Listener Feedback #100

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-265.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-265-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 265, recorded September 8, 2010: Your questions, Steve's answers, #100.

It's time for Security Now!, the show that covers your online security and privacy. And the man of the hour, of course, the great Steve Gibson, the guru at GRC.com, creator of SpinRite, the world's finest hard drive maintenance and recovery utility, creator of the first - and namer of spyware - the first antispyware program. And he's done so many great things like ShieldsUP!. And we've been doing this show for - we're in our sixth year now.

Steve Gibson: We are.

Leo: Our 100th Q&A.

Steve: Yeah, this is another milestone episode, Q&A #100.

Leo: Wow.

Steve: Yeah.

Leo: So we started doing Q&As, let's see, that would be four years ago.

Steve: Two years ago?

Leo: Two years ago, that's right.

Steve: Yeah.

Leo: And that's been a nice thing because it gives you a chance to elaborate on some of the things that you bring up in other shows.

Steve: Yes, sometimes our listeners will remind me of something that I actually had in my notes, but I forgot to mention. There's one like that. Sometimes they'll follow up on stuff that I mentioned and give us feedback on their own experiences. We've got a couple like that. Sometimes there's some confusion that's sort of remained, for one reason or another, that I see a number of people having, so I'll just choose one typically confused person, not that they're standing alone, and work on resolving that. So it gives - it sort of closes the loop and gives our listeners some chance to participate.

Leo: I believe it's four years, Steve, if we've done a hundred. And we do 25 a year...

Steve: Oh, right, right, right, right, right.

Leo: We've been doing it a long time.

Steve: That would be four years.

Leo: That would be four years. I beat him in math.

Steve: You got it.

Leo: That's all right, I've already had my coffee. So we will get to our questions. We have 10 good questions from our listeners. And by the way, you can always go to GRC.com/feedback if you've got a question for our next time. But before we get to the questions, we always like to see what's going on in the world of security. And we start with updates to major programs.

Steve: Well, and I didn't pick up on what I guess is an update to Safari.

Leo: Just happened. Just happened.

Steve: Because I'm seeing, yeah, I'm seeing it here, 5.0.2. When I turned my Mac on in order to fire up Skype for this connection, it says, oh, we've got some updates for you.

It's like, whoops. So we'll have news about that next week since I don't know what it is that they did. They just did it, somehow underneath my...

Leo: Sneaky.

Steve: ...my security window. And it looks like iTunes and iWeb and iPhone configuration utility, which probably is nothing other than just new features for the new iPhone stuff, I would imagine.

Leo: Of course iTunes 10 came out last Wednesday with the Apple announcement of the new iPods and their new social network, Ping, which is built into iTunes. But you only turn on your computer every week, so you didn't see that one.

Steve: Exactly. We do have some news, however. We have talked a couple times about the concern over this DLL hijack new problem. Shortly after it became publicly known, it began to get exploited. So we are now seeing exploits in the wild. Last week I talked about Microsoft having a security update with some confusing-seeming settings. So I didn't feel comfortable making a blanket recommendation that people jump on this because it seemed to me like what Microsoft did was a little strange. Well, Microsoft has since added one of their little Fix-it buttons to the same page, making it easy for people to make these registry settings after installing the security update. So a number of users have experimented with it. I have also. It's caused no one any trouble.

So I think at this point, given that Microsoft is not going to be making any fundamental changes to Windows, that is, I don't foresee anything happening except maybe, with the next second Tuesday of the month security updates, they'll just roll this patch in so that this becomes a built-in feature of Windows. Still, I'll be surprised if Microsoft makes this change themselves. If things get bad enough, they may do so. But for security-conscious, aware listeners like those who follow this podcast, given that I've seen nothing negative from it, I do think it's time now to have this change made.

So this is Microsoft's Knowledge Base article 2264107. So support.microsoft.com/kb/2264107 will get you to this page. There you can download a version-specific, for whatever version of Windows you're running, update to Windows, which you have to install first. That is, the registry settings only have new meaning when this update has been installed and is in place. And then there are four different settings which you can choose between. The setting "0" doesn't change anything. The setting of "1" blocks DLLs from loading from the current working directory, if that is set to a WebDAV folder, that is, essentially to a shareable folder, accessible over the Internet, which is probably the largest concern and the biggest threat. The setting of "2" increases the coverage a little bit. It's not only WebDAV, but also any other remotely located shared folders. So that's probably not as big a concern, but it is what Microsoft recommends. It's what many of our listeners have chosen, that is, like the strongest level of protection. It's what I've been running with no negative effects.

So that's what I would recommend people do. And, as always, for a few days after you make this change, sort of keep in mind that this is what you've done. I don't expect to see a side effect because the normal order that Windows searches for DLLs is to first look in the directory from which the application was loaded. Now, there has been some confusion about this, and in fact one of the Q&A questions is on this issue, or talks about this. But there's the directory from which it was loaded is where Windows looks first.

Then the system directory; then the older 16-bit system directory. Then the Windows directory. Then the current working directory.

And that's the confusion, is some people confuse the current working directory with the directory from which the application was loaded. The current working directory is - that's the vulnerability. That's what some applications change, for example, if you were - in the example I was using before with Adobe Photoshop. Apparently, when you open a Photoshop object, for whatever reason, the logic inside Photoshop sets the current working directory sort of on the fly to the directory where that object was loaded. And so the concern is that a bad guy could put, like, a Photoshop object in email with a remote location, something available by WebDAV, for example, over the Internet.

And so what would happen would be, if you attempt to open it, Photoshop is invoked because it's a Photoshop object that you're attempting to open. Photoshop would change the current working directory to that remote location. And also located there would be a DLL named the same as something that Photoshop was then going to load in order to handle the specific details of the object that you're opening. And that's the way that bad guys have now figured out they can sneak executable code into Windows.

The reason this is a serious concern is that it isn't technically a Windows mistake. That is, it's the way Windows has always operated. Some applications, unfortunately thousands of applications, but out of millions, so some applications have adopted the practice, for whatever reason, of changing their working directory to the same location from which they're loading some object, a picture or whatever. So this is not a defect that Microsoft arguably should change, and it's not one they even can change because to change it is too large of a sweeping effect. It would probably break things.

So what Microsoft has done is they've given us a tool to dramatically mitigate the consequences, that is, it's really unlikely that you actually want Photoshop to go get code from some remote location where you have told it you want to open an object. You can imagine that you might want to open an object, some Photoshop drawing or something, from some remote location. But it's hard to see why you would actively want them to go get code from there.

Leo: Yet they built in the capability, so...

Steve: Yes. Actually, it's sort of a side effect. It wasn't ever - it wasn't built in. It just wasn't built out. And so what this does is this does allow you, by setting this registry setting to two - and, by the way, that's not the default. If you just update with this Microsoft security fix, nothing changes. You need to then go and add this registry entry, either yourself, or then you use the Microsoft Fix-it tool to do that for you, which is probably what I would recommend because it's just going to do it for you. And then I don't think anyone will see a negative consequence. I don't think it'll break anything. It seems very unlikely.

And again, since what Adobe and all these other people probably expect is the current working directory is the same as the execution directory, well, Windows always looks in the directory where the application loaded from first. And that's typically where all of sort of the add-on DLLs are going to be clustered around in the same directory as where the application ran from. Which is why Windows probably finds this in the first place it looks. And so, anyway, that's the update on this. I think at this point, given that we're now seeing exploits in the wild, as was expected, that nothing is going to come along in the future to fix this for us; that this is probably what everyone ought to do at this point is

take some responsibility for this.

And now the problem is, I wonder how the rest of the world is going to find out about this. I mean, we cover it here on Security Now!. We've known about it from the beginning. But this is important enough that - I don't know what Microsoft's going to do. Maybe, if they feel comfortable enough with this change, they'll just end up adding it into one of the monthly security updates. That is, they'll not only add this patch, but also set the registry for people who don't already have it set, just as a preventative. Although you can imagine how reticent Microsoft's going to be to change the way Windows works, because that's what this does. This changes longstanding, 20-year-old behavior from Windows 3.1 or probably 2.0. So, but I think it's important to do at this point.

Leo: Okay.

Steve: And then the other bit of news is we have now seen the appearance of the first robust, successful, 64-bit Windows rootkit. There is a rootkit called TDL3 which used to be called Alureon, A-l-u-r-e-o-n. And we talked about this some months ago, I guess earlier this year, because there was - after one of Microsoft's standard second-Tuesday-of-the-month patches, a bunch of people were finding that their systems were crashing. And I'm sure I remember talking about it. It's like, okay, what's going on?

Well, what we discovered was that those systems that were crashing had an unknown, previously undetected rootkit already resident in them. So actually it was a good thing that these things crashed because what happened was the rootkit was assuming absolute locations of Microsoft's code in specific components of Windows. With this particular security update that Microsoft just pushed out innocently, those particular locations changed. Well, the rootkit didn't realize that Windows had been updated. It was still using the previous locations, which no longer functioned, and brought the whole system to its knees.

So then what Microsoft did was, once they figured out what was going on, that these systems were crashing because of this updated Windows technology that was interacting with rootkits, they basically pulled back that update from their updates because they didn't want to keep crashing people's machines. They added specific rootkit technology to this particular security patch, and then pushed it back out so that it would look for this rootkit and then be smart about removing the rootkit, or not installing itself if the rootkit was present.

So this rootkit has been around for a while, but it has not been able to infect 64-bit Windows installations. And this is significant, that is, the fact that it's now able to do that, because - and we've talked about this also in the past. Microsoft really threw down the gauntlet with 64-bit Windows; whereas moving the security of 32-bit Windows dramatically ahead has always been a problem because of it needing - increasing security means breaking things. So, for example, kernel-level hooking. Well, firewalls, personal firewalls, antimalware, antivirus utilities, many things have been making changes - benign, beneficial to the user changes - to the kernel code in order to hook themselves into Windows.

As Microsoft has been moving forward from XP, where this is possible, into Vista, and then into Windows 7, they've been incrementally sort of tightening the screws on these practices, notifying people that this is going to be going away in the future, so stop doing it. And also, and importantly, giving people alternative means. Which is, it's not that people wanted to be hooking the kernel, it's that there was no choice. Microsoft hadn't

published, for example, a clean way of getting at low-level packet traffic in the Internet connection. So vendors had no choice but to go in and modify deep level components of Windows in order to have an opportunity, for example, to scan incoming Internet traffic before it reached the more tender underbelly of Windows, where it could start doing some damage.

So Microsoft has been adding the necessary features in order to help people play by the rules. Well, they did something very different with the 64-bit Windows. Since there wasn't any Windows 64 prior to its relatively recent introduction, Microsoft said: From day one there will be no kernel patching; from day one we're going to force all drivers to be digitally signed. And so what this did was, this made Windows 64, the x64 version of Windows, much more secure from its first release because they had painfully learned all of these lessons as they have been moving their 32-bit product forward, wishing they could make it more secure but unable to because of the legacy that they would be breaking if they did. With 64 bits they said we're starting from the beginning, since there is no legacy there. Everything has to be digitally signed, and absolutely no patching.

And we've talked about the so-called "kernel patch protection," which Microsoft calls PatchGuard, which prevents any of this hooking of the kernel. Well, those two things, the enforcement, absolute enforcement of signed drivers and PatchGuard, without exception, I mean, you can't turn it off, you can't get around it, it's just always been there, hardcoded, deeply wired into 64-bit Windows. It's made those systems invulnerable to rootkits. Until now. And the way the rootkit got around this is a lesson in security for us. It hooks the master boot record. It hooks the boot sector of the hard drive and installs its own code in some location on the hard drive that allows it to execute from the first moment the system starts to boot, essentially getting into the system before Windows.

Now, we've seen this before in, again, in a benign way. That's what TrueCrypt does in order to provide whole-drive encryption. In order to encrypt the Windows boot drive, it has to be able to decrypt the first data that comes from the drive. Well, the only way to do that is if TrueCrypt is installed prior to Windows starting to read itself from the drive. So it's a great example of a clever approach which can be used benignly, but unfortunately can also be hooked by a rootkit in order to raise some havoc, which is what this thing is doing. So that exists now. And I guess what we're going to then see is some sort of round of protection of some sort. It'll be interesting to see what Microsoft does because, I mean, when you get in before Windows, you get in before Windows. You're running before Windows has any chance to do anything.

So I'm not sure what the consequence will be. There are BIOSes which allow you to physically protect the boot sector from changes because boot sector viruses have been around for as long as hard drives have been. The concept of a virus modifying the boot sector is not new, and consequently some BIOSes prevent that, although that's even a relatively weak protection because the BIOS used to protect it by looking at your use of the BIOS for accessing the hard drive. And now contemporary systems don't go through the BIOS. They access the hard drive directly. So it's not clear to me that it's even feasible to ask the BIOS to protect us. I don't know who's going to protect us. So it'll be interesting. Maybe the drivers can change so that they will refuse to write to the drive. Then the bad guys will write to the hardware directly, circumventing the driver. I mean, this is a fundamental problem in the architecture of our systems. I don't know what we're going to do.

Leo: Hmm.

Steve: Not good.

Leo: Yeah. Yeah [sighing].

Steve: So I did have a fun story to share. Actually it was when I was going through my mailbag, the security news for pulling the Q&A today, something caught my eye, a subject line that said "SpinRite saves the pepperoni."

Leo: Okay.

Steve: Which is not something you hear every day.

Leo: Is there a pizza involved?

Steve: Well, Doug J. is how we'll refer to him, although he does say Johnson here, so Doug Johnson in Provo, Utah wrote. He said, "Steve, yesterday the two other owners of a company that creates restaurant management software and I had just finished the last installation and training of our system in a new market for a national pizza chain." So this is a big, nationwide pizza chain. "In celebration, we had just sat down to a nice dinner and placed our order when we got a phone call from a very horrified IT director of the restaurant's parent company. On her way out of town she was stopping by all the restaurants to connect the USB cables of the UPS, the Uninterruptible Power Supplies, to all the servers, and the previous 12 had worked perfectly. But when she plugged in the last one, No. 13, the server crashed and refused to come back up. It wouldn't even attempt to power back on. And this was just minutes before their dinner rush. They were completely unable to process orders electronically or accept credit card payments, right as they were about to get bombarded with customers.

"After we hurried to finish our dinner, we rushed to the restaurant. When we arrived, the server indeed would not even power on. After figuring that there might be a cabling problem, I disconnected all the cables from the box, gave it a minute to rest, plugged in the power and network cables, and pushed the power button. It powered on and appeared to try to boot. But it wasn't visible on the network after the hard drive activity had settled down. So we found a monitor and keyboard and plugged them in, only to discover that it was blue-screening on boot. Neither the last known good configuration option nor the safe boot mode options would allow it to come up. Something was very wrong. Because the setup was still new, we hadn't yet provisioned our offsite backup server for them. So the only copy of their order, financial, and employee data was on that box. We had to get it back."

Leo: Oh, man. And they didn't have a backup.

Steve: Nope, hadn't gotten to it.

Leo: How, you know, for crying out loud.

Steve: They were going to do it tomorrow.

Leo: For crying out loud. All right. Go ahead.

Steve: "At this point I figured I was dealing with a corrupted installation of Windows, but I didn't have any installation media with me to attempt a repair. All any of us had was our master server image file - which, if restored, would wipe out all their data. I had to come up with another solution. I did have my copy of SpinRite. I popped it in, set it to Level 2, and let it run. Just a couple of minutes into the scan it found a bad sector on the hard drive and started the process of data recovery. It took a couple minutes...." So that would have been the DynaStat actual physical sector data recovery procedure that SpinRite drops into.

"It took a couple of minutes, but SpinRite was able to recover every bit of data from the bad sector. After SpinRite finished, I rebooted the machine, and after running CHKDSK it came up right as if nothing had ever happened. No data was lost. The owners of my company were astonished that I was able to get the server back up without reinstalling Windows. And the company's IT director was very relieved that her actions hadn't led to the store's data being lost."

Leo: I'd fire her right away.

Steve: He says, "SpinRite absolutely saved the day. Thank you for creating an absolutely indispensible product." And, Doug, thank you for the report. Sure appreciate it.

Leo: Why a business would have financial data unbacked up for even four seconds...

Steve: Well, they were going to get to that tomorrow.

Leo: Yeah, tomorrow. It's always tomorrow, isn't it. All right. Let's get to our questions. Are you ready, sir?

Steve: Let's do it.

Leo: Let's do it. Question #1 comes to us from Tom Sullivan in Indiana, feedback from our DLL Hell episode: Steve, there's nothing particularly difficult about the new Microsoft patch. You install the patch which enables the use of new registry - we're talking about, is this the LNK issue?

Steve: No, this is the DLL sequence deal that we talked about at the top of the show, where Windows is loading from a remote directory when it really shouldn't be.

Leo: Got it. So he says you install the patch. That enables the use of the new

registry entries to control the DLL search path system. It's in effect changing the path; right?

Steve: Right.

Leo: Without the patch, the registry entry is not recognized. After the system is patched, you can globally restrict DLL searching; you can also restrict it by application, by using appropriate registry entries. So that would protect you against the Photoshop flaw?

Steve: Exactly.

Leo: Few programs will actually need to get a DLL from the current directory, as opposed to their own execution directory. For any that do, on startup you simply set the shortcut, the .LNK file, to start in the directory wherein the program's EXE is stored. So that's simple. I mean, it's going to take a little IT setup here. However, it's clear that if a program like Photoshop changes its current directory, then it won't need to find DLLs from there. Overall, this seems like a safe patch. I've set mine to 2 globally, that's the most secure setting, and have yet to have any problems with XP or Windows 7, says Tom.

Steve: Yeah. There actually is - I didn't mention this at the top of the show. There is a more restrictive setting than 2. You can set it to hex FFFFFFFF, that is, all Fs. And Microsoft documents this on that Knowledge Base page. What that does is, whereas for example the setting of 2 restricts the loading of DLLs so that Windows will not load them from any remote folder, either WebDAV or a remotely located shared folder, you can go one step further with this all Fs in hex, which technically is negative one in signed two's complement binary math, which, again, Microsoft documents. And what that does is absolutely removes the current working directory from the sequence altogether so that nothing will load from the current working directory. And so that's a little stronger than 2. I think since the danger is really only from remotely located code, you could argue that, if something has already got the code on your system and has figured out how to run it, well, you're...

Leo: You're in trouble already.

Steve: ...probably in trouble anyway. So I did want to encourage, again, people from - encourage them to install this update, set it to 2. I don't think anyone will have any trouble. And as Tom mentions, and I forgot to mention this also, you can perform a per-program override. So if you learned after a few days that something that you were using did deliberately need to load code from a remote location, then you're able to have a global setting which is used by default, but then per-application tweaks. So you could, like, set Windows back to the normal behavior only for specific applications. So Microsoft gave us a good tool, but we do have to use it. And I don't think it's going to be used by default, so it's important.

Leo: Do you recommend FFFFFFFF?

Steve: I used 2. I'm tempted now to switch to FFFFFFFF.

Leo: You can never have too many Fs.

Steve: And just see if there's any problems. See, remember that it's so far down in the search hierarchy, that is, this current working directory is the next to the last thing where Windows looks for anything. So I'd be surprised if anything ever gets down there anyway. So I would say it's probably really safe to set it to all Fs and just really, really be robust. But 2 is probably good enough.

Leo: Steve in Florida says, "Solved!" Remapping router IP to "Linksys" fixed cert mismatch per SN-263. And you may remember his question, but I'll repeat: Steve, thanks for featuring my question about the certificate mismatch between my router admin page's gateway address, which of course is 192.168.1.1, and the certificate, which was issued to Linksys. And so he saw the warning that says there's a mismatch. The certificate is indeed self-signed by Cisco-Linksys LLC, but issued simply to "Linksys" as shown on the mismatch warning. I did as you suggested. I went into the hosts file, and I just mapped "Linksys" to 192.168.1.1. And it worked like a charm. Now I just type - this is actually kind of handy.

Steve: Yes, it is really handy.

Leo: Https:// - that's why he's getting a certificate mismatch because he's using secure links - //linksys. And since he's got the hosts file, boom, it pops up, no more mismatch warning. I get the password, and I'm in. You don't have to even remember 192.168 and all that.

Steve: Whatever that is.

Leo: Whatever. See, I always - well, the problem is sometimes on some routers it's 0.1; on some routers it's 1.1; on some it's 0.100. So sometimes you forget, if you use multiple routers. Fortunately the URL "Linksys" by itself doesn't go anywhere. However, if you just type "Linksys.com" or even "https://linksys.com," you do get Cisco's home page, which is what you want. There's no conflict. If you leave off the .com, you get your router. As for Leo and your question about the connection itself, yes, of course the laptop is WPA2-secured with a strong password. He uses HTTPS partly defense in depth, partly because sometimes when you're messing around with things in there you may have to restore the default settings or disable encryption momentarily. And it would be just my luck - this is the kind of listener I like, paranoid - just my luck to have a bad guy listening in at that very moment. TNO, Trust No One.

So I recommend that all super-paranoid people set their router admin pages to

accept secure connections only, now that you've solved the certificate mismatch problem. You might have to give a quick refresher on how to do the hosts remapping. Anyway, I'd never have thought of that idea. Super solution, and thanks. That's interesting. So essentially you're telling the browser that 192.168.1.1 is Linksys, and then it matches the certificate.

Steve: Well, you're actually telling Windows.

Leo: Windows.

Steve: What the hosts file does is it is traditionally, in any Internet-connected machine, the first place that the computer looks when you type in a domain name is in your system's own local hosts file. Because once upon a time, before DNS existed, and there were seven IPs in the world, when there were just a few educational .edu - or actually there wasn't .edu. But you had - because that's DNS. When you only had maybe, okay, maybe a hundred, there was actually a hosts file on each of those hundred different machines on what would eventually grow into the Internet.

Leo: That was DNS. That's how it worked.

Steve: Exactly. It was just one local, one non-hierarchical...

Leo: That's hysterical.

Steve: ...simple file where you had machine names that matched the IP addresses so that people could use those instead of IP addresses. Well, that's survived to this day. And we've talked about other uses for the hosts file where you would want to, for example, intercept your computer, if you, for example, put DoubleClick.net in there and had it point to 0000 or something, then your computer would be unable to go to DoubleClick.net, the real one, because it would never ask anyone else for the IP. Your hosts file provides it first.

So what occurred to me actually on the fly while we were talking about this two episodes ago was, Steve's question was, hey, I get this certificate mismatch error because I want to talk to my own router. Even though it's in the same room with him, he wants to talk to it over WiFi and over SSL. And we were saying, wait a minute, doesn't he have WPA encryption? And he says, yes, I do. But what if I'm reconfiguring things, and I want to drop my WiFi encryption? I'd still like my connection to the router to be encrypted. And so he was saying the problem was that he had to type in https://192.168.1.1. Well, the browser would complain because he was creating a secure connection because the certificate that Linksys has is the word "Linksys." So the browser sees that what he entered in the address bar doesn't match their certificate.

And of course this also comes back to STS, Strict Transport Security, that we've been talking about because, as soon as you start turning on Strict Transport Security, certificate mismatches are no longer click-aroundable. You can't say, oh, yes, that's fine, I know that it's mismatching. Strict Transport Security will absolutely tolerate no

variations from everything working. So what occurred to me was, if you made a mapping, essentially an entry in the hosts file, where you said Linksys is at 192.168.1.1 - not Linksys.com, just the word "Linksys" - then now using your browser, using SSL, https://, you put in "Linksys" and hit Enter. Your computer looks in the hosts file first, finds the IP, connects to it, and then the browser now sees that the certificate name, Linksys, matches what you entered in the URL field, no more error, and you're connected. So it ends up being a really cool little hack.

Leo: A lot of routers come with a URL. In other words, I can't remember what it is exactly, but I've seen routers before where you don't have to enter the number. You type - I think it's Linksys that does that.

Steve: Well, and the way they could do that is if the router was providing DNS. That is, if the router was providing DNS or even intercepting DNS, remember that the router is where your computer traffic has to go after - if it doesn't find an entry in the hosts file, then your system says, oh, I've got make a DNS query. So it formats a DNS query and sends it up the wire to whatever your DNS is. Many routers now have assigned themselves as - and we've talked about this before. They've assigned themselves as your network's DNS server. So that allows them to say, oh, he's asking a special domain name...

[Talking simultaneously]

Leo: ...that page, yeah.

Steve: Exactly. And so then the router is able to send back its own IP, which is kind of cool because it knows its own IP. See, this little hack that we just mentioned would break if you changed the router's gateway IP because then it would no longer be at 1.1. Yet your hosts file would say - it would still say 1.1. So you'd have to edit the hosts file entry also. So having the router sort of provide its own little funky DNS allows you to always access it transparently, which is also nice.

Leo: Chatroom is telling me that Netgear does that with Routerlogin.net. I don't think that's a good way to do it because what if somebody wanted Routerlogin.net?

[Talking simultaneously]

Steve: That domain name is gone; right.

Leo: Yeah, it's gone. You could never get there.

Steve: You could never get to the real one because the router would intercept it.

Leo: Presumably Netgear registered that domain and keeps it around.

Steve: We would hope so.

Leo: We would hope so. If not, then I'm sorry that you own Routerlogin.net. I don't - I think that's not a good way to do it. I think the hosts file is a much better way to do it. But obviously they can't expect people to edit their hosts file.

Steve: Well, and the hosts file is universal. I mean, all of our listeners could do it if they wanted to.

Leo: You have to do it on every machine that you were going to log in from.

Steve: Actually there's a way that a hosts file can have a pointer to other hosts files.

Leo: Ah.

Steve: That provision exists. So you could have one master hosts file for your whole network, and the other systems all point to it, so you only have to change it in one location.

Leo: So have a canonical hosts file on one machine that's never turned off.

Steve: Right.

Leo: And that machine's always got it.

Steve: And everybody else's hosts file goes there in order to look up things.

Leo: What a good idea. What a good idea. Question #3, Brandon Ivy in San Jose, California wonders whether his college is spying on him. Hi, Steve. In order to gain Internet access at the college I attend, Cal Poly SLO, the first requirement is every student has to first install a custom certificate. I remember years back you were talking about SSL and the few ways that companies could spy, if they wanted to, on their employees. Is this certificate a sure sign that the higher-ups are seeing everyone's passwords to PayPal, banks, and the like? Smart kid, Brandon.

Steve: Yeah. I would imagine that it's being done for a benign reason because Cal Poly probably wants to be able to provide filtering technology. They want to be able to prevent people from going to sites using their network that Cal Poly policy says they shouldn't. On the other hand, this absolutely does mean that your browser is connecting to Cal Poly, and your traffic is being decrypted and then reencrypted by - hopefully reencrypted by Cal Poly when it goes back out.

You can absolutely verify that by looking at the certificate when you make a secure

connection, make a secure connection to anywhere - Amazon.com, GRC.com, whatever. Make a secure connection, and then look at your browser at that web page's, the web page that's delivered, look at its security credentials and see whether you can see the chain goes back to - like GRC is signed by VeriSign. See if you see that, or see if GRC is signed by Cal Poly, which would indicate that Cal Poly generated a certificate on the fly for GRC.com in order to satisfy this requirement we were just talking about of the certificate matching what's up in the URL bar. If so, that confirms that you actually don't have SSL connections directly to the endpoint; that because of this policy, that I'm sure it's in the fine print of your student agreement somewhere, they said this is what we're going to do. In order to use our network, you've got to accept our certificate. And the consequences are that.

Leo: Would you have to always have a custom certificate to intercept SSL traffic like that?

Steve: Yeah. In order not to - if you didn't generate a custom certificate on the fly, then any site you went to would complain about a certificate mismatch. So you're constantly having to be clicking through that. And so this fakes out the certificate the browser's expecting by generating one on the fly. And our next question bears on this, too.

Leo: Well, let's go to it. Question #4, David in Utah noticed that Net Nanny, which is filtering software to protect your kids, installs itself as - drum roll, please - a root certificate authority. We got the Net Nanny because my wife wants to help the whole family not be exposed to porn. I think that's a euphemism for him, but okay. I didn't realize it for a while, but after installing it I discovered that it is the one issuing all the certs - oh, geez, Louise...

Steve: Uh-huh.

Leo: ...whenever I connect to my bank or https://grc.com. I recall a podcast saying this was a way for employers to spy on their employees. Shouldn't I be worried that ContentWatch, Inc., the creators of Net Nanny, can see all my bank data and probably LastPass passwords? Wow.

Steve: Yeah. Now, I wouldn't say that David should be worried, but David should know that it's possible. And that's what this means, is with Net Nanny or anything else similar which requires that you install it as a certificate authority in your browser - that's exactly what San Luis did for our prior question. What that means is that your browser will now trust, just as it trusts certificates signed by VeriSign and GoDaddy and Equifax and everybody, all the other hundreds of certificate authorities, it will now trust certificates signed by ContentWatch, Inc., the publishers of Net Nanny.

Well, the consequence of that, if you've got this Net Nanny software installed wherever it goes, in the computer or in some central location anywhere, that means that Net Nanny is, exactly as is San Luis Obispo's network, Cal Poly's network, that Net Nanny is, on the fly, generating certificates, handing them to your browser. And the browser says, huh, this thing was signed by ContentWatch. Do we trust them? So looks in its certificate store, sees that, sure enough, look at that, ContentWatch, Inc., is a recognized certificate authority for this browser, signed the certificate, so we trust it.

And so again, once again, the reason Net Nanny is doing this, the reason that software is doing it, is so that it can filter SSL connections, that is, it can decrypt it, scan it for, apparently in this case, pornography, and then reencrypt it when it goes outside the network. So the only way to provide that service is this on-the-fly decryption of SSL connections. Unfortunately, that's a lot of responsibility for ContentWatch to take, and something that users need to be aware of.

Leo: [Subdued exclamation]

Steve: Yeah.

Leo: There's probably a legitimate reason - well, legitimate. It's probably part of the functionality that it does that. I mean, I would imagine in order to filter content and to keep an eye on content it probably needs to do some pretty draconian things.

Steve: Yeah, I would wonder if there wasn't a way of backing it off of that. That is, like, say, just uncheck the "I want you to filter my SSL connections" box from Net Nanny. I've never seen Net Nanny. I don't know whether it offers that option. But, boy...

Leo: That's one of the reasons I like OpenDNS. It's not nearly so intrusive. You don't modify your system at all except for changing your DNS entries to point to OpenDNS. There's no certificate or anything like that; right?

Steve: Right. Right.

Leo: And it works.

Steve: And in fact I've just spent the last week adding DNS rebinding detection to GRC's DNS Benchmark, which will be out soon.

Leo: Excellent. So we'll know as that happens.

Steve: And, yes, and OpenDNS is doing that. And we now are able to detect that on the fly, so it's very cool.

Leo: That's not a bad thing; right?

Steve: No, it's a good thing. It's providing that.

Leo: Detecting it, yeah.

Steve: Nice feature.

Leo: Yeah. Philip Le Riche suggests that we shouldn't blame von Neumann. Oh, this goes back to something, and I wish I could - I never did find the article that says that really the security flaws in the world began because of the nature of a von Neumann machine. Steve, I think you should put Leo right on the von Neumann architecture as the root of all evil. Without it, it would be impossible to write a general purpose operating system. After all, how could you ever compile a program or load it into memory for execution except by treating it temporarily as data? The Manchester architecture, with its separate instruction and data address spaces, is fine for single-task computers like microcontrollers and embedded systems, but isn't much use for general purpose computing.

The big mistake was probably not von Neumann's but Microsoft's. The x86 architecture does provide you with a virtual Manchester architecture in user space, having separate code, data and stack segments and no means of writing to the code segment, or executing instructions from the data or stack segments. But as far as I can tell, the execution model used by Windows throws that advantage away by making the three segments coincident. Regards, Philip. Is that true? You couldn't write an OS in the Manchester architecture?

Steve: No. And I would take issue, I guess. I don't think that you were wrong, Leo, to suggest that a Manchester architecture is arguably more secure. I would say that it certainly would not prevent all the problems. Okay, so just to review a little bit, the question is, does data and instructions come from a shared memory space where instructions are able to point to data, and since data and instructions occupy the same space, the instructions could be pointing at instructions. So, for example, when you load - and we run across instances where this is happening, I mean, security problems, all the time. You load data from the Internet, which overflows a buffer, and that data is executed by mistake, and the bad guys have cleverly designed the data to be executable. So, I mean, this is the buffer overflow problem that Steve Ballmer famously steams around about, screaming why can't we fix these problems? Why are we still having this after all these years? That's the problem.

Now, the Manchester architecture, physically it architecturally separates instructions from data so that, even if the instruction referred to the same address as an instruction, it would actually be referring to a separate bank of memory, physically separate. And we discussed this once where there was the design of some voting machines, which I was really pleased to say, or to see, in their architecture they were using a Manchester architecture, that is to say, instructions and data were separate. The only reason they did that was the designers, the architects, recognized that was more secure. And I would say absolutely yes, that's more secure. Does it solve the world's security problems? Absolutely not.

It is the case, for example, that you could still write an operating system in it, with that architecture. You could compile a program, save it to disk, and then load it into the instruction side. So while you were compiling the program, you'd be building the program over on the data side. But then inherently you need to execute it. So you just save it to the disk and then load it over into the instruction side. So everything can still be done with that architecture. I think it's probably, in retrospect, too bad that we have this combined architecture, except it's so convenient. I mean, it makes so many things easy. The problem is it makes things, you could argue, too easy. It makes security vulnerabilities just trivial to have happen by mistake.

So if, turning back the clocks, we had evolved our systems with this awareness today, so that they were always separate address and data spaces, probably more secure. And to the Microsoft and stacks, well, the problem is this data execution protection, DEP, that we talk about, it's something Microsoft would like to have always done, always had enforced, except that the hardware architecture, the Intel hardware architecture, didn't enforce it, didn't offer that DEP, the so-called NX, the no-execution bit, until relatively recently. And then, when it came along, it turns out that there is code which, either deliberately or just through laziness, will not function with that DEP bit, with data execution protection enforced all the time.

So, again, it's one of these things where, well, due to history we really can't enforce that, even if we wanted to. We're moving forward. We're trying to encourage people to clean up their code so that they can have DEP enabled because it would prevent some class of misconduct, some cases of misconduct. But again, it's also not the cure-all answer. So more security is better.

Leo: And I'm not going to take any credit for this idea. I'm not smart enough to have come up with this. It came from a blog post I read by Charles Stross. He's at Antipope.org. It's a great blog post, "Where we went wrong." And he calls it the "Harvard architecture." He says, one, the von Neumann triumphed over the Harvard architecture. I think Alan - wasn't Alan Turing championing the Harvard architecture? I think he was.

Steve: I don't remember; but you're right, Harvard is a term I'd normally use for it, too.

Leo: He also says, and we've talked about this before, number two, string handling in C uses null-terminated strings rather than pointer delimited-strings, which is source of a lot of errors by just going off the end of the string.

Steve: Right. Pascal strings, famously, had a length byte as the first byte of the string, which was very convenient except it limited you to 255 characters for the string. So that was a problem.

Leo: Right. Charlie says C, the programming language, "will not only let you shoot yourself in the foot, it will hand you a new magazine when you run out of bullets." And, three, TCP/IP lacks encryption at the IP packet level. And he said you can blame the NSA in the early '80s for this. They wanted a fundamentally insecure system.

Steve: No. No, no, no.

Leo: No?

Steve: He's wrong on that one. I know the history too well. That's just - it just wasn't considered in the beginning.

Leo: Nobody thought about it.

Steve: Yeah.

Leo: He also said DNS lacks authentication, which is a subcategory, but shouldn't be underestimated. And he gives as number four the World Wide Web, which was designed for academics in a research environment, not by and for banks or somebody who wanted to be secure. It's a good post. And finally, six, Microsoft. "Sorry," he says, "let me rephrase that. Bloody Microsoft." So it's worth reading. I can't claim any knowledge in this area. I just thought it was a very interesting blog post, and that's why I posed the question to you a couple episodes ago. Thank you, Philip, though, for the question.

Number 6, an anonymous listener in Washington State, he's worried about STS (Strict Transport Security) token information leakage. Man, when you're paranoid, there's lots to worry about, isn't there. Steve, I enjoyed your episode on STS and am pleased with the new security offered by participating websites. But I have a couple of concerns. You mentioned that the STS tokens do not transmit anything. However, anyone who has access to the machine locally could view a user's list of STS tokens and deduce from the very recent tokens which websites a user visited. This could also be used to determine things such as what bank a person users in order to target an attack. While a minor issue, is there a way to conceal from others which STS tokens are present on your machine? Further, in your last Q&A somebody mentioned a "self-denial-of-service attack" - only for Lent. What prevents a random website from issuing a forged token for another website, thereby doing a denial of service to that domain? Thanks for the great show. So, Steve, what do you say?

Steve: Okay. I chose this question because it's an issue of security that comes up from time to time. And the issue is we need to be clear, it's super important to be clear about what different security measures are meant to do. So Strict Transport Security, which we discussed a couple episodes ago, its design is to force secure connections so that at no point in the connection between a user and a remote location is there an opportunity for non-SSL dialogue which could leak cookies or logon credentials, username and passwords, anything that might be going in the clear.

What it doesn't try to do is the things this listener is asking. That is, it's like, yes, if you look at the STS tokens in the .db file which Firefox maintains, you'll see all the domains that have issued that browser STS tokens. In the same way, if you look at the browser's cookies, you'll see pretty much everywhere the person ever went. So cookies are a privacy problem when viewed from the end of the machine, the local browser, and so is STS. I mean, there's some more information there, token information leakage, as he put it. But that's not what this is trying to solve.

So the reason this is important is it sort of comes up from time to time where there's a confusion about a specific security measure and what it's designed to do. And security professionals are careful, and it's crucial to be really careful, to circumscribe what it is that a security measure does and make sure that's what you expect, it's what you need, and that you're not asking it to do something it wasn't designed to do.

Leo: Don't overload it.

Steve: Exactly. So STS absolutely is successful at making sure that no connection is made to an STS-enabled site that isn't over SSL, and that you can't even bypass it. Except, unfortunately, if you're using Net Nanny, in which case the STS is out the window because you're not going to have any security problems that STS can, that your browser can detect. So that's what it does. It doesn't do anything more. So again, I see these kinds of questions all the time. I wanted to just take sort of a moment, not to pick on this anonymous listener, but it's why I thought it was nice that he was anonymous, to say you always want to be careful about what it is you're asking these security measures to do because they can't do more than what they do. If they're designed correctly, they do what they're designed to do, perfectly. And that's the most we can hope for.

Leo: Right. And it might be a mistake to have too much built in. I mean, I think we've all learned that the simpler tool that does the one thing well is probably the way to go.

Steve: Yes.

Leo: Question #7, Craig in California, he's worried about his MAC address: Hi, guys. I've been listening to Security Now! for years, love it. I particularly enjoyed the recent episode on privacy leakage. But you didn't talk about MAC addresses, which, as I understand them, are unique, generally unchanging. So why isn't that a big concern or factor of privacy or security? And I guess I would add the adjunct, well, who can see and who knows what your MAC address is?

Steve: Correct. And that is why it's not a concern. We've discussed this a couple times. Although it's funny, the reason this caught my attention was that it turns out that, over Windows filesharing, Microsoft gratuitously sends your MAC address. That is, as part of the protocol.

Leo: Of course they do. Of course they do.

Steve: I know. And I built that into Security Now!. I mean, into Security Now!. Into ShieldsUP!. From the very beginning, when people...

Leo: Yeah, because you'd tell me my MAC address sometimes, yeah, yeah.

Steve: Uh-huh. Exactly. And so there is a privacy concern for people using Windows and Windows filesharing, if that's turned on, because if I can see it across the Internet, so can everybody else.

Leo: Right.

Steve: So there is that concern. Although with filesharing closed down and firewalls running now, it should not be a problem. And in general, remember that the MAC address is used only within a LAN because a MAC address is for Ethernet, not for IP. IP runs on top of Ethernet. So IP packets jump from one Ethernet network to another as they go from router to router across the Internet. IP packets are briefly wrapped in different packets with a MAC address in order to go to the next interface, the next NIC, the Network Interface Card, in an Ethernet network. Then that IP packet is unwrapped from that. It goes through the router, gets rewrapped in a different MAC address to go out to a different network. So normally the user's MAC address never survives past his own local area network unless in the weird case of Windows filesharing, which for some reason provides that just because, why not?

Leo: Sounds like a Perry Mason novel: "The Weird Case of Windows Filesharing." Question #8, Ronald Wilson in Upstate New York had a thought about "side-channel fingerprinting." Ooh, that was a great topic from last week. If you didn't hear that episode, by the way, great, just fascinating episode. I'd like a utility, says Ronald, that alerts me when a website polls my browser for data, if that's even possible. For example, hey, PayPal just asked for your screen resolution, your time zone, your current system time. Is something like that possible? Perhaps even a browser info firewall that would block the information from being returned, or randomize it?

Steve: Well, last week's episode, as you mentioned, Leo, which caught a lot of people's attention, generated a ton of email. And there were a lot of questions like, well, what can I do? I mean, you sort of, like, painted a gloom-doom scenario. Relative to Ron's question, the problem is that there are legitimate reasons for, for example, JavaScript reading your screen resolution. You can imagine some advanced code that Google would produce, because they love JavaScript to the degree they do, where if it saw that you were on, like, a tiny screen, like a mobile browser, or just like a restricted screen, more like a laptop instead of a desktop, the server might have a legitimate reason for knowing, not to track you in any reason, but in order to customize the content they're delivering for the size of screen that you have.

And in fact, once upon a time - you'll remember this, Leo - the user-agent field used to contain all that. The original, early days browsers would embed, like, your screen resolution as part of the user-agent information because they just figured, hey, maybe the server would like to give different pages, depending upon how large the user's screen was, sort of change the resolution or change just the amount of content on the page that was delivered. So, and you can imagine time zone could be used nicely. For example, a server could deliberately, like say for an auction, could show you when the auction is closing in your time zone, rather than always in the Pacific time zone, like eBay does, or they just use UTC - or, no, I think eBay just always shows it in Pacific time. But it'd be nice if it showed it in your time zone. Well, in order to do that, it needs to know what your time zone is. So again, there's an arguably valid reason for a server knowing that, too.

So it's not that this stuff is all spy information, it's just that it's stuff that you could reasonably make available, but it does provide some differentiation between you and somebody who's otherwise identical to you in a different time zone. So it provides some more info. I did have a thought about - for all the people that wrote in and asked what do we do about this. And Leo, this is the kind of thing you would suggest. And that is, any of these CD-based bootable CDs, it's going to always be the same. So everyone with some Ubuntu Linux distro that they boot from CD, it's going to fire up and load, and there's a web browser.

Leo: Really.

Steve: And whatever it is, it's all self-contained. Every single one of them looks the same.

Leo: So there's no individual information that the browser knows that is unique.

Steve: Correct.

Leo: Because you don't have any cookies on there yet.

Steve: Yeah. You know, your system clock is going to be you. But that's not...

Leo: That's probably not enough in itself.

Steve: ...all the other stuff. And so my point is that everyone using that particular build, there may be some build information, there may be some browser version information that changes. But everyone using that build is going to look very much the same. And so if people really were concerned about this, that's one way just to be very anonymous. And of course it's a way to be very secure, too, because you have a self-contained environment which is difficult to escape from.

Leo: And unmodifiable.

Steve: Correct.

Leo: Every time you boot, clean slate.

Steve: Correct.

Leo: I think more and more people should be doing that. And we did hear a rumor, we don't know yet, but that Windows 8 when it comes out will have this kind of feature, this kind of restart and reset each time, as an option.

Steve: Interesting. They have a [indiscernible].

Leo: It's like SteadyState.

Steve: SteadyState, exactly.

Leo: Our last question...

Steve: Is not really a question.

Leo: Just a statement. Security Now! listener Richard says: Have you ever looked at OAuth? Have you ever looked, Steve, have you really looked at OAuth in the eye? It'd be a great topic for Security Now!. By the way, great show. Keep up the good work. I'm a supporter of your work and SpinRite owner with no disaster tales to tell. Which is a good thing. What is OAuth?

Steve: Well, it's our topic for next week.

Leo: Oh, how exciting.

Steve: It is an emerging standard which is really interesting because it's had a little bit of a spotty past. There were a couple mistakes made early on. But it is a very intriguing means for providing authentication to third parties where it's not necessary for you to disclose your credentials to them, yet they can still authenticate on your behalf. This has been in the news in the last couple of weeks because Twitter officially stated that they're going to be requiring all of the Twitter apps, like Seesmic and Twitter Deck, or TweetDeck and all these different things, they're standardizing on OAuth.

Leo: They flipped that switch August 31st.

Steve: Yeah.

Leo: If you don't use OAuth [sound effect]. Which is right because we were giving some third party our login credentials.

Steve: Yes. And that's just not a safe thing to do. They could perform any mischief they wanted to. And Twitter users may have noted, for example, sometimes, if they were using OAuth, they would be taken sort of like back to Twitter, where you would then say yes, I want to allow this application to operate on my behalf. And then Twitter would maintain a list of things that you had authenticated, and you were able to revoke that authentication any time you chose, never having to disclose your credentials and never having them disclose to that application. We're going to explain next week how that magic is possible.

Leo: Very sweet. Oh, I'm really looking forward to that. I'm a big OAuth fan. You know Google just announced that you can use OpenID, which is I guess kind of related to OAuth, to create a Google account, which means you can use your Yahoo! account to create your new Google account [laughing]. I don't know why I thought that was funny.

Steve: And of course you can use your YubiKey with OpenID.

Leo: Ah, of course you can, yeah. Steve Gibson is the man at GRC.com. In fact, you should go to GRC.com, not only to take a look at SpinRite, the world's best hard drive and maintenance utility - I use it all the time, recovery and maintenance utility - but also his free stuff like ShieldsUP!, which we just talked about, DCOMbobulator, Shoot The Messenger; and, if you're a fan of the show, we've got every episode, all 265 of them, online there. You can download each and every one and listen, both in the full 64KB version as well as a 16KB version, for people who want to save bandwidth. There are transcriptions, too. Is it every show? I think you did go back, didn't you.

Steve: Every show from the beginning, yup.

Leo: Every show. All of that is available at GRC.com.

Steve: And of course...

Leo: Steve is also on - go ahead.

Steve: I was going to say that the transcripts make all of this content searchable, too, which is very valuable for people who go, you know, when did they mention something about that? Well, you can find it.

Leo: You can search it on his page, or even on Google. Which is nice. Steve's on the Twitter. SGgrc is his Twitter handle. If you want to follow his iPad musings or tablet musings - and actually I bet you there are going to be some more, now that there are many more tablets on the way - SGpad. And the corporate account is GibsonResearch. All at Twitter.com.

Steve, thanks so much for being here. I'm rebooting my Mac now, now that the show is over, so I can put the - I can install the updates.

Steve: Me, too.

Leo: We'll talk to you next week on Security Now!.

Steve: Thanks, Leo.