routers out on the Internet, and since it's a single UDP packet, you can spoof the source address. So it's impossible to know where it came from. Someone drops it on the Internet, aims it at you, and it executes a vulnerability that exists in your router.

So 11 years ago - I've seen some people say, hey, Steve, doesn't Unplug n' Pray, your free utility from December 28th of 2001, doesn't that fix this? No. The problem was that, in classic Microsoft fashion, whatever it was, it must have been XP, an early version of XP, first offered, whoa, the Universal Plug and Play interface. Isn't this wonderful. We'll be able to configure everything automatically. They had a bug in their Universal Plug and Play service which allowed that computer to be immediately taken over. And at the time there were no other Universal Plug and Play devices anywhere. This was the first instance of Universal Plug and Play. There were no TVs. There were no toaster ovens. There were no BitTorrents at the time, or Skype or anything. Nothing else existed that used it. Yet Microsoft, forward thinking, had it turned on even if you didn't need it, and it could be exploited remotely.

Leo: I'm really convinced this goes back to Xbox and Xbox Live, that they wanted - Xbox Live is the ability to play videogames against other people on the Internet. And of course...

Steve: Yes. Remotely configuring your router is an absolute need. I agree.

Leo: Yeah, well, it's not a need, but it's certainly - most people who buy Xboxes are not going to be sophisticated enough to do that. And if you do start your Xbox, and you say, I want to - and by the way, this goes back to the original Xbox, as well - and you say I want to play games online, it will say - it says, let me check. Oh, you can't, you've disabled UPnP. Please reenable - they call it something else. But they say please enable it so that we can configure your router so that we can do it. So, I mean, I remember getting calls from radio listeners regularly saying, yeah, I'm getting this error message on my Xbox. And if you want to, you can send out outgoing game requests, but you can't accept incoming game requests, obviously.

Steve: Yes. And there have been articles published which show you how to manually configure a bunch of ports. It's not simple. It's not like they were...

Leo: Port-forwarding, all routers support this. I mean, it's not impossible.

Steve: Yes.

Leo: And so that's what I would end up telling people because I learned from you

not to say don't turn that on, but here's what you need to do, you need to port-forward your router.

Steve: Yes. So...

Leo: But blame Microsoft. It's their technology, and they actively pushed it on people. I mean, really actively. Like insisted on it without any mention of the potential risk.

Steve: Right. And you can imagine then from the viewpoint of the router vendors, this was a feature that their routers had to offer because everyone wanted it. Everyone was demanding it. We need our routers to be autoconfig or whatever it is, was probably what the Xbox...

Leo: Who wants to make the router that doesn't work with Xbox Live? Not you.

Steve: Yup. Yup.

Leo: So it really is shameful. Really.

Steve: Yup. So I looked over the source code which HD Moore has some snippets of in this PDF report. And the mistakes, okay, they're not even mistakes. They're not subtle. And the fact that this was - this portable SDK for UPnP devices originally came from Intel.

Leo: It was a reference library. It's what everybody uses.

Steve: Yes, well, it was a reference implementation. And when you look at it, it is so devoid of real-world safety checks that I don't think it was ever intended to be used.

Leo: Right. It's just this is how you would do it were you to implement this on your own device.

Steve: But of course we removed all of the extra extreme...

Leo: We made it simple so you can understand it.

Steve: Right.

Leo: And people just used it.

Steve: Exactly. I think that's how this happened.

Leo: So it's really twofold. It's the fact that Microsoft had UPnP and encouraged people to use it, and then a bug in the UPnP library...

Steve: Oh, not "a." Not "a."

Leo: Several bugs.

Steve: Oh, my god.

Leo: That made it possible to do inbound UPnP as well as outbound UPnP.

Steve: Yes. Well, yes. It should never be "bound," that's the term, you should never have the UPnP service bound to the public interface. It was always meant to be on the private interface, the LAN inside interface. It makes no sense to have it on the WAN. Yet, who knows? People took an implementation that was never - from Intel originally that was never meant to actually be used in the real world. And because, I mean, it doesn't have any checks. None. No checks for buffer overrun.

Leo: Oh, dear.

Steve: There's a while - no, I mean none. There's a while loop that is scanning for the string you've received, looking for a \r\n, a carriage return/line feed.

Leo: It doesn't sanitize the inputs at all?

Steve: No, not at all.

Leo: Just send me anything. I'll just look for it.

Steve: And so this while loop doesn't have any other constraints on its execution.

Leo: So that's not a dumb programmer. That was never intended to be in the wild.

Steve: Correct. The only - when I'm looking at the source code...

Leo: That's like taking example code from a book and putting it in your production product.

Steve: Exactly right. And we've often seen books that say, here's - and you can see that Intel could do this, saying - and it was probably in the footnote or in the text, but that they didn't read. They just copied the code where it said this is an example of how to use the API. We've left out all of the checks and balances that you would normally use in real production code because that's, as anyone who's done that kind of coding, you take something simple, and it hugely complicates it to check for every possible thing that could go wrong.

Leo: And it [indiscernible] it if you're doing example code. It just gets in the way. You don't want that.

Steve: Yes, exactly, because you think, wait a minute, what's this about? It's like, oh, this is to check to see if I...

Leo: This is a range check. I don't need that, yeah.

Steve: Right. And so this very simple, I mean, simplistic code is in production. It's in people's routers. Oh. And it's bound to the public interface. So there are a few libraries. And HD Moore and his group looked at them closely, worked with CERT, also CERT/KR that must be Korea and CERT/CN that must be China, trying to get the word out. The libraries that they have had access to have fixed all the bugs they know of. But this was just a visual inspection. This wasn't, like - it's just really not their job to fix these libraries. But when they were looking at the code, they're like, oh, my god, how can this be?

And so those things have been fixed. And in fact even this MiniUPnP library had been fixed, I think it was like several of the things they found were fixed in '08 and '09, but 69 percent of the current MiniUPnP on the 'Net, which was 14 percent of the total fingerprints collected, were still using 1.0 that had none of these fixes. So, and this is the point that he makes in his executive summary at the beginning is that we think of these as little plastic boxes. No, you know, I mean, there isn't the same, except for within our own - within our Security Now! community certainly people know about keeping their router firmware up to date and the importance of that. And we don't - we wish that, for example, Cisco had published router firmware to fix the WPS security problem more than a year ago, but no.

So there's, like, they sell them, and they forget them. They don't know who their customers are. And more or less the customers just think of it as an appliance that they plug in. Their Windows box is busy updating itself multiple times a day. Chrome's version number has gone into seven digits now. But the router is sitting there with version 1.0 code that was never meant to be made public, that is exposing itself to the Internet.

So under mitigation, what do we do about this? The good news is that this all starts with a UDP port 1900 query. That's the port, port 1900 of UDP that all these boxes are listening on. That was assigned in the same way that the web uses 80 and POP uses 110 and so forth. UDP port 1900 is Universal Plug and Play. ISPs we know are blocking 137

through 139 and 445.

Leo: That's NETBIOS, and what's 445? Is that NETBIOS also?

Steve: Same, it's the newer version of Microsoft's...

Leo: That's filesharing. You don't obviously want to do that on the Internet. And that was a real big flaw many moons ago.

Steve: And that's why, that's what induced me to create ShieldsUP! in the first place was that everybody had their C: drives exposed to the Internet.

Leo: You'd go to a hotel, and you'd see all the hard drives available to you.

Steve: Oh, my goodness, yes. So what ISPs could do and must do immediately is block that port.

Leo: And there's no other use for that.

Steve: None, no.

Leo: So there'd be no side effect.

Steve: No collision. There is no reason for any ISP subscriber to ever receive incoming traffic on port 1900. None. The router won't open it for incoming use because it's already in use by Universal Plug and Play. So it's reserved for that. And there is no need for it out on the Internet. And in fact, anyone who is interested, if you run a scanner on the 'Net, watch for UDP port 1900. I bet we're going to start seeing this because the other thing that happened is that all the hackers know this now, too, since yesterday. The Metasploit Framework is open source. HD Moore has published every detail...

Leo: Open source [laughing].

Steve: Yes, every detail about how to do this, how to take over 81 million routers anonymously.

Leo: And he's not doing that maliciously. He's doing that as a public service.

Steve: In full disclosure, yeah. Before he went public he worked with all the various CERT organizations. They contacted all the router vendors.

Leo: Were promptly ignored.

Steve: Yeah. And this is just the problem, is in the same way that there's still Code Red and Nimda out there scanning around, looking for something vulnerable from the Flintstones era, similarly there are routers that are never going to get fixed. There's 81 million of them now, and it's not clear how many of them are going to get fixed. And the other problem is many of these do not have an ability, they may in fact disable Universal Plug and Play on the inside but leave it enabled on the outside.

Leo: [Laughing] That's really bad.

Steve: Yeah.

Leo: So disabling UPnP isn't going to work on all routers.

Steve: It's not clear. It's not clear that it shuts down the demon. It may just...

Leo: This is worse than the WPS flaw.

Steve: Oh, yeah, this is way...

Leo: This is horrible.

Steve: This is way worse. Yeah, WPS you have to be nearby. Here...

Leo: Right, anywhere. Could be Russia.

Steve: They're not going to sleep in China and Russia right now because they're going to have too much fun with this. Unbelievable.

Leo: So mitigation really has to come from the ISP for most people. You certainly should turn off UPnP in your router. But I do that routinely. It may not be enough.

Steve: And again, it's why a priority for me is to add detection to ShieldsUP!. When I disconnect from you in an hour from now, Leo, that's what I'm going to start on.

Leo: That's easy because you'll just test for port 1900; right?

Steve: Well, yes. But right now ShieldsUP! is a TCP test.

Leo: Oh, you need a UDP.

Steve: Yeah, well, I have all that technology. I've got DNS servers and all kinds of other stuff. I mean, it's not, I mean, it'll take me a couple days to nail it down and get it right and do the UI and all that. But I will.

Leo: Now, I use ShieldsUP! right now it will tell me if port 1900 is open; right?

Steve: Only if TCP 1400 is open.

Leo: Oh, no, I get it, I get it.

Steve: And, now, if you knew which TCP port your UDP service was on, then it would tell you that that was open. But you'd have to do a 65535 port scan.

Leo: It's random.

Steve: Yes, it's random. So that's crazy.

Leo: And I'm sure, but I'm going to ask, that DD-WRT, Tomato, and the other custom firmwares for routers don't have this problem, I would guess.

Steve: There are three spreadsheets that I encountered. And I don't know how I found them. I don't think they were links. They may have been links in the report, or they may have been on the Rapid7 website. But they have published Google Docs spreadsheets showing all the router makes and manufacturers. It's like 536 pages long, though. I mean, it's crazy. It's an analysis of what they found and the make and model and version number and so forth of all the vulnerable routers. Fundamentally, I mean, the only way to know is to get a test from outside.

Now, he mentions in the executive summary there's this ScanNow thing. I did create a bit.ly link for it before I tried it. And then I decided not to share it on Twitter because it's a disaster. I mean, hackers write hacking code. It's not production code. It's 6MB. Apparently it requires Java. Mine, when I tried to run it, it scanned a little bit, then just died. And who knows why. So maybe it's useful. It's bit.ly/upnpscan. And that will take you to the page to download Rapid7's free scanning tool. And that's a Windows-only, by the way. There are instructions for how Mac and Linux users could use Metasploit Framework to do that.

Leo: Well, that's the other thing. You could try to exploit your own network.

Steve: Well, yes.

Leo: Through pentesting.

Steve: It would be interesting to find out which devices in your own network are vulnerable. And of course the problem is it would be nice if it were cross-platform, and at this point it isn't. But it's also...

Leo: Well, let's just stop Security Now! right now. Go to work. We need this. Get this thing done. Fix ShieldsUP!. We need a UDP port scan now, stat, pronto.

Steve: I know.

Leo: What do you think, it'll take you a day? Two? Because, I mean, really the clock is ticking on this; right?

Steve: Oh, you're not even going to be around for your show this weekend.

Leo: No.

Steve: Okay. Well...

Leo: But I'll make sure, if you send Chad the information, we'll plug it like crazy.

Steve: Yeah. I will let you guys know as soon as I have it up. And, yeah, it's...

Leo: And we'll tweet it, and I'll, I mean, I'll be at least on email, so I'll be able to tweet it and everything.

Steve: Yeah.

Leo: Rapid7 has a UPnP tester, Zenserver [ph] says in our chatroom. Have you seen that one?

Steve: That's the one.

Leo: That's the one. And it's not good.

Steve: It's, well, use it at your own risk. It didn't - it's huge and awkward. It asks for your email address, so they want to upsell you. And it requires Java. Apparently someone tweeted that he had to install Java in order to run it. I've got my Java well under control

here. So it's not a great solution.

Leo: Well, Steve's will be the best. Just get it...

Steve: Oh, and that's the other problem is it's an internal scanner. What we want is an external scanner.

Leo: Which is what ShieldsUP! is. Your servers probe our network and then report back on what the results of the probe have been.

Steve: Yup. I'll have it up in a couple days.

Leo: Okay, good. Thank you, Steve. And of course it will be written in assembler.

Steve: Yeah.

Leo: Not that it matters because it's server-side. Although don't you - you used to, and I think you still do have a download that I can run also; right? There's like an additional little...

Steve: There was something I had, I don't even remember the name of it now.

Leo: I'm trying to remember myself.

Steve: It was some little agent thing.

Leo: Yeah, the agent, yeah. You don't do that anymore?

Steve: No. There was just no need for it.

Leo: No need for it.

Steve: I think I did a rewrite of the whole technology, and I obsoleted that.

Leo: And what we're trying to do anyway is to see what it looks like from the outside world. That's what you report back.

Steve: That's the whole point, yes.

Leo: Well, you should still disable UPnP on your router, if it's enabled. I don't think anybody who listens to this show has it enabled.

Steve: Right. I would say, if you don't know you need it, disable it, and we'll hope for the best. And I will very soon have a proactive test that anyone can just go to GRC, I'll make it easy to find, and it'll just send that packet to you and let you know if your router responded or not. And if it did, that's not good.

Leo: And to reiterate, does DD-WRT or Tomato, do you know, or have you - it's a lot of...

Steve: I don't know, but I have to imagine that at least the latest versions of them are fixed. So I would say, across the board, get the latest firmware you can, for whatever router you're running. And that's another point that I wanted to make. I'm glad you brought it up again, Leo. Many existing routers who are still using the standard manufacturer firmware may have this problem. But switching to, as you said, Tomato or DD-WRT firmware, if it's available and compatible with your router hardware, that also solves the problem, presumably.

Leo: And I would guess, but I'm going to have to check this database, that Apple routers are not prone to this. They don't actually have UPnP. They use NAT-PMP instead.

Steve: And remember, it's not the UPnP per se that's the problem. It's the external exposure, which is a complete mistake.

Leo: Right. But in theory, if you didn't have UPnP in your router for whatever reason, it's unlikely that there would be an external exposure, like they would put in a feature that they don't give you access to.

Steve: Yeah.

Leo: Yeah, okay, wow. This is terrifle. Terrible. Terrifying and terrible.

Steve: Yeah, well. I didn't need anything more to do, but I've got to do this.

Leo: Oh, I'm so sorry. But, I mean, golly, get to work.

Steve: Oh, it'll be good.

Leo: I'm sorry. Get to work. Oh, my goodness. Well, it's frustrating to not be able to

test for it; right? You don't know.

Steve: Yeah.

Leo: Amazing. Continue on.

Steve: So I'm just wondering whether - the ScanNow utility does allow you to give it a starting and an ending IP. And if you knew a friend's IP, you could run ScanNow with that single IP as the starting and ending IP. And they mention in the documentation that that causes it just to scan the one IP. And that would route the packet out through your router across the Internet to their router and back. So you could use, if you had a partner, you could use ScanNow...

Leo: You mean the built-in Windows tool of Scan Now.

Steve: No, no, no, sorry, no. This thing they call ScanNow...

Leo: This Rapid7 thing, yeah.

Steve: The Rapid7 thing that you get at bit.ly/upnpscan. If you set that up outside your network, that is to say on someone else's network, and knew what your public IP was, you could say, okay, John, scan me. And then that...

Leo: Could you...

Steve: I think that would work.

Leo: Could you use something like Nmap or tools like that? Nmap's used to do...

Steve: Well, you could certainly use Metasploit. You could use Metasploit.

Leo: Use Metasploit, yeah.

Steve: But the idea is it's got to be on the outside. So you need...

Leo: I get it, I get it.

Steve: You need someone else...

Leo: You need a friend to do it.

Steve: ...a friend to send it in your direction from the outside. And of course ShieldsUP! will do it in a day or two.

Leo: This is traditional pentesting. This is what people do. They look for vulnerabilities. And now there's another one. We should offer a service. I'll scan you. I'll scan you if you'll scan me.

Steve: Well, I have that. It's called ShieldsUP!. It just doesn't do this one yet. But it will.

Leo: And again, we're not recommending Rapid7. It crashed on Steve's computer. I'm seeing others in the chatroom say it crashed for them. You have to use Java in the browser to make it work.

Steve: Yes. It's a piece of junk, actually.

Leo: Yeah. So we're not saying that's the solution. It just seems to be the one available right now. You know what, I'm going to put the chatroom to work. Maybe by the end of the show they'll have another choice.

Steve: Okay. So, many people asked about, want the full security dump on Mega, Kim Dotcom's new offering. And because of all this, I didn't get to it again. But it will be the topic for next week's Security Now! podcast.

Leo: We can do triage. Pick the biggest vulnerability first.

Steve: Yeah. We have - we're sorting these.

Leo: So next week, good. Look forward to it.

Steve: Yes. Okay. A little miscellaneous kind of fun stuff. I don't know why I got this newsletter. I remember I subscribed to it years ago. In fact, the email address it came in on demonstrates how long ago it was. It's the Journal of Physical Security. And we have a listener who's at the Argonne National Laboratory or something because I'm dimly remembering this. But the new issue of the Journal of Physical Security came out. And this is sort of interesting. I just thought I'd give our listeners a heads-up. It's jps.anl.gov. So JPS of course is Journal of Physical Security. So jps.anl.gov. And of course ANL is Argonne National Laboratory. And just to give you a quick little table of contents, they call themselves "The first scholarly, peer-review journal devoted to physical security R&D."

The first paper, pages 19, is "The Japan Earthquake and the Tsunami: Their Implications

for the U.S." Paper 2, "Lock Opening by Bumping: Physical Analysis and Secure Lock Designs," pages 1021. The third paper, "How to Choose and Use Seals," as in, like, I don't know, King Arthur and wax or something. And that's pages - so how to choose and use.

Leo: I want to subscribe to this. This sounds like the best journal ever.

Steve: It's wonderful, yes.

Leo: Is it email only, or is it a print journal? I want it.

Steve: Just go over there, yeah. It's neat. So that's pages 2231.

Leo: This is great. This is the best magazine I've ever heard of.

Steve: Paper 4 is "Election Security: Don't start with fraud investigations, start with security investigations." The fifth one is a "Viewpoint Paper: Common Election Security Myths," pages 4345. And then, get this one, Paper 6, by D.B. Chang and C.S. Young: "Comparison of Window Stresses from Explosions and Projectiles."

Leo: It's the government publication.

Steve: Yeah.

Leo: From the Argonne National Labs. I mean, this is not hackers.

Steve: No. This is...

Leo: This is our tax dollars at work, ladies and gentlemen: "The first scholarly peer-review journal devoted to physical security R&D." I love this. You could subscribe free of charge. Argonne National Laboratory. By the way, just as long as I've interrupted you, somebody in the chatroom has come up with this Android program called Port Scanner, lets you scan ports on a remote host via its IP or domain name. Now...

Steve: But if it's only TCP...

Leo: It's got to be UDP; right? So we've got to look at this and see.

Steve: In which case you just want 1900.

Leo: So if it did UDP, you would just scan port 1900 UDP.

Steve: Oh, and the other problem is you need to actually send a deliberately formed discovery packet.

Leo: It's not just a ping.

Steve: Right.

Leo: Ah.

Steve: Yes.

Leo: You have to wake it up and say...

Steve: Yeah, I'll be busy, Leo, for the next couple days.

Leo: Yeah, interesting. All right. So this isn't enough. Jps.anl.gov, I love that.

Steve: Isn't that wonderful?

Leo: That's the best magazine I ever heard of. They don't come out very often.

Steve: No. In fact, when I got it in my mail...

Leo: Not even yearly.

Steve: Yeah, it's been a long time.

Leo: It's only six volumes in eight years.

Steve: Yeah. But they're good.

Leo: They're good. They're juicy. You can download them as a PDF. I'm sending these to my Kindle right now.

Steve: "Comparison of window stresses from explosions and projectiles."

Leo: Freaking awesome.

Steve: Ever wanted to know.

Leo: Who has that job?

Steve: And lock bumping is really interesting, the fact that pins in a lock can be caused - just by knocking on the lock, you're able to use some vibrations in order to migrate the pins to their splice points and then end up being able to turn the tumbler. So, yeah.

Leo: Apparently there's a whole team at the Argonne National Laboratory that does this stuff.

Steve: A bunch of monkeys.

Leo: Now, that's what tax dollars should go to. This is wild. Okay. And it's available via PDF for free. Good.

Steve: Okay. So I'm going to skip the PDK update. We'll do that next week because we're running short of time. I did get a nice tweet from a listener, Dan Fox in Illinois, who has a how-to on running SpinRite in a VirtualBox VM.

Leo: Oh, good.

Steve: On Windows 7, Mac, or Linux. And so we had talked about there was a listener, a show listener who talked about booting an external Mac drive, thus the Mac was running on that drive, not on the Mac's internal drive, and then running SpinRite on a VirtualBox VM in order to run it on the Mac's native drive, which is possible. He referred to a bunch of command line options or necessity, and Dan has it all laid out. So it's another bit.ly link that's easy to remember: bit.ly/srvm, as in SpinRite Virtual Machine, SRVM. And so it explains how to do that. And he said, "P.S.: I met Leo in early June of last year after a recording of TWiT and showed him a prototype of my Masters Project called TrustAuth on my MacBook Pro."

Leo: Yes.

Steve: "I'm sure he'll remember it."

Leo: I do.

Steve: "It's an authentication system using public key crypto instead of passwords for

web logins."

Leo: I didn't understand it at all.

Steve: Another smart listener.

Leo: But he's smart.

Steve: And then Michael Vail in Throop, Pennsylvania says, "Yes, Virginia" - this is very short - "SpinRite is for real. I'm a proud owner of SpinRite 6 and finally got to use it. My father-in-law's computer died hard and of course had no backup despite having an irreplaceable photo collection. The drive was in such bad shape that it took SpinRite almost exactly three weeks. But SpinRite was able to recover 500 photos, including ones of people who had since passed. Thank you for a great product and for all the sci-fi recommendations. I bought ALL" - he has in caps - "of Michael McCollum's after reading 'Euclid's Wall.' You and Leo have great taste." And I have to say, Michael McCollum's stuff, if you liked "Euclid's Wall," you're going to love the other stuff. They're great and reasonably priced.

Leo: So you know what, in the interest of time, let's go right to the questions; right? You ready?

Steve: Yeah.

Leo: You feel good? You've got your thinking cap on? Listener-driven potpourri No. 160, starting with Mike Nash in Salt Lake City, who asks the question we all ask: Am I really secure? Steve, you continue to receive so many questions about SSL sessions, certificates, and concerns over whether the connection is truly secure or not and whether the hosting network might be intercepting and spoofing the remote host's SSL certificate. So what if you were going to create a single web page that users could visit any time they're worried to show whether they're really secure or not? In other words, if you go to GRC, and it's maybe an SSL site, and you would say, hey, you should be seeing this. What are you seeing? Is that possible? Could you do it? Thanks for the great information and laughs along the way.

Steve: So this is the note that I encountered two weeks ago, where I said, oh, my god.

Leo: What a good idea.

Steve: I can do that.

Leo: Did you do it?

Steve: And it's what I've been working on ever since. I brought it online Friday night. And it stayed up for about an hour or two and then died. Now, the bad, well, and unfortunately it took our eCommerce system along with it because...

Leo: Oh, no.

Steve: [Laughing] And in the morning, on Saturday morning I got up, and it's like, uh-oh, what's not working here? And it's not Peter Gutmann's fault. You remember Peter. We had him on the show in 2006 [SN-077 January 2007]. He's a fantastic security cryptographer, cryptologist, researcher in New Zealand. We had him on the show in '06 because he fully examined Windows Vista's DRM content protection model, and he wrote this beautiful paper called "The Longest Suicide Note in History" ["A Cost Analysis of Windows Vista Content Protection"].

Leo: By the way, Microsoft's still alive. But okay.

Steve: Yeah. But Vista didn't last very long.

Leo: Vista didn't last.

Steve: So he was certainly right about that. Anyway, I use a library called "cryptlib," which is Peter's. It's very nice and very clean. And it turns out that there were some things I didn't understand about the way it was caching sessions because in doing this test - in fact, Leo, you could probably grab this PNG, if you want to take a look at it. It's just GRC.com/ssl_fingerprints_technology.png.

Leo: Dot htm.

Steve: No, .png. It's a picture I made of the web page when it was working. Yup, there it is.

Leo: Yeah. It's the most I can blow it up. It's so long, I see why, it's formatted at - so it's so long, I can zoom in a little bit, but you can't really read it. Wow, this is cool.

Steve: Anyway, so this...

Leo: So why did it crash? What happened?

Steve: Oh, it's that I announced it into my newsgroup crowd.

Leo: Oh, you told people about it.

Steve: Yeah, and they all began to play with it.

Leo: Oh, dear.

Steve: And so it was reissuing sessions for sessions it had already secured. I was trying to obtain the security certificate from them. But, being smart, Peter reestablishes existing sessions using the standard SSL session resumption technology, so he doesn't get a security certificate every time. I didn't understand that. He and I have a dialogue open now, and I have some other questions and confusions.

Anyway, what this will do when it's up, it's not up now, is exactly what Mike was asking for. I was hoping that I could make it completely automatic, that you could bring up a web page that would get the fingerprint that it has received from the server and compare it to one that it has received from me and see if they're different. Unfortunately, JavaScript does not have permission, for reasons I don't understand because I don't see it being any kind of a security problem, but there is not in the current JavaScript API a means of obtaining security information about the session. Privileged JavaScript, running as an applet, for example, in Firefox, can get that. But that's an applet, and I didn't want to do an applet. And then it would only be Firefox and not Chrome and IE and Safari and Opera and so forth.

So what this does is it's still pretty simple. If you're anywhere where you think someone may be spying on you in your company, in your educational institution, anywhere, you can simply go to this page, it'll be called "Fingerprints," and just bring it up. And it will show you a block of the most popular websites' fingerprints and also GRC.com, where you happen to be. You then check the fingerprint that your browser sees by right-clicking on the page and saying "View Certificate," and all the certificates show you the fingerprint. The key of this is, it's not possible to forge a fingerprint. That is unforgeable.

So an SSL proxy, as we've discussed often, can forge the identity of a remote site if your computer trusts it. And it would trust it if it has received a certificate authority cert from your company, from your university or whatever, in which case it can do on-the-fly forgeries. But it's not possible for it to duplicate the actual fingerprint of the remote server because the fingerprint is everything. It, for example, includes the server's public key. Well, it can't use the same public key because that only matches the private key. And no one knows what the remote server's private key is.

So the point is I will, probably next week, announce that this system is up and running. And it'll be the first new service that GRC has offered in a long time. Or at least I thought it was. Now it looks like we're going to have the UPnP scanner also. But it'll be very cool 'cause it'll allow people to just quickly check to see if anyone might be running a man-in-the-middle attack on them, through any form whatsoever. So it'll be very cool. And thank you, Mike, for the idea. I replied to him immediately when I saw his note and thanked him and told him that I was going to get on it and hopefully announce it in two weeks. Well...

Leo: You've got too many things to do now.

Steve: Yeah, yeah.

Leo: That one's going to wait, as well. Question 2 comes to us from New Jersey. PJ says: In the last couple of Q&A podcasts the topic of corporate root certificates has come up in relation to a company being able to intercept and view all SSL traffic to its employees. Just what we were talking about in the last question.

Steve: Yup.

Leo: Is it true that IE uses the root CA on Windows, but Chrome and Firefox use their own CA? Or their own list of CAs, I guess.

Steve: Right.

Leo: If yes, does that mean if I install a browser after a computer's been issued to me, then my company won't be able to decrypt SSL traffic to this alternate browser? I guess this would only work if the web proxy or gateway does not require the root certificate before allowing you to proceed. Is that true?

Steve: Well, it's not quite correct. What very likely happens is that, when your computer is given essentially a fraudulent certificate authority certificate, it is also given a client certificate. We've not talked about that very much because our typical use of SSL is only to authenticate the remote server. We're anonymous, so we're making an anonymous connection from our end to a non-anonymous authenticated server on the other. But there is in the SSL protocol the provision for the server asking you for a certificate. Well, we don't normally have one. I mean, most of us don't. Except that, in a corporate setting, this spying SSL proxy, this interceptor, it would require a client certificate because it also wants to identify who's making this request. So it's going to lock onto you and ask you for a client certificate.

So what that means is, if you had a computer that came in, you brought your own computer in from the outside and tried to connect it to the corporate network, it would probably fail. It would say, I'm sorry, this computer is not authorized to use our network. What that really - that's code for we asked for its certificate, and it didn't have one, so go talk to IT. Well, of course, if you talk to IT, they'll give you not only a certificate, but a fraudulent certificate authority, and they'll say, oh, now you'll be able to use your computer on our network, which opens it up to spying. So that's likely what happens.

Now, as for certificate authorities, it is the case that Firefox, from day one, implemented their own - they call it NSS is the - it might stand for Netscape Security Service or something. That's the library, their own SSL library. It contains all of the Mozilla security infrastructure, and it does have an entire populated certificate authority store. So that's true for Firefox. Obviously IE is going to use Windows' native store.

And I wasn't sure about Google because from the stories we've talked about in the past we've heard them being very proactive with certificates. Yet when I was doing the work for the fingerprinting system last week, I noticed that the dialogues that came up in Chrome looked like Windows, which led me to believe that Chrome was in fact using Windows' security model. So I did some digging, and I found Google's - they call it the

"Root Security Policy." And they said, "Google Chrome attempts to use the root certificate store of the underlying operating system to determine whether an SSL certificate presented by a site is indeed trustworthy, with a few exceptions."

And then they say, "In order for Chrome to be able to trust a root certificate, it must either be included by the underlying operating system or explicitly added by users," which of course we're talking about here in the interception concern.

"If you are a root CA, the following contacts should be used." And then they list how to get a hold of Microsoft, Apple, the Linux people, and Android. If you were a certificate authority who wasn't - who was, like, just launching, you would need your root certificate moved into all of these places. So they're saying don't call us. We're using the OS's underlying technology, not our own. So that's the answer. Everybody uses the OSes with the single exception of Firefox.

Leo: And that's probably as it should be. I mean, you don't want a whole bunch of...

Steve: I think so.

Leo: ...cert stores.

Steve: No, they'd be out of sync and messed up and, yeah.

Leo: And, well, the security issues would be significant, as well.

Steve: Yes.

Leo: And we do install, I have installed my own certificate, just once, a certificate authority. And I think it was LastPass. I'm trying to remember. I had to run an app for it to authenticate. I'm trying to remember. But it came with a little plugin that you'd run, and then it would use their own - which obviously is a scary thing to do. But in this case...

Steve: I was just going to say, that's scary.

Leo: I think it was LastPass, but I'm going to have to try to remember. Anyway...

Steve: Don't know why it would have been unless there was a standalone app or something.

Leo: Yeah. It had to do with using its - I don't use it anymore. I use Google Authenticator now. But remember you used to have a standalone app that you would run for LastPass to do second-factor authentication? So you'd give it your password,

and then it would say, if this isn't a trusted machine, it would say, all right, now you have to run it, and you can put it on a USB key. But in order for it to work you'd have to install the certificate authority.

Steve: Ooooooooooooooh [laughing sardonically].

Leo: You don't remember that, huh?

Steve: No.

Leo: Yeah. Well, anyway, I now use Google Authenticator. Although probably that certificate is still sitting...

Steve: Still there.

Leo: Yeah, it was Sesame. Thank you, Serge. Sesame was the name of the program, the LastPass program you'd run. And it would give you a second-factor code. But it wouldn't work unless you had their certificate on there.

Steve: Well, okay. It might not have been a certificate authority. It might have been...

Leo: It might have been just a certificate. You're right. You're right. It wasn't a CA. What am I thinking? Of course it was just a certificate. Which happens all the time. I install certificates. Whenever I use PGP you install certificates.

Steve: Exactly. Exactly. To identify your system to it.

Leo: Right, yeah, doy. Stupid of me. Never mind. Forget I said a thing. Back to the questions. Lee in London, in England...

Steve: Let's skip that one because we're going to have to skip a few. We're running out of time. And we've spent lots of time...

Leo: Yeah, sorry, and we started late because I had to give a tour. My fault.

Steve: But 4 is a good one.

Leo: Yeah. Bill Welker in Colorado Springs, Colorado: Steve can you refer me to a source to describe and explain the physical handling of packets in Internet data

traffic? I understand the definition of a packet and its contents in terms of the protocol. I cannot comprehend how a packet is physically sent on a wire. Since a wire can only have current on it or not, i.e., as a potential between two endpoints, that is, voltage, I'm having difficulty understanding how a wire can handle so much data in such a short time and carry that data so that discrete packets are maintained. Am I making sense? I've searched and searched on - it's amazing how fast electrons are.

I've searched and searched on this topic and not found any description of the actual physical handling of a packet on a physical wire. I'm not suggesting it's a topic for Security Now!, I just want to know. If you could refer me to a technical description, I'd be grateful. I just always assumed it's switching; right? It's on and off? It's ones and zeroes?

Steve: Well, yes. But one of the cool things, one of the cool aspects of the technology, he talks to about how it's able to go so fast, and how we're able to have wires draped around the floor, and they're running gigabit speeds.

Leo: It's pretty amazing, isn't it.

Steve: That's just crazy, yes.

Leo: We just take it for granted, frankly, because everything else in the computer industry is magic.

Steve: Yeah, it's like, why not?

Leo: It's all magic.

Steve: Yeah. It's funny, too, because that was - I got into a discussion with Jenny about "Euclid's Wall" where the theme of the book, as we've described, is that a truly massive set of earthquakes shakes civilization to the ground, I mean, and regresses us to sort of a pre-technology mode. And Jenny sort of, who's not a technologist, said, oh, well, that'll take, what, a week to get back? I said, oh, no, no, no, no. I said, where do chips come from? Well, you know, the chip place. Okay, but...

Leo: The chip store.

Steve: But the more you think about how dependent we are on...

Leo: It's a fine thread we hang from.

Steve: Oh, my god, on a hierarchy. And, for example, when I said the same thing to

Mark Thompson, he said, oh, yeah, we'd never recover. It's just...

[Talking simultaneously]

Steve: ...so true.

Leo: One electromagnetic pulse, one air burst, we're gone. You know, that's one of the - I've talked about this before. We were talking about Jerry Pournelle and Larry Niven, and "Lucifer's Hammer" is his story of how an asteroid hits the Earth.

Steve: Yup, meteor.

Leo: Meteor hits Earth and sends us back to the Stone Age. But what he says, and I think it's very germane, is early on one of his characters says, you know, we all use this stuff. How many people actually know how it's made, how to make it, how to fix it, or, if we were to lose the technology, how to bring it back? And that's another problem. We take it for granted, but do we know how to make it? You do. Thank god. But how many of the rest of us who use this, we just - thank you magic, magic person.

Steve: Yeah, it is phenomenal how much we build on. And again, I know a lot. I could start making batteries. But...

Leo: But you couldn't build a cyclotron.

Steve: No.

Leo: Or even maybe an automobile.

Steve: It would be rather low down on my list, but I could build it.

Leo: Could you build a car? You could build your own combustion engine from scratch?

Steve: Oh, yeah. Yeah.

Leo: Really.

Steve: Oh, yeah. I took mine apart completely on my - I had a little Fiat.

Leo: I know how they work in principle. But the thought that I could actually make

one that would work...

Steve: Oh, yeah, I mean, it would be stinky and noisy and make a lot of fumes.

Leo: I'm coming to your place in the case of atomic war, then.

Steve: The EPA would just roll over in their grave. But, you know...

Leo: But would you know how to refine petroleum so you could use it?

Steve: Ah, now, that's a problem. See? You're right.

Leo: But this is life. We've always been interdependent on one another. And it's just - we forget. Conveniently. Because if we really thought about it, it'd be terrifying.

Steve: Oh, we'd just stay in bed.

Leo: Question 5 from - I love his Twitter handle. Are you done? I'm sorry, go ahead.

Steve: We never really answered Bill's question.

Leo: Oh, we didn't answer the question, yeah.

Steve: We just said...

Leo: How do packets...

[Talking simultaneously]

Steve: It's bits. Go to the next question.

Leo: It's bits, on and off, on and off, real fast. Is it really switching?

Steve: They're speedy little electrons, as you said.

Leo: Is it modulation? Or is it actually on and off?

Steve: They get up to speed. The concept I wanted to actually convey is the notion of differential electricity. That is, it's not one wire, it's two. And that's the key. It's a so-called "twisted pair." And people have probably heard the jargon, "twisted pair." But it's really crucial. And the reason is that our environment is full of interference, all kinds of noise. And, I mean, electrical noise, radio noise, microwaves in the kitchen, I mean, just all kinds of interference. How do these wires not pick that up? Well, they actually do. The trick is that two wires are intimately physically close to each other and truly wound around each other so that they pick up the same noise. They both pick up the same thing. So on one end the sender, the transmitter pulls one wire high and the other one low. And that signal moves through the wire and gets to the other end.

Well, any interference which occurs along the way happens to both of them. So the only thing the receiver cares about is the difference in the voltage at the receiving end. It sees one is higher than the other. Doesn't matter how much. It matters in which direction because that's the data. But it doesn't matter whether they're both at 50 volts and 49 volts.

Leo: Amplitude doesn't matter, just their relative amplitude.

Steve: Exactly. The so-called differential. And the term for the noise is called "common mode." The common mode is that voltage or noise that they share is completely ignored, and it's only the difference. And so the transmitter, it moves the - when it wants to change it, the one that was at a one or a high level, it pulls low, and simultaneously pulls the one that was low high. And so out at the other end they flip positions. And then to that you add - there are many, many protocols that have been developed by clever people over time. The one that was used by 10Base-T, which is the most simple to understand, is a technology called "Manchester coding." The idea is that you have moments in time which represent bits of data. And with Manchester code the two wires are - and so we'll subtract one from the other and call it a one or a zero. So we've dealt with the differential aspect. And so what's coming out is a one or a zero. If it's a one for the first half of the bit cell, then we call that a one of data. And if it's a one for the second half, we call that a zero of data. Now, the reason that's important, the reason it was not just a one or a zero, is say that we wanted to transmit 50 zeroes. Well, that would be nothing happening for a long time. Well, is it 50 or 51 or 49?

So the problem is, and we've talked about this relative to hard disk drives, they have the same kind of problem called Run Length Limited, RLL encoding, because as time is passing and nothing happening, you need to know how much nothing happened, and you need to know exactly how much nothing happened. So instead they use what are called "self-clocking codes," Manchester code is one, where you always have something happen. It's just when it happens. Does it happen early, or does it happen late? If it happens early, it's a one. If it happens late, it's a zero. And that way, independent of the transmitter, the receiver is able to reconstruct the same data that the transmitter sent at the receiving end. And the encoding technologies have gone from that 10Base-T simplicity to stuff that, I mean, just boggles your mind as they go from 10 million bits per second to 10 billion bits per second. A thousand times more bits over the wire.

Leo: Reliably. Reliably.

Steve: Yes. Yes.

Leo: So it is kind of related to the analog way it was done then.

Steve: Yes.

Leo: I mean, analog is modulated; right?

Steve: Yes. And so here there's no carrier that we modulate because that would waste bandwidth. So we're using absolutely all of the wire's ability to reflect the change in voltage in time as possible. And remember that there is error correction. So packets have checksums. If something happens where it doesn't work, we say, send that again. And so we do. So all of this stuff is working at some low level of unreliability where it mostly works. Sounds very much like the way hard drives are these days. Mostly they work. And if they don't, we just ask for it again.

And of course that's actually one of the differences with magnetic storage versus data in transit is data in transit you can always ask for it again. If it's been written to a hard drive, and you can't read it, then there's no one to ask for. Whoever wrote it's long gone a week ago. Who knows how long ago that got written on the drive. So there's no one to ask. Thus the reason we actually - we pad the data in a hard drive sector with error correction code so that we can fix it ourselves if we're unable to read it easily. And Leo, with that, we wrap another fabulous podcast.

Leo: We'll save the rest for later. Thank you for your - I have one question, though. I do have a question of my own.

Steve: Okay.

Leo: Where is your PD - your top PDP is gone.

Steve: Ah, you didn't notice that. Some listener noticed that; right?

Leo: Oh, of course. Immediately. The chatroom has been talking about nothing else for the last hour and a half. Normally, for those listening, Steve has three blinking lights behind him that duplicate the face - they're not actual PDP-8s, but they're replica PDP-8s that are actually doing work. And there's normally three of them, and one of them's missing. So what's the story?

Steve: Yes. The guy who did these, Bob...

Leo: He wants them back.

Steve: I'm blanking on his last name. No.

Leo: Bob called. He wanted his PDP-8 back.

Steve: He found 30 more of the microprocessor chips. They were made by Harris. It's a 6120 is the number. It is a PDP-8. And so he's seriously considering bringing the project...

Leo: Oh, good.

Steve: Yes, back to life.

Leo: This was a limited supply, and he does the silkscreen of the face and everything.

Steve: Oh, and, I mean, it's so gorgeous.

Leo: He doesn't even have the original silkscreen? He's using yours as a template?

Steve: Oh, no, no, no. What happened is he loves my frames. I found this fabulous frame that is a two-inch-deep frame that the whole thing fits behind. And anyway...

Leo: Aren't I lucky to have Steve as a friend? Aren't we all lucky that we know a guy like Steve Gibson? Unbelievable.

Steve: I took it down in order to go over to Aaron Brothers Art Mart, where I got the frames...

Leo: To get it made.

Steve: Yes. And what I found on the door was "Veggie Grill Coming Summer of 2013."

Leo: Oh, no. Bye bye, Aaron Brothers. Hello, Veggie Grill. We have an Aaron Brothers up here.

Steve: And, well, and Bob's got one near him. It turns out that there's all kinds of serial numbers that I discovered that were stenciled on the back of the frame. So I'm going to take photos of it. It's the reason it's not already back up there. I'm going to take photos of it, send them to him, and he'll be able to track it down. And he's not yet sure - I'm sort of pre-announcing something that I shouldn't.

Leo: You know he could sell every single unit to the Security Now! audience, like that.

Steve: They're just so fabulous. He's, for example, he's wondering should he just do them as ready-to-go, turnkey, finished assembled units? He's asked me for permission to embed the code that I wrote, my little utilities, the blinking lights program and the little puzzle program that I wrote, embed those in the ROM so they'll come built in with that, and of course I said, oh, absolutely. So there may be a project of some sort that would just use up - he feels badly that he's got these 30 beautiful PDP-8 chips, and he could turn them into more of those. So, and they're just works of art. It's just a beautiful little machine.

Leo: But on the bright side, you've got a Veggie Brothers store opening near.

Steve: A Veggie Grill, yeah. Jenny, who is a vegetarian, loves it.

Leo: She's happy about that. Steve Gibson is at GRC.com. That's where you need to go right now. Obviously there's so many great things. But in a couple of days...

Steve: You will be going there soon, yes.

Leo: You'll be definitely wanting to go to ShieldsUP! And we will announce it. I'll tweet it. I'll make sure - Tom Merritt and Father Robert Ballecer are hosting TWiT on Sunday. Robert, of course, he's probably in the chatroom right now. He's our enterprise tech guy, and he will absolutely want to talk about this. So just send him a note. You can send it to padre@twit.tv, if you want, Steve, and you can include him, and I'll make sure he's up to date on that. And we will let you know. And of course, once you're there you want to buy ShieldsUP!, I mean SpinRite, because that's the world's best hard drive and maintenance utility. You might want to check out the amazingly low bandwidth 16Kb version of the show that Steve makes available and the transcripts, which are, frankly, lower bandwidth still, at GRC.com. Also that's where future questions can be asked: GRC.com/feedback. And someday, perhaps, we'll have your SSL tester there, as well.

Steve: Yeah. I'm going to suspend it for the UDP vulnerability scan in ShieldsUP!. That's got to take priority. So I'm going to do that. And I'm at a perfect point right now. I sent email to Peter Gutmann this morning, asking how he wanted to proceed on a couple of things that I'd found. And he's, like, in New Zealand, so lord knows what time it is there.

Leo: He's just getting up; right?

Steve: I think it's probably the middle of the night or something. So I will immediately add the functionality to GRC. I expect it'll take a couple days for me to - mostly just a matter of me coming up to speed on the details of the protocol. All of the infrastructure is in place and bulletproof. And, I mean, this is what it's made for. This is what I built GRC

originally to do was ShieldsUP!. So we'll have a new service soon.

Leo: Yeah, that's really good news. Thank you for doing that, Steve. This show is also available on our website. We make both high-quality audio and video available at TWiT.tv/sn, but you can also subscribe wherever podcasts are aggregated, like iTunes and the Zune Marketplace, places like that. And watch live if you want. We do the show live every Wednesday at 11:00 a.m. Pacific, 2:00 p.m. Eastern time, 1900 UTC on TWiT.tv. So tune in next week. We'll be back with more Security Now!. Don't forget our Security Now! YouTube channel, another place you could find it at YouTube.com slash - I think it's twitsecuritynow.

Steve: No, it's just securitynow. It works.

Leo: It's just securitynow. Oh, good, okay. We were able to get most of the shows. If you go to YouTube.com/twit you'll get there, and that's got links on the right to all of the shows.

Steve: And you'll be tempted by all the other goodies.

Leo: There's so many good shows now, 25 of them. That's why we basically had to create YouTube channels for individual shows because, if you subscribe to our YouTube channel, you get 25 show announcements, like four or five a day. That's too much. Thank you, Steve. We'll see you next week.

Steve: Thanks, Leo. It's a pleasure. And enjoy your red-eye to the game.

Leo: NOLA. NOLA. Go Niners.

Steve: Talk to you next week.