SECURITY NOW!

Transcript of Episode #571

Phishing & Filtering

Description: Leo and I catch up with the past week's security happenings, including LastPass vulnerabilities, new wireless keyboard headaches, deprecating SMS as a second authentication factor, obtaining Windows 10 for free after July, and a bit of errata and miscellany. Then we discuss RAID storage redundancy, the pervasive problem with website spoofing, and the power and application of multi-interface packet filtering.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-571.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-571-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here to explain all. We've got a bunch of security news as we head into DEF CON and Black Hat, the big hacker conferences coming up this week. He'll talk a little bit about that, answer some questions from the Twitter audience. We will also talk about phishing scams and filtering, particularly the Ubiquiti EdgeRouter X. And we were going to talk about RAID 5 and BeyondRAID, Drobo's RAID 5 interpretation, but I think we'll defer that for another day. So phishing and filtering, coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 571, recorded Tuesday, August 2nd, 2016: Phishing & Filtering.

It's time for Security Now!, the show where we cover all the latest security news and help keep you safe and help you understand what all this stuff means. And thank goodness we have the Explainer in Chief here. I noted, by the way, that at the Democratic Convention they stole your title, Explainer in Chief. Did you note that when that happened?

Steve Gibson: And I'm wondering, was that a title that Clinton, that Bill always had?

Leo: No. George W. Bush called himself the Decider in Chief. The Explainer in Chief, as far as I know, is you and no one else. So was it Hillary, or was it Chelsea who called Bill their Explainer in Chief?

Steve: Yeah.

Leo: I think referring mostly to his garrulous nature, more than his deep understanding of technical topics.

Steve: Yeah, yeah. He certainly wasn't giving his wife very good technical consult about how to handle her email.

Leo: No. Anyway, so here he is, the real Explainer in Chief, not Bill Clinton, but in fact Steve Gibson.

Steve: Indeed.

Leo: Hi, Steve.

Steve: So I titled this podcast RAID, Phishing, & Filtering for a couple reasons. There's just sort of three big topics I want to talk about. I'm thinking maybe I'll wait till we have another Drobo sponsorship before I go into RAID stuff because this was sort of relative to that. But there was some news about QR code hijacking, which is really about phishing. And we haven't talked about the phishing problem, but it's interestingly intractable. And so I wanted to spend some time talking about that.

And we've been talking a lot about this notion of multiport routing and packet filtering. But I realize I've also just sort of glossed over some of the details which I think a lot of our listeners would find really interesting. It is, it's down at that packet plumbing level.

So we have some interesting news this week. But then I thought I'd spend some time covering some just sort of relevant technology. Also, this is the calm before the storm. This happens every year around this time. It gets kind of quiet.

Leo: Yeah, yeah.

Steve: Now, we're also going to talk about LastPass vulnerabilities that happened last week. There are some new wireless keyboard headaches. SMS as used for second authentication factor is being formally deprecated because of the vulnerabilities that actually we anticipated months ago. Some news about obtaining Windows 10 for free after July. We have a little bit of miscellany and errata, and then our main topics.

But one of the reasons - this wireless keyboard content is interesting because it was a sneak peak into one of the presentations that will be happening at the end of this week. We are approaching - Thursday, Friday, Saturday, and Sunday - DEF CON 24. And so it was in, what, 1992, I think, was the first one, so 24 years ago. And it's been going strong ever since. And of course it's always at this time, in the late summer, we just are buried with really interesting new fun hacks to talk about which arise from that. So things kind of get quiet beforehand because all the hackers were saving up their goodies for presentation during DEF CON. And so we've got that.

The Picture of the Week someone sent me. I got a kick out of it. It's someone named Robb Stark, R-O-B-B Stark, and he tweeted it, @5stringplayer.

Leo: You think that's his real name?

Steve: Maybe. I mean, on the Internet you never know.

Leo: You never really know.

Steve: But what we do know is that this guy is crazy mad for SpinRite. And I don't know, like, what the setting is. You can kind of see behind, maybe it's an office facility. But he's got - this picture on the show notes is four separate screens, each running a copy of SpinRite on four different machines down below. And he's brought the power and SATA connections out the front so that four different drives are hanging, dangling from the SATA power and data connections. And if you look, you can see there's two little cartons over between the first and second and the second and third. The first one says "Awaiting SpinRite," and the second one, on a green big Post-it note, says "SpinRite Complete." So it's whole...

Leo: This is really amazing.

Steve: It's a SpinRite production system, essentially.

Leo: Wow, good for him.

Steve: I don't know any of the back story behind this, but I just got a kick out it.

Leo: He's got four PCs, and he's SpinRiting everything, man.

Steve: And he says, "Now, this is how you SpinRite." And I said, yeah, that's definitely the case.

Leo: That's great.

Steve: Kind of spooky. Kind of spooky.

Leo: That's really awesome. Yeah.

Steve: It's spooky, too, because if you look in the upper right corner of the screen, that's actually - see that larger rectangle in the upper half on the right side on each of those screens, that's the actual data in the drive. And it goes flashing by as you're watching it. And it's a little disheartening sometimes to see what is happening because it's the actual data on the drive as SpinRite is reading it.

Leo: Wow. Well, I hope this isn't an NSA facility because, if it is, he's probably leaked some critical passwords or something.

Steve: It's definitely a nice little setup.

Leo: That's cool. That's really cool.

Steve: So after the podcast last week came the news of some problems with LastPass. And it was a little muddied because there were two different researchers. Tavis got involved, our friend Tavis Ormandy, who took a look at LastPass. And he tweeted that he had found something and was in communication with LastPass, and they were working on it.

Then separately, for an unknown reason, I don't know if it was just to get some click traffic or what, but another researcher who had reported something a year ago, that has long since been resolved, chose to freshly post the URL. And so that got everybody in a concern, thinking that this was a new problem. And it wasn't a new problem. And in fact we talked about it a year ago.

So, but there was one thing that was interesting and new. What LastPass wrote in their blog in two parts, the first part said: "We want to share a quick update with the LastPass community about important fixes that we have made in response to two recent security reports." Now, that's a little odd because they said "...in response to two recent security reports. Our team worked directly with the security researchers to verify the reports made and issue a fix to LastPass users.

"The recent report only affects Firefox users. If you are a Firefox user running LastPass 4.0 or later, an update will be pushed via your browser with the fix in version 4.1.21a. If you would like to update your client proactively, you can update with our download link here." And then it's lastpass.com/lastpassffx. I'm not sure why. Oh, Firefox fix, I guess. Or just Firefox, ffx, Firefox. "You can check which version you are running in your LastPass browser add-on, under the More Options menu in About LastPass. If you're running LastPass 3.0, you are not impacted and do not need to update."

Well, now, okay. So a little more information here. The 3.0 series is what Mozilla has available from download LastPass add-on. I was running v3.3.1. So this never - and of course, as we all know, I'm still an avid Firefox user. But no one using LastPass v3 point anything was ever in danger. So something that they did only for the Firefox version, when they went to v4, caused this problem that Tavis found. And they continue: "Other browsers are not impacted by this report, and users do not need to take action for other browsers. As always, we appreciate the work of the security community to challenge" and so forth.

And they said, in their second portion: "Security is fundamental to what we do here at LastPass. Our first priority is always responding to and fixing reports as quickly as possible. In follow-up to recent news, we want to address in more detail two security reports that have been disclosed to our team. One report was disclosed yesterday, while the other report was responsibly reported and fixed over a year ago." So their initial statement was incorrect, that this was in any way something new that needed some attention. Apparently, as they said, that was long since fixed.

And then they said: "Notably, both exploits do require tricking a user via a phishing attack into going to a malicious website." And my interest in talking about phishing is, I mean, this is an interesting coincidence, but not my main focus. They said: "The first report was responsibly disclosed to our team over a year ago by security researcher Mathias Karlsson and fixed at that time. Karlsson recently posted his findings on the URL parsing bug," which he had found. "All browser clients were updated, and Karlsson confirmed our fix at that time, requiring no action from our users." So this was just old news that sort of recycled because he posted it, even though it had been fixed a long time ago.

"The second report," they write, "was made yesterday by security team researcher Tavis Ormandy, who contacted our team to report a message-hijacking bug that affected the LastPass Firefox add-on. First, an attacker would need to successfully lure a LastPass user to a malicious website. Once there, Ormandy demonstrated that the website could then execute LastPass actions in the background without the user's knowledge, such as deleting items. As noted below, the issue has been fully addressed, and an update with a fix was pushed to all Firefox users using LastPass 4.0."

So this is, as we know, this is the model, in fact this is almost better than the model. One of the things that - I have Tavis's disclosure, which he withheld. And this is all part of Project Zero that we've talked about often, where Google's Project Zero is looking for bugs, not only in Google's stuff, but elsewhere in the industry. He commented on and actually remarked about the speed with which LastPass responded, virtually immediately.

And it was funny because in their standard boilerplate they talk about, I don't remember now if it's 60 or 90 days. But they start a clock, as we've often talked about. And when that clock expires, this thing goes public, whether it's been fixed or not. And so Tavis commented that the clock is irrelevant in this case because it's already fixed. Before he got his posting done it had been resolved.

So anyway, what he found was a way of malicious script interacting with the JavaScript which LastPass injects into a page in order to function under Firefox. We don't yet have a standard for browser add-ons, so every browser exposes a different means for automating its functions. And whatever they did in v4 created this opportunity which Tavis found where a window was being created which was used to pass messages to the add-on. And if malicious script from a malicious website was targeting LastPass Firefox users while this was not fixed, then it would be able to essentially execute LastPass commands in the background.

Again, it's really difficult, especially in JavaScript land. It's why there have been postings on the 'Net saying JavaScript is harmful to security, just because it's very tough, due to the nature of the JavaScript language, which was designed a long time ago. They're trying to increase the security, adding technology to create more containment. But it's just difficult. And we have the problem that, when you go to a website, browsers download script from that website. It's hard to find something that is more problematical.

Anyway, so this got fixed in as good a way as you can imagine. And for what it's worth, anybody like me who stayed on v3 was never in danger. This was a bug that got introduced for Firefox only, whatever they did to change it to v4. I did update to v4. The UI has changed. It sort of seems like it's running a little bit better for me. So I'm glad for that. And bravo for LastPass continuing to do good by us.

Somebody, there was a lot of - when Tavis posted this, there were a lot of people who said, hey, what about 1Password, which was like, I just saw several tweets to that effect. And he actually, Tavis actually posted a kind of a joke picture of some guy, kind of bugeyed, looking at the screen and saying "My first reaction to what I've seen in 1Password." So we don't know what that is yet. But again, this is hard to do. It is very challenging to do this correctly.

So the best we can hope for is that the fundamental architecture is solid, that is, the concepts are solid; and that people like Tavis will look, pry it open and look closely at the specific implementation details because that's what this was. It was an implementation flaw that apparently, I mean, didn't require any rearchitecting or anything, they just fixed it instantly. So it was like, ooh, crap, sorry about that, and everybody gets a new one. So, okay.

So, wireless keyboards. This was in the news last week because there was a sort of a little snapshot, actually a big snapshot. You can go to KeySniffer.net, which is a site that was created to host some of the documents and presentation which will be shown later this week at DEF CON 24. So the short version is the only way to use a keyboard safely is with wire or Bluetooth. And the problem is that Bluetooth is a little expensive. I mean, it's not prohibitively expensive. There are plenty of Bluetooth keyboards around. But what companies try to do, because they're trying to produce very low-cost keyboards, is they think, we don't need all that Bluetooth overhead. It's complicated. You need a processor, I mean, you need a lot of technology. So they just - they try to take a shortcut.

These guys, Bastille Research, looked at eight different keyboards. And these are not obscure, so these are non-Bluetooth wireless keyboards. That's the phrase of death: "non-Bluetooth wireless." You don't want those words all put together in one phrase to describe the keyboard you're using. Anchor, EagleTec, General Electric - I didn't know they made keyboards. Hewlett-Packard, Insignia, Kensington, Radio Shack - I don't think they're making keyboards any longer - and Toshiba.

Leo: Not Logitech.

Steve: Not Logitech.

Leo: Because they're easily the largest wireless keyboard maker.

Steve: Yeah, although they do have their own little dongle gizmo.

Leo: They do, yeah.

Steve: And so I don't know if these guys didn't look at it, or if they didn't find a problem. But here's the story. It turns out that all these keyboards are using the 2.4 GHz band. And they're simply depending upon obscurity. That is, they've developed their own little protocol. There's no encryption. They're just figuring, eh, you know, nobody's going to notice. So last Tuesday...

Leo: Last time we talked about that, weren't they using, like, ROT13 or something to...

Steve: Worse. It was XOR.

Leo: Oh, they were XORing.

Steve: It was a Microsoft keyboard. And as we know, ASCII is eight bits. And so there was the secret was a byte. And so...

Leo: Just XOR it with a byte.

Steve: And it was a static byte. And so what that would do is it would flip, it would invert some of the bits. That was the encryption. And so that technically allowed them to say "encrypted keyboard" and make everyone feel all warm and fuzzy. But encryption like, oh, goodness. And it would be a perfect, like a perfect test question on a final exam in Crypto 101. Here's a series of bytes coming from the Microsoft encrypted keyboard, received over the air. What does this say?

And what you would do is you would look for a byte that was repeating like where a space would probably be. And then you'd check your assumption, if that makes sense. Then you would know that that was a 20 hex, which is a space. But it wouldn't be a space, it would be something different. But the bits that were different from 20 hex would be what was known as the "syndrome," that is, that thing that you XOR. That would then allow you to immediately determine how to flip all the other bits in the message, turning it into ASCII, and then you could read it. So that, you're right, that's what we talked about back then.

Leo: I now understand why Bastille didn't mention the Logitech, because in February they published a report saying the Logitech, Dell, and Lenovo keyboards that use dongles have a design flaw that makes it easy for hackers from as far as 90 meters away to pair with your dongle.

Steve: Oh.

Leo: So they do use 128-bit encryption. They use transceivers made by Nordic Semiconductor. But not all of the keyboards and mice apply it, or [crosstalk] apply it.

Steve: Ah. Okay, so to give people a sense for this, what they're going to be demonstrating in a couple days allows a hacker with a $12 radio device - actually there's a microcopter, you know, like a microdrone? There's a microdrone transmitter which is a USB dongle that you can find for $12. And so it's just a little - it's got the USB connector on one end and one of those cute little mini RF connectors on the other to hook an antenna to. And that's all you need. So $12 to intercept the communication between any of these eight wireless keyboards and a computer from up to 250 feet away. So it gives the hacker the ability to both type keystrokes on the victim's machine, so injecting keystrokes, and record all of the target's typing. So this is - we would call this "no security through obscurity." No encryption used. It's just, I mean, just no thought given to security.

So in their write-up they note that: "Each of the vulnerable keyboards is susceptible to both keystroke sniffing and keystroke injection attacks, keystroke sniffing enabling an attacker to eavesdrop on every keystroke a victim types on their computer from several hundred feet away. The attacker can recover," of course, "email addresses, usernames, passwords, credit card information, mailing addresses" - anything that the user types.

The keyboards are vulnerable to this product of theirs, this KeySniffer software they wrote, using USB dongles at the computer end, because the USB dongle sends out a ping. At regular intervals it's broadcasting, which allows attackers who are aware of this to quickly survey an environment such as within a room, a building, or a public space, for any vulnerable devices, whether anybody is using the computer, typing on the keyboard or not. So all of these also broadcast the fact that this is a target-rich environment. As long as the computer is turned on, and this USB keyboard dongle is powered up, it's sending out a beacon saying "Hack me, hack me, hack me," which their software is able to pick up.

So anyway, bottom line is Bluetooth, as we've discussed years ago, is a well-designed, very secure protocol. You either want a Bluetooth keyboard or a keyboard with a wire in order to be safe.

Leo: Good to know. Good to know.

Steve: Yeah. And I'm glad you followed up because I knew that they had been involved, like nine months ago, that they were doing something else, but I didn't pursue that.

Leo: Yeah. Just use a wire. They don't mention Apple in either of these, but I'm guessing Apple probably is aware of these issues.

Steve: Apple's all Bluetooth.

Leo: Yeah, it's all Bluetooth. So that's safe.

Steve: Yeah.

Leo: Yeah, okay.

Steve: NIST, our National Security government group that generates standards, essentially, for interoperability and use, just posted on the 26th of last month, so last week, an update to their guidelines, among other things dealing with authentication. And they said - they made some changes from what they had said to what they're now saying. And it's a big long document, but there's two relevant paragraphs. The first says: "If the out-of-band verification" - so that's of course what this is. The idea of anything out of band is some other channel than the one you're using. So if you're sitting in front of your browser, talking to a remote server, and it wants to authenticate you, then your phone is out of band. It's not part of that channel between you and the server.

"If the out-of-band verification is to be made using an SMS message on a public mobile telephone network, the verifier shall verify that the preregistered telephone number being used is actually associated with a mobile network and not with a VoIP or other software-based service. It then sends the SMS message to the preregistered telephone number. Changing the preregistered telephone number shall not be possible without two-factor authentication at the time of the change." So they're just sort of formalizing that, but then they deprecate the whole thing. They then say: "OOB, out-of-band verification, using SMS is deprecated and will no longer be allowed in future releases of this guidance."

So this is sort of interim, if you have to use it, if you're using it, at least use it wisely. But we're now formally saying this is not safe. And of course we talked about this weeks ago. And I don't remember now, I was setting up - oh, it was with Hover. When I was establishing my account at Hover, they offered multifactor authentication, and I had a choice of using the temporal-based key or SMS messaging. And I mentioned on the podcast at the time, no way am I using SMS messaging. Not only is it a problem, but you're sending something important every single time you want to authenticate.

The beauty of using a key-based, time-synced cryptography is that you establish that once, and you never need, you know, nothing goes over any wires anymore. It's just the fact that each endpoint knows the same secret that allows them, based on the current time of day, to generate the same six-digit code in order to verify. And of course that changes, due to the secret key, in an unpredictable fashion. So that's the right solution for that kind of additional factor not sending SMS messages. SMS messages, of course, were used because people who have a smartphone can receive them without any - it's just sort of easier. What's the message we just sent you? Unfortunately, as we've seen, the inter-service provider messaging, that SS7 is just...

Leo: Broken. Broken.

Steve: Just never was built with strong authentication, and it's just too easily hacked. The movies that show it being hacked are a little less fiction than we would like them to be.

And then I got a kick out of this. Just on the Windows 10 front, there's a page that I linked to in the show notes. You might get a kick out of it, Leo, if you click on that and bring it up. It's the assistive technologies backdoor into free Windows 10 upgrades. As we know, Windows 10 upgrades ended on July 31st. So they're no more. Yet, if you tell Microsoft that you need to take advantage of assistive technologies, then you press that button down below, and you immediately get a download option for the Windows 10 update. It's a little 5.5MB thing. Probably - I don't know what happens if you're not in Windows. Probably either it won't...

Leo: It downloads an EXE, and I can't use it.

Steve: Ah, right, okay. Yeah, so it's a little 5.5MB EXE that runs the Windows 10 upgrade for you. So Microsoft says on that page: "For the general public, the free upgrade offer for Windows 10 ends on July 29. However, if you use assistive technologies" - I guess, for example, someone tweeted, well, does that mean if I've ever used a screen magnifier, then I qualify? It's like, well, yes, that would be an assistive technology - "you can still get the free upgrade offer, even after the general public deadline expires, as

Microsoft continues our efforts to improve the Windows 10 experience for people who use these technologies. With the Windows 10 Anniversary Update, we've taken a number of steps to improve the accessibility of Windows 10 accessibility. To learn more" - I fact, we've made it very accessible. Just press the button, and you can access it.

Leo: It'd be sad to take advantage of that. I mean, they're trying to be good to people who use JAWS and other screen readers that aren't compatible.

Steve: Well, come on. After a year of having this force-fed down the throats of...

Leo: No, I disagree. I think that's taking advantage of Microsoft doing the right thing.

Steve: Ah.

Leo: I would not do that if I were - I think that'd be unethical to push that button if you weren't using assistive technology.

Steve: Well, the FAQ says: "When does the free upgrade offer extension end?" Oh, you mean because it's bypassing buying a copy of Windows 10?

Leo: Exactly.

Steve: Oh. Anyway, so in the FAQ they said, "When does the free upgrade offer extension end?" And then they say: "We have not announced the end date of the free upgrade offer for customers using assistive technology. We will make a public announcement prior to the end of this offer."

Leo: They're probably waiting till the major assistive programs all work with Windows 10.

Steve: And I posted in the newsgroup the question this morning, GRC's Security Now! newsgroup, wondering what experience anybody has had so far since July 31st. Of course relative to, for example, Never10 and the need for any GWX management. And one person, by the time I put the show notes together, Dave DeBruce, responded. He said: "I never installed KB3035583, which installs the Get Windows 10 installer. It has been sitting in my Recommended Updates for quite some time. Yesterday I noticed" - that is to say Monday, so August 1st - "that after an update check, it is gone. So Microsoft has at least pulled this update out. I know that's not what you asked, but it does look as if they are pulling this stuff out."

Leo: I did get an email from somebody who didn't try to take advantage of the upgrade offer till the 31st. And it ran and worked, and he got authenticated.

Steve: On the last day.

Leo: No. The last day was the 29th. So he waited a couple of days after the last day.

Steve: Oh, on the 31st.

Leo: Yeah.

Steve: Ah.

Leo: So I'm not recommending this as a policy. We also have a number of people I know, and one of them works for me, who got stuck at 99%. This is a fairly common upgrade problem. And it's still stuck. And Microsoft's advice is to go to the Microsoft Store, and they'll help you get through it. And I would imagine at that point they'd arrange for authentication. So they're not - I don't know how cranky they're being about it. Apparently there are some people still doing it.

Steve: Well, I mean, I'm sure you'll talk to Paul and Mary Jo about it tomorrow and see, like, is it, I mean, it's always been a question. I've heard you and Paul and Mary Jo talking...

Leo: They don't - it's not the, yeah, I mean, Microsoft's very clear that it is not free anymore. But if their software lets you do it, I guess it's okay.

Steve: Well, and also remember, too, that everyone knows, and we've talked about it on the show, and you and Paul and Mary Jo talk about, if you upgrade to Windows 10 and make sure that you're authenticated, I think I've heard you talk about it in fact on The Tech Guy.

Leo: Yeah.

Steve: Make sure that you are registered and - what's the term?

Leo: Entitlement.

Steve: Entitlement, right.

Leo: Your machine will get an entitlement to Windows 10. And that entitlement is good forever for that machine.

Steve: Right.

Leo: So you could opt to go to Windows 10 later. But you have activated on that machine, that's the key.

Steve: Right. A couple little bits of errata. Bruce Wilson, who's an enterprise architect with Oak Ridge National Laboratory, he tweeted a note. His Twitter handle is @usethedata. So he confirmed my concern, which we discussed last week, over Firefox not being immune to certificate tampering.

Remember that we had, in our Q&A last week, one of our listeners said that his corporation does have an encrypting or decrypting proxy, and has pushed a certificate onto their machines. But Firefox uses its own certificate store. Does that mean that he's safe? And I wasn't able to definitively say one way or the other. I said, well, yeah, it is true that it has its own store, but I don't know that it's not possible for that to be affected also. And even if it weren't affected, that is, if you could maintain security, it might very well be that a tightly locked-down corporation would prevent Firefox from getting an HTTPS connection out to the Internet unless it went through their proxy. So you might not be able to use Firefox either way.

Anyway, but Bruce confirmed, from his experience, he said, regarding using Firefox to avoid corporate spying: "If a Windows box is joined to an Active Directory domain, the corporation can run any arbitrary script on the box, including scripts to push a certificate into the Firefox certificate store. Fundamentally, if it's a corporately managed system, tools like" - and he writes SCCM, which is the System Center Configuration Manager, formerly known as SMS, which is the Systems Management Server - "allow the admins to do pretty much anything."

So I wanted to close that loop, that, yeah, I mean, so the next solution is certainly more overhead, but that would be to use a VM. Get like a little virtual box or as small a little VM environment as you can and put Firefox in there. The corporation and its scripts will not be able to get in and alter Firefox's certificate store running in a virtual machine, yet the virtual machine would still be able to have access to that workstation's networking, and then you could see whether you're able to serve privately, whether you are able to establish a connection out to the Internet without going through the decrypting proxy.

And second, a bunch of people - I sort of anticipated this, but it was fun to see the response - took some exception to my comments about how TIFF format was dead and saying, no, no, no, no, no. Apparently library sciences and archiving are big on TIFF images. And due to its age, that's sort of the default format for fax scanning. And that's all true.

Leo: Yeah. A lot of scanner software still scans to TIFF, as well.

Steve: Yeah. And I should have noted also that, in fact, file formats never die. I mean, in the same way that old software doesn't, it's still around, file formats, eh, no.

Leo: TIFF has a good place because bitmap is uncompressed. JPEG is lossy compression. TIFF has lossless compression built in.

Steve: Like PNG. PNG is sort of the inheritor of the lossless compression format.

Leo: PNG is lossless? It's not lossy?

Steve: Yeah, lossless.

Leo: Oh, all right, cool.

Steve: Yeah. And actually very good compression technology, too.

Leo: Yeah, much better than TIFF, I'm sure. TIFF uses Lempel-Ziv.

Steve: If you ever want proof that old formats never die, look at any graphics program under the Save As menu. It's like, I mean, it's just - and it's not like anyone would implement a driver for some of those. It's just that v1 of Corel Draw or Photo Paint had support for it, so v25 does, too. So, I mean, really obscure things that no one has ever seen. And, yes, TIFF, as well.

Leo: TIFF is - somebody in the chatroom is saying it's also multipage, which is why it's still used for faxes. Like PDF, you can have multiple pages in a TIFF. You can't do that in a PNG or a JPEG.

Steve: Right.

Leo: So, see? There's still some good. There's still some life left in her.

Steve: I got a note - no, I didn't get a note. I stumbled on this last week, and I tweeted it. Peter Hamilton has written a short book, believe it or not. He calls it a novella. And he's actually written several others before. So this short form is something that he likes. Anyway, it was apparently just released last week. I just wanted to give our listeners a heads-up. It's called "A Window into Time." Four dollars, so not very expensive. Not very long, though, 95 pages. So, yes, an actually short book.

The back cover gives us a little clue into what it is, saying: "Whip-smart 13-year-old Julian Costello Proctor, better known as Jules, has an eidetic memory. For as long as he can remember, he has remembered everything. 'My mind is always on,' he explains. But when an unexpected death throws his life into turmoil, Jules begins to experience something strange. For the first time, there are holes in his memory. But that's not the strangest part. What's really weird isn't what he's forgotten, it's what he remembers. Memories of another life, not his own. And not from some distant past. No, these memories belong to a man who's alive right now.

"With bravery, ingenuity, and quirky good humor, Jules devises a theory to explain this baffling phenomenon. While tracking down the identity of his mysterious doppelganger, he finds himself enmeshed in the hopes and dreams of a stranger - and caught in the coils of a madman's deadly plot." So it sounds fun. And I don't think I've read a bad Hamilton book. I've read some really laboriously long ones, notably "The Great North

Road."

Leo: Short might be good. Yeah.

Steve: "The Great North Road."

Leo: Short might be an advantage.

Steve: Yeah. Yeah. And I just trust Peter. I remember thinking, you know, "The Dreaming Void," that just really sounded like not something I want. I don't want - and it turned out to be fabulous, a wonderful trilogy.

Leo: That wasn't the Al Capone one.

Steve: No.

Leo: That was not so good.

Steve: That was the reality dysfunction one.

Leo: Yeah.

Steve: And I agree with you, Leo, that kind of went a little bit off the deep end.

Leo: Yeah, yeah.

Steve: But a lot of people loved it. And by the way, somebody has already finished it from my tweet, someone named Steverino, of all names, says: "Thanks for this fun little read, Steve. 'A Window in Time' was thoroughly enjoyable and short."

Leo: Yeah, nice.

Steve: And I did have a fun story. I thought that maybe we would be talking about Drobo, so I had a SpinRite Drobo adventure to share. Istvan Burbank, maybe Istvan Burbank. Anyway, he said, "Ah. I had used a Drobo for years before my own NAS, and have only good things to say about it, especially about being able to put different sized drives in." We'll see why in a second here. He says: "I ran SpinRite on friends' broken drives, recovered the drive, and copied their data to a new drive. And if the fixed drive was bigger than one of the drives in my Drobo, I would hot swap it in without much worry about the drive failing again due to the Drobo's redundancy, and my Drobo's capacity would automagically increase."

So I thought that was kind of clever. He's helping his friends. He's using SpinRite, recovering their data, then cloning that to a new drive. And he's now got a drive that's like, eh, well, we're not quite as sure about it as if it had never failed, so we'll stick it into a place where, if it has any trouble, we'll be protected. So kind of clever.

Leo: That's cool. That's a good idea, actually. And that would be a good place to put it, a Drobo would be, actually, because if it fails, no big deal, just put another one in. Yeah.

Steve: Yeah. So phishing and filtering. [Sigh]

Leo: [Sigh] I like that. [Sigh] Where do I begin?

Steve: Where do we start? So here's, okay, so here's the problem. And phishing is sort of the way you stumble into the larger problem, which is website spoofing. It's something that I've spent a lot of time over the last three years looking at because of course I'm actively involved with SQRL and this authentication solution, where this has really dogged me.

And in fact the whole project, essentially, I wouldn't say it came to a stop last summer, but I took it on a diversion because I didn't feel like I clearly understood exactly what the nature of the problem was in the context of SQRL. And I absolutely wanted to make sure there wasn't a solution, there wasn't something we had missed. And as it turns out, I found a solution and implemented it and made it go, and it works. And then we decided, eh, we would alter the spec to support it. But as I'll explain later, we ended up not - I ended up removing it from the client.

Okay. So here's the problem. And to varying degrees, every authentication technology is vulnerable to it today. Some modes of SQRL are very resistant, but there are still edge cases. And it is just a problem that the industry as a whole doesn't have a good solution to. And that is, if the user of a web browser is fooled about where they are, that is, you believe you're on a website that looks exactly like your bank - and we've talked about this through the years but never really looked at the exploitation side, which is what I wanted to talk about today. If you miss the fact that the domain name is not correct, it's not BofA.com, and there have been, of course, all kinds of, I mean, this problem has been around for so long that there's a long history of exploits of this problem, that is, how do we fool the user into believing they're at one site, when they're actually at another?

Leo: Oh, I got a phishing scam I almost - I came this close to falling to. It wasn't from Twitter.com, it was from Tvvitter.

Steve: Perfect example.

Leo: The two V's looked just like a W. You could barely tell the difference.

Steve: Perfect example.

Leo: Yeah. It's just a spelling difference; right?

Steve: And so here's why this is a problem. Well, first of all, obviously you don't want to be at a site that, you know, the only person, the only reason someone is going to have you at T-V-V-I-T-T-E-R dotcom is they want to get up to some mischief. And what should typically - what typically happens is you're presented with, oh, please enter your username and password. So in the simplest case, you land on a spoofed website asking you to authenticate. And most users will say, oh, okay, and enter their username and password. Now, the bad news is, obviously, you've just sent your username and password to this malicious site that now has access to your actual account on the actual site.

Now, one mitigation is password managers because they're not fooled. LastPass, for example, it's not going to get a string match on T-V-V-I-T-E-R dotcom. It'll just come, I mean, it won't show anything. It won't populate the fields. It'll think you're at somewhere that you've never been before. So that would be, like, a solution except that, in my experience with LastPass and password managers in general, and I know many others, sometimes what website do confuses the password manager. So that even on a site that you know LastPass has an entry for, it's not populating the fields.

And so I know LastPass users do what I have done in such cases is you open the vault, and you manually copy your username and password over into the fields because for whatever reason the script on this page is fighting with the add-on. And so what would have been a protection, unfortunately, through sort of social engineering, it's been defeated.

Now, this article that appeared that sort of put this on my radar and brought it to the fore was in the Hacker News last week. There was an article talking about how the QR code logins can be trivially defeated with this approach. And so in looking at it, it's like, yes, all logins can be defeated with this approach because, unfortunately, this spoofing is that powerful that we're still looking for some way to protect the user from fraudulent websites. And unfortunately, as we've talked about in different contexts, even using multifactor authentication isn't a solution.

And so here's where it gets a little bit trickier at the plumbing end. Because, for example, so you go to the spoofing site. And the fake site presents you with a page asking you to authenticate. What happens behind the scenes is that it, that is, the server running the site accesses - the server running the fraudulent site accesses the real site as if it were you. That is, for example, maybe you fill in username and password and hit Enter. That goes to the malicious server. The malicious server essentially does the same thing you've just done, pretending to be you. So it brings up a web page on the real site, provides it with your username and password.

It then notices that that site is requesting some additional authentication. That is, even if you have established second-factor authentication, an out-of-band authentication, that site that it is impersonating you to challenges it for your second factor. And it doesn't matter what the challenge is. It could show a QR code and say, here's a QR code you need to scan. Well, what does it do? That malicious server grabs that QR code that is the second-factor authentication challenge and does, again, does the same thing to you on the malicious site. It says, oh, you're using second-factor authentication. Please scan this QR code.

So it has essentially inserted itself as a classic man in the middle. And it has not done so by hacking into HTTPS or not having security and using an HTTP connection or anything. It's simply done it because you're not actually at the site you think you are. And that allows it to interpose itself into the communication chain. And so this is what I spent so much time last summer brainstorming, was is there nothing we can do to prevent this from happening? That is, for example, if you had a time-based second-factor, it would say please enter - the real site would challenge the malicious site for the six-digit code. Seeing that, the malicious site challenges the user for the six-digit code. The user says, oh, right, gets their phone or wherever they're running their authenticator, even one of the old-style PayPal footballs, you know, whatever, and enters the six-digit code into the site, which the malicious server then forwards to the real server, having succeeded and defeated the second factor. Same thing for SMS.

I mean, whatever the challenge is, once this malicious actor has imposed itself between the user and the real site, anything - essentially, it's able to, even with HTTPS over SSL connections, because you're making a secure connection to T-V-V-I-T-E-R dotcom. And it could even be an EV cert. I mean, it could look authentic, although I'm not sure what it would say for the, well, I guess the issuer of the certificate would have had to have been willing to issue a certificate which was suspiciously like Twitter, and maybe you couldn't pass that.

But as we know, things like Let's Encrypt doesn't have any kind of human intervention there. So certainly getting SSL certs no longer requires human interaction, making it even easier to pull off this kind of spoofing. So once there is a malicious server in the chain, anything the real site provides, would be providing to the user, it provides to the malicious server, which then provides it to the user. And it's able to maintain its position.

So the first thing I wanted to note is that this is a problem today that, to some degree, automated password managers can help because they're not going to see a string match. They're going to say, wait a minute, this is not T-WI-T-T-E-R. And so they won't automatically populate the field. In my experience, the password managers tend to run across sites where they don't function as smoothly as they might often enough that an unwitting user would think, oh, well, okay, it's one of those, and then go manually populate it.

So again, it's not, doesn't give us the kind of strength we would like to have. And any kind of additional logon information that anyone has been able to think of succumbs to this, whether it's an optical QR code, it's a time-based code, it's an SMS message. For example, for a while we were talking about, like, choose which is the proper picture. And then so you'd be shown a grid of pictures. Well, again, the valid site challenges the user, which in this case unfortunately is the malicious server, with this group of pictures, which the malicious server forwards to the actual user as, oh, look, you've got to choose, find the kitty cat that you have chosen as yours and click on it. So the malicious server sees you do that, it simply sends your response through it back to the valid site in order to defeat authentication.

So I wanted to explain first how this plumbing works, and why this is such an intractable problem. It is very simple to get into this situation. And phishing is normally the way you fall into spoofing because no one is going to themselves type in T-V-V-I-T-E-R dotcom. We're all going to type in Twitter.com. But links in email, links in social media postings, links in Twitter messages, anywhere where the process of going to URL has been automated, you just - you are expecting to go to Twitter. And what comes up looks like Twitter. And unless you are really good about looking at the URL, like making sure - and I have to say, as the co-host of a Security Now! podcast, when I'm doing something that is really important, I will, you know, I do make sure that, if this is something crucial, I go look at it. But I know that most people don't.

And so this is the way we fall into this problem so much. So the only mitigation that I've been able to come up with, working with the gang in the SQRL newsgroup, was a proposal that I presented to them last summer and then implemented. And the idea was that the problem arises because of this man in the middle. And what is somehow necessary is to short-circuit that man in the middle. That is, to somehow get a direct connection between the authenticator and the user. And this is something I've understood from day one was a problem.

The very first working spec for SQRL, for example, had this notion of it's a bit in what the server sends back to the SQRL client called "same IP" because that's one of the first giveaways which SQRL is able to capture. And that is that, when in the SQRL implementation of this problem with phishing, the bad guy, that is, this malicious server, would ask the good server for the QR code. And embedded in the SQRL QR code is the IP that made the request. And when the SQRL client then scans the QR code, or if you have the client built into the computer, you just click on the QR code because it's a standard clickable href link. The client running in the computer performs the authentication query with the server.

Well, we would expect the IP addresses to match in that case because the IP address of the user which requested the QR code should be the same IP as performed the authentication, if the SQRL client is in the same machine as what the user is using. But that's not necessarily the case in a mobile login, unless your smartphone is also on the same network, on the same LAN, in which case you would have the same public IP. But if it's on a wireless LAN, it's going to have a different IP. So we've handled that in the specification, that is, this concept of whether the same IP is expected or not.

So in the case of somebody using SQRL to login where they have the client running in the same machine, or if they're logging into a mobile site with their mobile device, same thing. You are protected because the IP that requested the QR code is the same IP as performs the authentication. And that completely shuts out this man in the middle that's inherently at some other IP, somewhere on the Internet.

Now, it's true, if there was an evil person somehow operating in your same network, then you'd both have the same IP. So we recognize that comparing the IP that requested the QR code to the IP address that's performing the authentication is not a guarantee of no spoofing, but it is very strong protection that doesn't exist anywhere else.

For SQRL, when the authenticator, the authenticating device, whether it's the same computer or a mobile device on the same network, is on the same network as the computer that you're using. But we were able to do something one step more clever. And this is what we implemented last summer, and that is, when the user logging into a site clicks on the link to authenticate, the web browser would generate a query to the client itself, to the local SQRL client. There's a longstanding tradition in Unix of using so-called localhost servers. You run servers on 127.0.0.1 is the localhost IP. Basically it's the machine's own IP. And I just got a pop-up here. I'm sorry, it distracted me.

Leo: What does it want? Is it from some guy in Nigeria offering you $14 million?

Steve: Just microphone settings. It's complaining about my microphone volume.

Leo: You know, Skype has been laboring a little bit. Your picture was kind of wonky for a while. The sound's been fine, so I haven't said anything.

Steve: Good.

Leo: They did, by the way, arrest Mike, the Nigerian prince. He's been arrested. He scammed...

Steve: No kidding.

Leo: Yeah. Well, he's one of them, I'm sure.

Steve: [Crosstalk] one Nigerian prince?

Leo: They got one. Apparently it scammed one person out of $11 million. But many others, as well. And, yeah, just arrested him.

Steve: Even now.

Leo: Even now people fall for that. You know what, probably lonely people who, you know, maybe just want to make a friend.

Steve: Maybe he's a nice prince.

Leo: He's a nice prince.

Steve: Yeah, a nice prince. So in this model, the SQRL client running in your machine sets up a localhost server on a well-known port that we reserve for SQRL so that the client itself, running in your computer, is able to receive queries from the browser. And again, this is commonly done. Unix, I don't think, could operate without localhost. If you ever, I mean, and Windows is using it like crazy, as well. It's just a very convenient way for different processes within a single machine to talk to each other through the sockets interface.

So the beauty of this is that, when SQRL is running in that mode, and you authenticate with SQRL, what we call it is we have an abbreviation, CPS, Client Provided Session. What happens is the SQRL client sets a flag in the query to the SQRL authentication server saying "Client-provided session is in use." So the server, instead of authenticating the browser's session, it sends the authentication token back to the client, which then it's able to set in the browser in responding to the browser's query because, as we know, when a server responds, it's able to set cookies.

So essentially this is a means where using SQRL with this feature, the remote server is sending the authenticated session back to the SQRL client, which it then gives to the local browser. And the point is there's no way for any man in the middle to obtain that. A man in the middle is wanting the session that it has established with the server to become authenticated. The idea is that the malicious server initiates this login and then is forwarding everything to the user, to essentially perform the authentication on its behalf.

But the point is that by then forwarding everything to the server, the malicious server's session with the real server that gets authenticated, using this client-provided session feature, the real server sends that final authentication to the SQRL client, which then, in the same machine, provides it to the browser, to the user's browser. So the user's browser is what gets the login session, not the man in the middle.

Now, we implemented this. It was up and running. Everybody was happy. And then we found out that Microsoft was making noises about shutting down this whole localhost feature. There was somebody who was participating in the newsgroup who was following developments at Microsoft, sort of the way Paul and Mary Jo do, you know, very much into what's going on. And in Windows 10 it was not going to be allowed. That is, this next version, the one that's now a year old, was going to be - and so imagine, this is just happening. It was last summer around the same time. We'd, like, solved this major problem, absolute bulletproof site-spoofing protection for SQRL.

And then the news that we cannot - we will soon not, like Windows apps will not be able to establish servers that other apps can access. I assume this is just Microsoft looking for more ways to tighten things down. At the last minute there was such an outcry from the developer community, because this was going to break so much, that Microsoft backed up, and the default setting was changed from default disallow to default allow. But it's still there, and they're telegraphing their intentions. And apparently there's - I don't remember now the term. I'm sure you'll know it, Leo. There's some class of apps that, like, Microsoft-approved apps or there's like somewhere you get them or something.

Leo: Yeah, they're called UWP, Universal Windows Platform. They're in the Microsoft Store.

Steve: Ah, exactly.

Leo: It's like the Apple Store or the iOS stuff, yeah.

Steve: Right. And those will not be permitted to use localhost communications. Now, I don't know if I care about that for my own Windows client. But it just seemed like, oh, shoot. I mean, here was this perfect solution for this problem. But it looks like we're not going to be able to count on it in the future.

So what we did was I ripped all that plumbing out of SQRL. We backed out, went back to the previous design, but kept the client-provided session feature in the spec because what I think is foreseeable is that my Windows client, and even Jeff's running on iOS, and a bunch of people are working on them for Android, these are separate clients running on the platform. It is entirely foreseeable that this will move into a browser add-on. That is, it'll be maybe ultimately, if SQRL were to succeed, built into browsers natively. So it's not even an add-on, it's just this browser is able to authenticate you with SQRL.

And if that happens, then the beauty is we no longer have this communication problem where the browser and an external client are trying to talk because the client will no longer be external. It will be in the browser, in which case SQRL will be able to use this client-provided session, and it will be impossible to fool SQRL and for any man in the middle to obtain a login session when you're using SQRL that is either an add-on or natively built in the browser.

So we have strong protection today. The only way of defeating it is strong protection with the same IP, which will catch any instance where we would expect the IP to be the same. That is, you're using a mobile phone on the same network, or you're using a client on the same computer, in which case the public IP should be the same. If they're not, we don't proceed. And that would protect SQRL users from all of the typical man in the middle where there's another IP. That malicious server IP would be detected and prevent any authentication.

But in general, this site spoofing is a big problem. And so I wanted to create some context for that article that the Hacker News reported from someone who said, oh, look, you know, it's easy to fool these multifactor authenticators. It's like - my screen keeps blanking out. It's like, yes, it's actually - it's easy to fool everything. This is a problem that has not been solved. And the good news is SQRL provides strong but not perfect protection now, but the promise of perfect protection as soon as it actually moves natively into, well, either as an add-on or natively into browsers.

But, yeah, this phishing and website spoofing is a problem. There's just no way around it. It's been around forever. We've talked through the years about all the ways of, like, obscuring a URL so that it's like www.amazon.co.uk - well, that's actually a valid domain. But, I mean, www.amazon.info, you know, who knows who has that. Maybe Amazon preemptively registered it. But there are a lot of people who would see Amazon.info and think, oh, okay, that just - I guess that's okay. I mean, this whole URL domain name thing was never designed for primetime. It just sort of happened, and now we're stuck with it. So anyway, I wanted to sort of go into more detail into the nature of the site spoof and why none of today's authentication systems provide perfect protection. But happily there's some perfect protection on the way, courtesy of that little guy.

Leo: Woohoo, SQRL.

Steve: Okay. Second topic. Packet filtering. We've been talking a lot in the last couple weeks about these various routers. This sort of began months ago when I switched from those two old T1 lines that I had to a cable modem. And at that time I decided it was time to sort of dust things off and upgrade. So I went with that little Soekris Engineering PC platform running pfSense. And I'm so happy with pfSense. It's also, as our listeners know, it is a great solution if you have an old PC, or maybe even like a little fanless, diskless PC. It doesn't need much. It just needs a couple interfaces because pfSense is freely downloadable, and you install it in a machine and set up a very capable router.

Then, in the context of Internet of Things, we've been talking, of course, for quite a while about the need for isolating networks. And that's really what I want to talk about here. And it's germane because not only could pfSense do that, if you had multiple interfaces, but then we found the little Ubiquiti EdgeRouter, which has five separate interfaces, each that can be configured to be a separate set of IP addresses, that is, separate subnets operating within your domain as individual Intranets. And I haven't really gone into detail about what you do with these multiple interfaces. And so I wanted to talk a little bit about packet filtering in the context of how any of these different multi-interface switches or firewalls would function.

And we also talked, for one week, I was talking about the Cisco SG300 device, which was the first thing I had found and liked. That wasn't a router. That was a multiport switch. And there's a little, well, there are definite distinctions in what switches offer, what features switches offer versus routers. But then of course we went from the SG300 to the Ubiquiti, which for $49 arguably provides the most bang for the buck I've ever seen. The previous simple, low-configuration approach we've referred to often as the "three dumb routers," where you create - you take just three generic routers in a "Y" configuration in order to provide just plug-it-in, bulletproof network isolation. However, if you begin to want to have some cross-network communications, things get a little trickier.

And I would argue, now that there is the Ubiquiti $49 router on everyone's radar, that just makes way more sense. First of all, you've got much more flexibility and all kinds of cool features. I don't think I've mentioned that the various rules which you use for configuring the router can, for example, have time-of-day enable and disable. So, for example, the kids' network could shut down at 9:00 p.m. and just not work anymore until morning. Or whatever you wanted to do. So all kinds of flexibility.

So the way to think of these, any of these multi-interface devices is that plugged into each one of the physical ports is a network segment, which is, I mean, it could be just one IP. But normally it's a block of IPs. And I know that everyone who's been playing with home networking has seen 192.168.0.x, typically, or some routers are .1.x. And in fact the spec, the IETF spec sets aside 192.168 dot anything dot anything. So the last two bytes are completely available. So that means you've got 64,000, well, minus some overhead per network, IPs available.

So, for example, one thing you could do, one way you could configure things is to number - you'll have one port that is the WAN, that is the Wide Area Network side, connected one way or the other, DSL modem or cable modem or whatever. And as we know, it is possible to load-balance the WAN side, that is, you could have two WAN links, maybe of different types, so that if one goes down, the other one picks up the slack automatically. That's the kind of power that we have with these late-model router devices like the Ubiquiti Edge Router.

But say that you just had one port for WAN going to a DSL modem or cable modem. Now you've got four more ports. And you could label them 0, 1, 2, and 3, and assign to each one of them one of those 192.168 networks. So port 0 would be 192.168.0.x. Port 1 would be .1.x. Port 2, 2.x, and Port 3, .3.x. So now you have four disjoint networks where they're able to be treated individually and independently. And that's the key. That's what's different between these routers, as I've said before, where you actually have not only physically separate ports, but logically separate ports, so that they can have their own networks assigned to them. Well, more than that, they can have their own filters assigned to them.

So what's a filter? The way to picture this is that on each of the ports, there is something inspecting the packets coming up the wire into the router. Some access control systems allow you to specify filtering inbound separately from outbound. But in general, most systems are inbound. That is, so it's data coming into the port runs through, is like inspected with the so-called filters. And once the data is past them, sort of in the inside of this device, it's free then to go wherever it wants to based on the address.

So what this essentially allows is the individual traffic to be differently filtered on different networks. So, for example, imagine that .0.1 is your main, normal, non-IoT, your PCs, maybe your entertainment systems. You might want them to be on their own network, just to keep them separate. The point is this gives you complete control. Years ago, we spent a lot of time talking more about the plumbing of the Internet and the fact that packets have both, well, essentially the design is hierarchical. So you have Internet packets that have a source IP and a destination IP. And some of the protocols that are carried within that IP packet, like TCP or UDP, have this notion of ports, a source port and a destination port. All those are, are 16-bit values carried in the packet.

When the packet arrives at its destination, the device is able to aim, to sort of route that packet to the proper destination within the device. So, for example, say that you had a server, you had server hardware running a web server and a mail server. Well, packets with a destination port of 80 or 443, for HTTP and HTTPS respectively, they would be given to the web server because when it started up its service it said to the operating system, I'm listening on ports 80 and 443, which is sort of an abstraction.

I mean, there aren't physical ports anywhere that it's actually listening to. What it says is it just sort of registers itself in a table inside the operating system, saying any inbound packets with a destination port of 80 or 443, I get them. And similarly, the email server, for example, an SMTP server wants to listen for packets coming in on port 25, which is the agreed-upon port for email.

So all of the packets moving through this switch are coming from the devices outside. Their traffic is coming into the switch. And at entry to the switch, we have the opportunity to decide what we want to do. We can perform matching on any of these fields - on the source IP, on the destination IP, on the source port, on the destination port - and also some additional characteristics I'll talk about in a second. But that's a very powerful facility because, for example, we could say, if traffic was coming in, we could allow traffic in as long as it was going to be going out to the Internet and not to any of the other networks on the switch. That is, traffic is allowed through that port into this little switch router so long as it's not going to 192.168 dot anything dot anything. So that it can come in, but it won't be allowed in if it's going to try to go out of one of the other connections.

And so you can see that that's an immediate simple way of creating isolation so that, for example, your IoT devices you might have configured that way so that they're able to have access to the Internet, but no traffic from them is able to go to any of your other networks, so there's no way for them to scan or probe or get up to any mischief because you've said only allow their packets in if they're not destined for any 192.168 addresses. And so that's an example of the kind of rule.

Now, that's just IP-based filtering. But because we also have ports, we have a lot more power. So, for example, we might say only allow them if they're going to remote web services, that is, if the destination port is port 80 or 443. In which case suddenly now we've also essentially blocked anything from any external connection other than remote web servers. Or maybe we want to allow or block remote email servers at port 25.

The point is that, by inspecting every packet that enters this little router switch and performing a series of tests that match specific fields, we're able to either just say drop the packet, just discard it, pretend we never received it. Or in some cases we can log it, if we think that would be interesting, if we want to see that it happened. We can reject it, which sends back a message saying, sorry, that packet is undeliverable. Or it can be accepted and allowed to pass through. And once we create disjoint, that is, completely separate subnetworks for the various ports, then because they're in separate address spaces, we can use the same rules to control inter-port access.

So, for example, some traffic would be permitted, and others would not be. There's also this notion, we've all heard the term "stateful packet inspection" or "stateful firewall." Well, for example, the Ubiquiti EdgeRouter and many of the good routers are more than stateless filters. They're stateful, meaning that actions that packets take can alter the rules dynamically. So, for example, in the Ubiquiti EdgeRouter, you're able to allow - you're able to set a rule to allow a new connection to move through the filter, or an established connection to move through.

In TCP, we have famously the SYN packet, the synchronized packet, which is absolutely required to establish a new TCP connection. So although the UI, for example, and the Ubiquiti doesn't specify SYN packets, it broadens the term to a new connection. That's what it means. It's allow a connection to be created through this filter. And then this notion of an established connection can also apply to UDP packets that don't have SYNs.

But, for example, sort of in the way a NAT router works, we've talked about how good NAT router protection is because nothing unexpected from the outside is able to get through. It's only outbound traffic that creates a table entry that allows the reciprocal packet, that is, a packet coming back from the IP and port that the outbound packet was going to and coming back to the IP and port that that outbound packet came from is allowed to enter. That is, the router remembers that there was traffic that was initiated outwards, and so the reflection, the reply traffic is able to come back.

The Ubiquiti does the same thing with this notion of an established connection. So it would be possible to have this isolated network segment of IoT that has no connectivity into your network, that is, like into the highest security zero network. No unexpected packets are able to get in. In exactly an analogous fashion to the way NAT protects us from the hostile exterior Internet, we can create a NAT, that is, the equivalent, it's not really doing translation, but it is doing stateful firewall filtering such that the low security IoT network has no access into any of the other networks.

But those networks do have access into the IoT network. That is, by configuring the rules correctly, new connections would be allowed to be made into the low security network, and then its reply traffic, only as long as it was coming back to the same IP and port as originated it, would be allowed to flow.

So think about that. I mean, first of all, it means you can get yourself easily tangled up in how all this works. But it's all just sort of try things and experiment with them, see if they work the way they don't, and you'll learn a lot in the process. So what this essentially means is we're used to thinking in terms of routers as WAN and LAN. That is, outside hostile, inside safe. And it's because that boundary has filtering and the equivalent of a stateful firewall.

Well, now what we have is within our own network we've got, instead of just sort of two sides, an outside and an inside, now in the case of, for example, a five-port Ubiquiti router, we have five networks, each with separate ranges of addresses, each where we can put inspection filters on the traffic trying to enter the common router from the network and decide what we want to do with them.

The last thing I'll say is that the way these rules function is very clever. It was established a long time ago, and it's so powerful and flexible that it's the way things are today. And that is, the rules are ordered so that when a packet comes in - they're typically called ACLs, Access Control Lists. You have an ordered list of rules. So, for example, if a packet has this or that, and it's whatever you want to have matching, IP address or source and destination port, whatever, it will be accepted, rejected, or dropped. Or maybe it won't match that rule at all, in which case the next rule is processed.

But the first rule in this ordered list, which is where the selection criteria for the rule match, ends the processing of the list. And that packet is then thrown away, or accepted and allowed to proceed and enter, or perhaps a logging entry is made or whatever. As soon as the rule matches, we're done. And in terms of high performance what that says is, if you really want the most performance, you want the highest bandwidth rules to occur earlier in the list because it means you have less rule processing necessary per packet. You wouldn't want to, if you had a choice, to put a high bandwidth rule that is, like, doing a whole lot of work, like streaming media or something, at the bottom of the list because you'd have to check every single packet through the entire list until you finally got the permission that you were looking for. If it doesn't break your security model, you'd like to put the most often executed rules at the top of the list.

So anyway, we've sort of glossed over the power, that is, the application of the power of these multi-logical interface routers and switches. I mean, it's just another whole world of fun that people can have with networking. And, boy, when you take this concept of rules, the fact that you can put time locks and clocks on them, you can log things, you can drop the packets, you can selectively decide what you allow in and what you don't, you can come up with a really locked-down, very secure, powerful network architecture for $49. And then a lot of hours' worth of your time fiddling with it. But it's fun fiddling.

Leo: Well, as long as it's fun fiddling, I'll take it. I will be doing that exact fiddling soon.

Steve: It's just neat.

Leo: Yeah.

Steve: It's just it's fun to see this stuff work.

Leo: Yeah. And you use, somebody said, Neo's saying in the chatroom, well, he likes pfSense. You use pfSense for other purposes.

Steve: Yeah.

Leo: But you use pfSense, yeah.

Steve: Yeah. pfSense is a router on the top of the bookshelf behind me. And as I was mentioning before, for example, one of the last things on my to-do list for SQRL - I'm starting into the final list of to-do stuff. I posted a pre-podcast update to the SQRL gang last night as I switched over to start pulling the podcast together with a rejiggering of some of the user interface terminology because we came up with this concept of rekeying an identity, and that wasn't the way I was thinking of. I was considering it a new identity originally, and the UI still represented that. So that's all been changed.

But, for example, one of the things that I need to do is to add proxy support because in corporations, as we've said, the web browsers are configured through a proxy, and you can only get to the outside world through the proxy. So I needed to make SQRL proxy aware. Well, I'm not a corporation. I'm just me. But in order to test this I need to create a proxy. And what do you know, pfSense has the world's most popular proxy, called Squid. And so...

Leo: That's great, yeah.

Steve: ...with a couple button presses, I now can set up a corporate emulating environment in order to verify the proxy support.

Leo: Oh, nice. Yeah, that's a great idea, yeah.

Steve: So, yeah.

Leo: Yeah. So in honor, I think, of DEF CON or Black Hat, somebody dropped this off for you. And I gather he tweeted you or let you know that he was coming?

Steve: Yeah.

Leo: He said: "Steve will know what this is." But I think there are some clues on it. It's a big PC board, I mean big, like 15x15 inches. And it has slots for surface-mounted chips. There's only one on here right now. But the chip, I think, might be a clue to this: AWT-4500 Deep Crack.

Steve: Yup.

Leo: So I guess you could put 1, 2, 3, 4, 5, 6 by 1, 2, 3, 4, 5, 30 Deep Crack chips on here, and it looks like this row would be controllers. It is from Cryptography Research Advanced Wireless Technologies and the Electronic Frontier Foundation. What is this?

Steve: This was, and we've talked about this in the past, this was a prototype of the Deep Crack DES cracker. When the EFF wanted to prove that DES was not secure - and that was, you remember, DES is a 56-bit cipher.

Leo: Right.

Steve: So it's only 3DES where you have the individual 56-bit, three separate 56-bit keys that you get security, or reasonable security. So this was 56-bit DES cipher. And they said, you know, we're going to crack this. And this was a long time ago . So this was early LSI integration to produce essentially a hardware GPU-style-ish accelerator in order to brute-force crack DES. And so this board wasn't used because they only got one, they got that one chip populated, and then they ended up obsoleting that one and doing a second-generation board. But so you're holding a piece of history in your hand there, my friend.

Leo: Well, and it's autographed. I can't quite read the autograph. It says "Carl." I guess it's Carl that gave you this. I don't know. "Thanks for your long-term friendship." And then it's signed by somebody with the last name J-O - Johnson, I think, maybe?

Steve: My guess is that that was a gift to him, and then he's re-gifting it to us.

Leo: Ah. Well, to somebody named Carl. Anyway, yeah. So do you know the Twitter handle of the person who tweeted you?

Steve: I do. I'll be happy to send it to you.

Leo: Yeah, because I'd like to thank him. I think what we'll do is we have some framed - we have a framed core memory, and a framed motherboard from a Macintosh IIfx, which stood for "too freaking expensive," as I remember. And then I'd love to add this because that's great history. That's great history.

Steve: Yes, yes.

Leo: I will definitely frame this and put this in our wall of memories. Which is getting smaller because we're moving to a smaller place. But I like the providence of this. It's very interesting.

Steve: Yeah.

Leo: That's fascinating, yeah. Well, we've done it again, Steve, killed a couple hours talking about security and technology and networking. And we'll save the RAID for another day.

Steve: We will. And presumably, given that DEF CON is as full a basket of goodies as it always has been, we'll have some really fun things to talk about next week.

Leo: Can't wait. Father Robert is there on our behalf, risking his life and limb, or at least his data.

Steve: Just keep your WiFi turned off. It's deadly.

Leo: Bring a burner phone.

Steve: You do not turn WiFi on in that place.

Leo: In fact, don't bring a smartphone at all. You don't want to end up on the wall of sheep, that's for sure. But we'll have a report on The New Screen Savers on Saturday from Father Robert. Someday you should go. Have you ever been?

Steve: Nope, never have.

Leo: Be fun, I think, for you. And you'd certainly be lauded as you walked in the door. You could play Spot the Fed, always fun. And, you know, try to decipher the name badge. They always do a kind of arcane name badge that has an encrypted code on it.

Steve: It's got [crosstalk] great spirit.

Leo: You know, it's got great traditions. It really is cool. I've never been, either. Maybe you and I can go next year. Be kind of a fun field trip.

Steve: Some year.

Leo: Some year. I'll bring my BSD box.

Steve: There's always too much to do.

Leo: That's the problem, isn't it.

Steve: And the problem is, you know, I'm able to get four days' worth of work done here and then look at the summary and talk about it.

Leo: Right. And that's true of all conferences nowadays, thanks to the Internet. You don't really need to go to a conference. You can get all the information.

Steve: Yeah, I think the social interaction...

Leo: It's what it's all about, yeah.

Steve: ...is what it's for.

Leo: Yeah. Absolutely. Agree 100%. You'll find Steve at GRC.com. Again, don't know if we'll be doing a Q&A next week. But if you have questions, you can go to GRC.com/feedback or tweet him. He is @SGgrc on Twitter and even accepts DMs, long DMs from - well, don't make it too long. He's got other things to do. If you want SpinRite 6.1 to come out, make it short. You also can go there and get this podcast, this very show. Audio and written transcriptions of the show are available at GRC.com. Don't forget SpinRite, the world's best hard drive maintenance and recovery utility. You know, set up four machines. Buy four copies. Really, go all out. Create a SpinRite factory.

Steve: And then SpinRite all the drives within sight. Someone sent me a tweet this morning saying, hey, "Is it all right if I run SpinRite on multiple VMs?" And I said, "Of course."

Leo: Would it work?

Steve: Absolutely. Run it on every drive that you own. By all means.

Leo: That's good to know. And that's where SQRL is; the Healthy Sleep Formula, the revised, more economical, easier to swallow Healthy Sleep Formula.

Steve: Unfortunately, still completely sold out at the moment.

Leo: Out of stock. I'm hoarding mine. I'm hoarding my niacinamide. I'm not letting that time-release niacinamide out of my sight. Let's see. Oh, just all sorts of stuff. It's a great site to browse because Steve is fairly eclectic in his interests, and you'll find all sorts of tidbits in there. We also have copies of the podcast, audio and video, at our website, TWiT.tv/sn for Security Now!. Or you can always subscribe, and every podcatcher has it.

People sometimes are baffled by the fact that we only have the last 10 shows in the podcast feed. But that's kind of the nature of a podcast feed. If we had all 571 shows, the feed would be hundreds of megabytes, and you really wouldn't want to download hundreds of megabytes every 15 minutes, just to see if there's a new one. So we only put 10 there.

But all 571 episodes are on the website. And I even have some scripts in PowerShell and various other languages for downloading every episode on my blog, LeoLaporte.com/blog. If you want to use some scripting, you can get them all. Or you can go one by one, download them by hand from TWiT.tv/sn. Episode 572 next week. So get listening. Get cracking. And we'll see you next time.

Steve: DEF CON, probably DEF CON follow-up, would be my guess.

Leo: I'm guessing.

Steve: Thanks, Leo.