Listener Feedback #66

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-196.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-196-lq.mp3

INTRO: Netcasts you love, from people you trust. This is TWiT.

Leo Laporte: Bandwidth for Security Now! is provided by AOL Radio at AOL.com/podcasting.

This is Security Now! with Steve Gibson, Episode 196 for May 14, 2009: Listener Feedback #66. This show is brought to you by listeners like you and your contributions. We couldn't do it without you. Thanks so much.

It's time for Security Now!, the show that covers all things security oriented with Mr. Steve Gibson of GRC.com. Hey, Steve. How are you today?

Steve Gibson: Hey, Leo. I'm great. I'm glad to be with you once again for week number 196.

Leo: Oh, you just love rubbing that in, don't you. Never missed an episode. In fact, we were just talking before the show began. I'm going to China in July, and Steve's already arranging for pre-records to get ahead so that we don't miss a single episode.

Steve: Yup. And you gave me the option of having, I guess, Alex Lindsay...

Leo: Yeah, I think Alex is going to be here, so I think we'll have Alex actually do

some of the shows, which is kind of cool.

Steve: Yeah, but I thought, well, I don't mind having a vacation, too. So if you're willing to double up, and we'll pre-record those, then while you're in China I get to have a couple weeks off. So...

Leo: So there you go.

Steve: ...that works for me. Not that I'm going anywhere. I'm just going to - actually, speaking of Starbucks, or I am speaking of Starbucks, I broke my record on Monday. I got there at 5:00 a.m. I left a little after 5:00 p.m.

Leo: What? You spent 12 hours writing code, huh?

Steve: A little more than 12 hours of code-writing, yes.

Leo: Now, here's the real question. How many coffees is that?

Steve: It's still my standard three mugs, each with two short shots of espresso.

Leo: Well, that's not so bad.

Steve: So it's not that much, no.

Leo: A mere six shots over 12 hours.

Steve: Just I sort of pace myself. And this particular mug I love. It keeps the coffee warm for hours, so...

Leo: And they don't mind you sitting there just beavering away while you're...

Steve: It's so strange, I mean, the model seems to encourage that. They've got - it's across from UCI, so it's sort of a study hall environment. They've got study tables with electrical connections down the center. We've got plugs. There's like tables on the side, all with electrical plugs, like much more closely together than they would need for their - to run their own vacuum cleaner. So, I mean, it's designed to encourage people to come and plug in and hang out.

Leo: Well, it's a pretty good deal because you're spending, what, 12, 15 bucks on

coffee, and you get 12 hours of rental, of space rental.

Steve: Oh, it's better than that, actually. It's $2.85...

Leo: Oh, it's nothing.

Steve: ...I think for my coffee. And that includes two refills, drip refills, which is what they qualify this as. So that's 50 cents each. So that's in the $2.85. So that's my coffee for the entire time.

Leo: Oh, geez, are you getting a deal.

Steve: I give them a Starbucks card which is registered. And as a registered Starbucks card user you get to use the AT&T hotspot for free.

Leo: Really. Oh, so you're getting your WiFi for free, too.

Steve: I disconnect - yeah. There's an up and a downside, though. There's one guy who shows up that, whenever he's there with his laptop, the whole network just collapses. And I've got a feeling he's running BitTorrent or, you know, he's doing torrent downloads. He probably comes to the hotspot in order to be anonymous.

Leo: Great.

Steve: And he's sucking down, you know, who knows what. So that's a bit of a problem. But, again, unfortunately it's free so it doesn't weed out anybody. But overall it's effective. And what I'm doing doesn't need much connectivity. Mostly I'm in my own environment, writing Assembly language. And I'm having a great time.

Leo: Oh, that's so much fun. And I'm sure they like having some cool coder in there plugging away at his Assembly. People peer over at the screen and go, ooh, ooh. I don't know what MOVE and LOAD mean, but that seems serious.

Steve: They see this Northgate OmniKey old-school keyboard. It's like, what the heck are you doing? It's like, ah, well.

Leo: Couple people asked me, UCI is UC Irvine.

Steve: Yup.

Leo: University of California Irvine.

Steve: And in fact we had a listener come by.

Leo: No.

Steve: I think it was Monday also. And he said, "Are you Steve Gibson?" I said, "I am." He said, "I love the podcast." So he'd heard me talking about me being at the Starbucks across from UCI, and the keyboard, and he said, okay, that's got to be the guy, so.

Leo: That's hysterical. Yeah, how many people in that Starbucks with a Northgate keyboard?

Steve: That would be zero. Well, that would be one.

Leo: You.

Steve: Yeah.

Leo: So we're going to...

Steve: In fact, Elaine was curious about the keyboard. And, oh, actually she was Googling in order to get the spelling, to see whether the "K" of OmniKey should be capitalized, and she found one on eBay for $150.

Leo: Wow. I think somebody sent me an email saying they still make them.

Steve: They're, well, there's a company, oh, it's three initials, I can't think of the name of it. But yes, they're still around. They're a little clankier feeling. They're, like, a little kind of hollow. But one of the - she mentioned, for example, all of the lettering on her key tops has long since worn off. And my keyboard, these keyboards, are from back in the day when they were so-called "two shot injected," where the coloring is actually plastic that is injection molded up through an opening in the shape of the letter on the top surface. So, I mean, you can't rub these off until you, like, wear the key down to its post under, you know, down below it.

Leo: Wow.

Steve: So they were really made right.

Leo: Yeah. So you don't - that put the company out of business because nobody needed a second one. I've been putting my keyboards in the dishwasher. I don't know if that - if Northgate would stand up to that. How do you clean it? Do you actually pry it apart and clean it?

Steve: Yeah, it does acquire a distressing level of, like, strange hair and skin cells and stuff. Like, okay.

Leo: I don't really...

Steve: So every decade or so, whether it needs it or not, whether it needs it or not I open it up and get a brush out and clean all the gunk. And it's good to go after that.

Leo: You probably wouldn't want to put it in the dishwasher, only because it's about a 50/50 proposition whether the keyboard survives.

Steve: Well, Greg, my illustrious tech support guy, as we were shutting things down and getting back into all of us telecommuting and working at home, we had this stack of keyboards from all the employees I used to have. They wandered off; I kept their keyboards. And I said, Greg, I'd like you to go through and clean all these keyboards - because he had time - and bag them up, and then I want them. And so I literally have an inventory of cleaned, as new, keyboards.

Leo: Oh. And they're all Northgates?

Steve: All Northgate OmniKey 102.

Leo: Oh, you're set. You're set for life. You've got a lifetime supply.

Steve: Well, I'm still using my original one. I mean, they don't die. So, yeah, so all my machines have them. And now Starbucks does, too.

Leo: You should see if you can leave it behind the counter.

Steve: Well, what I'm doing, there's one - I want to make sure - well, it only goes - I leave it in the trunk of the car, so I'm not having to schlep it around too much. But right now, of course, it's the old-style PC barrel connector, which was that large five-pin DIN connector. That then has a dongle to convert it to the PS/2, that has then a dongle to convert it to USB. And what I'm going to do, as soon as I settle on this, there's one keyboard that I've got on order which is the original so-called "buckling spring" IBM keyboard. I just want to make sure that's not better. I don't think it is, but I want to make sure because I'm then going to build all the conversion electronics into the keyboard, converting it into a USB keyboard. So it simply has a female USB connector on

the back, and I've got a little 12-inch short little pigtail USB dongle that'll go from keyboard to the tablet PC. So I'm working on really getting this refined. I have a feeling this is going to be a mode I stay in for a while.

Leo: I love it. I think it's great. That's how people should work, I think, in coffee houses. That's the best place to program.

Steve: Yeah. I'm getting a huge amount of work done. And the DNS benchmark is really coming along nicely.

Leo: Oh, that's fun. How cool.

Steve: And we'll be talking about it soon.

Leo: Pretty soon he'll be giving lessons, programming classes to the UCI students in Starbucks. It'll become like the Greek, what do they call it, the stoia, where Aristotle and Socrates would hold court?

Steve: Right.

Leo: All right, Steve. First I guess we should cover any news.

Steve: Yup. We've got a bunch of news. This is of course our podcast after the second Tuesday of May, so we've got Microsoft's standard "whoops" list of goodies. The most important one is a critical remote code execution vulnerability in PowerPoint, which I think we might have talked about last week. It was known. Now it's been patched. So they've got this one patched, as well. I also think I noted, but I'll remind people, that IE8 is now being offered through Microsoft Update. So, you know, it's checked by default. Microsoft's encouraging everyone to download it. And if you've got your Windows Update set to just accept everything that Microsoft provides without question, then you'll find you've been updated to IE8 behind your back, without your knowledge.

I don't have - I always go for "Notify me but don't download." Then I go into custom mode in Microsoft Update or Windows Update so I can see what I'm doing. I've actually had a problem in the past where updates had been offered that were not compatible with my hardware, like on various laptops, and it messed things up badly. And so it's definitely not something that you necessarily just want to - that every user, certainly the more informed user, just wants to accept blindly.

And then the third thing they're doing is we've got the standard monthly MSRT, the Microsoft Software Removal Tool update. So that's been added. And now that you've got, after this process, a new version of that, you might consider doing a deep scan of your system since it's a month more current, and hope it doesn't find anything new that it didn't know about a month ago. But it's always worth...

Leo: So you do that every time. Doesn't it do it when you download it anyway? It does a scan?

Steve: No, not the whole deep scan. It does sort of a cursory, quick, sort of in the background scan as you're booting. It's the next time you boot that it does it. And not surprisingly, most of these patches require you to do a reboot. So it takes that opportunity to sort of scan on the way back into the system. And you may notice sometimes that Windows seems to be starting a lot more slowly after an update. That's one of the things that's going on back there. But the deep scan is hours, and you'd definitely know if that was going on because, I mean, it takes as much of your system as it can, and for hours. So it's the kind of thing where you could do it before you go to China, Leo, that'd be good.

Leo: To all my machines. But you do it every second Tuesday after you get...

Steve: No.

Leo: No.

Steve: No. I'm just suggesting that, I mean, for example, I don't run AV, either. So I'm really not...

Leo: You're the bad example, yeah.

Steve: I'm not the person I'm talking to. But there are lots of people who like to scan their systems. This is there. It's free. It's useful and good. And you got a new one on the second Tuesday. So I would say go for it.

Leo: I'll do it before I go home tonight, maybe.

Steve: Yeah, that's good.

Leo: Yeah, then it'll be done in the morning.

Steve: Google had a little bit of a trip-over-their-own-foot update. They updated Chrome to fix a critical graphics integer divide overflow rendering flaw, taking Chrome to 1.0.154.64. And they didn't do it right. There were a bunch of problems relating to that. So they went to .65 two days later. It's stable now. So if we've got Chrome users, you may want to verify that you're running the 154.64 version. If not - I'm sorry, .65 version. If not, you may want to update yourself to fix a problem there. And the Mac had one of the largest wads that I've ever seen it download. In fact, frankly, I was concerned here as the clock was ticking that I wasn't going to be online with you, Leo, in time, because this thing just - it took a long time. It was half a gig download.

Leo: Wow.

Steve: And then the whole, like, restarting, hold on a second, I'm doing stuff phase, it just went on and on and on. And but I guess you have some information about that.

Leo: Yeah, it is the biggest update, according to people who watch this thing, ever for OS X. It depends, if you got the standalone one that works with all Macs, which probably, if you had more than one Mac, it'd be prudent to get that and just install it locally. That was over 700MB. And then the almost 500MB for the download that's specific to your machine, that would vary. Over 16,000 files changed, or 13,000 files changed.

Steve: 13,000 file changes.

Leo: Yeah, that's according to a guy who apparently - the interesting thing is, unlike Microsoft, Apple is not very forthcoming about what's in these updates. So there's a guy who actually goes through what they call the "bomb file," the archive file.

Steve: He was busy this time.

Leo: Yeah, he had quite a bit - and he does - I think he even compares what's been changed and so forth. And he does this and publishes his results in Macworld magazine. Rob Griffiths. Great guy. And I guess he just - this is his job in life. And he gives you a list of all the files that are changed. Sometimes version numbers change; sometimes they don't. But it's pretty much everything, including quite a bit of UNIX software. Remember that the Mac is running BSD and has many, many, many BSD programs, including Apache, Perl and Python, Ruby, all of those are updated. PHP, X11 was updated. So they don't just have to update their own stuff. They have to update also a lot of open source software.

Steve: There was a critical vulnerability that they fixed specifically in PDF printing. There was a JBIG2 token rendering problem that was part of this. So that was one thing. And obviously they just sort of brought a whole bunch of other stuff probably up to current level, essentially.

Leo: Yeah. I think they, you know, they don't have a second Tuesday policy. What they do, it's interesting actually, their process is very different from Microsoft's. They treat it almost like a new operating system update. They beta test it. They send it out to developers. So they beta test in public with these things. If you're, you know, you have to be a developer. And then after a month or so they will release these massive updates. From time to time they'll do small updates. But these dot updates, these point updates, 10.5.2, 10.5.3, in this case 10.5.7, these are pretty big updates and often do modify quite a few files. This is a big one.

Steve: Certainly in this case, yeah.

Leo: Yeah, yeah.

Steve: Well, a bunch of readers wrote to me about a story that ran in the San Jose Mercury News. Since it's a topic near and dear to our hearts, i.e., the Conficker worm, I wanted to just read it. It's brief. It was posted by, or reported by, Elise Ackerman of the San Jose Mercury News. The subject or the title of the story was "Conficker Worm Found in Hospital Equipment."

Leo: Oh, dear.

Steve: "A computer worm that has alarmed security experts around the world has crawled into hundreds of medical devices at dozens of hospitals in the United States and other countries, according to technologists monitoring the threat. The worm known as Conficker has not harmed any patients, they say, but it poses a potential threat to hospital operations. 'A few weeks ago we discovered medical devices, MRI machines, infected with Conficker,' said Marcus Sachs, director of the Internet Storm Center, an early warning system for Internet threats. Around March 24, researchers monitoring the worm noticed that an imaging machine was reaching out over the Internet to get instructions, presumably from the programmers who created Conficker. The researchers discovered that more than 300 similar devices at hospitals around the world had been compromised. The manufacturer of the devices told them none of the machines were supposed to be connected to the Internet - and yet they were.

"Normally, the solution would be simply to install a patch, which Microsoft released in October. But the device manufacturer said rules from the U.S. Food and Drug Administration required a 90-day notice before the machines could be patched. 'For 90 days these infected machines could easily be used in an attack, including, for example, the leaking of patient information,' said Rodney Joffe, a senior vice president at NeuStar, a communications company that belongs to an industry working group related to dealing with the worm." So...

Leo: That's interesting that they have that weird requirement.

Steve: Well, and so here you have a situation where clearly Windows - I mean, okay. I should tell our readers in advance that this Q&A, as our Q&As sometimes do, has a theme to it. Because there was so much listener response to the issue of instances of Windows being embedded in things that a lot of listeners have personal direct experience of one form or another, pro and con, with this behavior, or this practice. So some of - there's maybe half of our 12 questions are people telling us interesting things and asking questions. So I think it'll be - listeners will hear that sort of theme.

Well, this report is exactly that. More than 300 MRI imaging machines based on Windows got themselves infected because, A, they were based on Windows, that is, if they were based on some industry standard operating system rather than the consumer operating system, that couldn't have happened. And clearly the manufacturer says, oh, well, you weren't supposed to connect to the Internet. Well, I guess my question is, what, does it have a standalone box standing next to the scanner? I mean, don't give it an Ethernet adapter. Don't install Ethernet drivers. Or is it some just standard installation of Windows? Anyway, I could go on. And we've already discussed that. We'll be discussing it

more throughout this next hour. So I'll let that stand. But yes, that's certainly a problem.

Leo: Wow. Very interesting stuff.

Steve: Also we had a ton of people wanting to make sure that I knew about the controversy of NoScript versus Adblock Plus.

Leo: You know, I almost mentioned that last week. But, well, I'll let you tell the story. NoScript responded, I think appropriately, so...

Steve: Yeah. And I read the Adblock Plus posting, which I felt was very well written, even handed, not being out of control. We'd discussed previously this fundamental problem of an ecosystem, which I so much like, in Firefox. I mean, it happens that I'm using both NoScript and Adblock Plus. Adblock Plus with the easy list just deals with advertisements. It's amazing, when I go to a machine now that isn't using Firefox with Adblock Plus, everything's jumping around, and stuff's happening, and it's like, whoa, I forgot how peaceful things could be if you used Adblock Plus along with Firefox. And of course I believe in controlling scripting. So there's a fundamental problem if the interests of different add-ins conflict, and if the authors of the add-ons decide to respond to that conflict. The position taken by Adblock Plus's author is that the NoScript author has an advertising-based revenue system, and the NoScript technology was deliberately changed and the code even obfuscated so that it would resist being seen and resist analysis, was arranging to defeat the ad blocking for the NoScript site. And every time you get one of these NoScript updates, which really do seem distressingly frequent, I notice that I'm given a whole big page from the NoScript site, which I have to close because it has nothing else I really care about on it. But apparently it does have ads.

Leo: That's where the ads are.

Steve: Exactly.

Leo: And that's the only place the ads are?

Steve: I don't know that because I haven't dug deep into this. But I wanted to let our listeners know that I am aware of this issue. And many of our listeners are really unhappy with the conduct of the NoScript author, saying, you know, I want to use this, but this really puts a bad taste in my mouth. So anyway, that's the story.

Leo: You read his response, though; right?

Steve: No.

Leo: Oh, okay. See, this is - I thought this was kind of over last week, otherwise I

would have brought it up last week. So he actually says, "I screwed up big-time."

Steve: Okay.

Leo: "Not just with Adblock Plus users, but with the Mozilla community at large. I did something extremely wrong which I'll regret forever. I abused the power and wasted the enormous trust capital gained by the NoScript add-on through the years to prevent Adblock Plus from blocking stuff on four Internet domains of mine without asking an explicit preemptive user consent. This is absolutely inexcusable, something I would never conceive again for the life of me. Please let me apologize first." He responds to Vladimir's post, says it's not quite as Vladimir said. And he says he's changing it. So to be fair...

Steve: He really did, he took the responsibility and he apologized. And also Vladimir's follow-on postings did refer to responses back from the NoScript author. So I would say lesson learned.

Leo: Yeah, I think lesson learned. I think that he has come up with a solution. He realizes it was inappropriate to do what he was doing. But there has been a back-and-forth dialogue between him and Vladimir. And, I mean, he said, "I didn't obfuscate code." I mean, in his defense he said some things. So it's not completely one-sided. I would say read, you know, go to his Hackademix.net site, read Giorgio's responses...

Steve: And give him as full an opportunity to explain himself.

Leo: Yeah. I think these are - my general gist of it, after reading the back-and-forth, is that both people are well-meaning. Giorgio realizes he made a mistake, and I think he's going to fix it to the satisfaction of everybody involved.

Steve: Great. Great. I've been, as I've mentioned a number of times, I have sort of a background thread running relative to my plans for my next major commercial product, which as we know is called CryptoLink. I've got the trademark on that now, so it doesn't hurt to mention that. One of the things that I run across is this question of what mode CryptoLink is going to use for doing its ciphering and authentication. We've talked about modes. For example, Cipher Block Chaining, CBC, is a mode where you take the result of one, of the first cryptographic operation, and you XOR that with the plaintext of the second block's worth of text before you encipher it. You take that, and you XOR that with the third block's and so forth, so that it forms a chain running down through.

What that does is it makes all the blocks dependent upon everything that has come before and, for example, is a much stronger solution than if you independently were to encrypt each block by itself because you could then - a bad guy could change something, and there would be no rippling effect. It would be an isolated change. Also, and more significantly, any patterns that were occurring in the plaintext could be seen very clearly because the patterns occurring anywhere would be replicated in the enciphered blocks. Even though you wouldn't know what the plaintext was, the fact that it wasn't changing

gives you information, and that's a bad idea. So these modes take the basic cryptographic operations and make them a little more complicated.

Well, the problem is, all that does, for example, Cipher Block Chaining, is give you privacy through encryption. It does not give you authentication. So what everyone does is, for example, you have CBC. Then, for example, there's CBC MAC, which is to say Message Authentication Code, where with a different key - and that's important. We know that you can't use the same key, or you shouldn't, for ciphering and for authenticating. That's one of the mistakes that the early versions of SSL used, and that got fixed in later versions. So if you want to authenticate, you use a different key, and you often do the whole thing again, but, for example, using CBC to create - as your cipher for a hash. And so you hash the message with a different key. What this means is that you're essentially doubling the amount of work you have to do.

Well, there's one very cool mode which is called OCB, which was invented by a cryptographer, Phillip Rogaway, who is at UC Davis. I wrote to Phillip in February, so a little over three months ago, asking him what the terms of licensing were. His site says this is licensed, it's proprietary, it's patented, but we will make licenses available. Well, I never heard anything from him.

So yesterday I remembered that I had let that ball drop so I sent him a short note just saying, hey, I'm just kind of wondering if I could hear something one way or the other. A lot depends upon whether I could use this. I just realized I forgot to tell you why this is so cool. OCB does both - both privacy, that is to say, encryption, and authentication in a single pass with, like, one additional block operation at the end rather than doubling the number of block operations that you need. So it's, like, twice as efficient as any of the existing technologies. Anyway, I received a response this time from Phillip, like within an hour. He said, "Hey, Steve, great to hear from you. You can use OCB for free, now and forever, in any future versions, for your CryptoLink VPN."

Leo: Oh, that's great.

Steve: He says, "I have no intention of profiting from a three-man software publishing company, so it's yours."

Leo: He just wants to protect against the big boys.

Steve: Yes, exactly. And, I mean...

Leo: Understandably.

Steve: ...who can blame him? I mean, so I'm just delighted because this, even though it's all going to be Assembly language, and mine would have been as fast as anybody else's, now it gets to be twice as fast. So...

Leo: That's great.

Steve: ...that's a cool thing, too.

Leo: That's really great.

Steve: And I did want to make a mention, again, responding to all the email that we received, that I'm aware that my little freeware SecurAble has been mentioned all over the place in connection with Windows 7 because there's been a lot of question about what processor capabilities you have. Windows 7 cares about whether you've got the advanced virtualization capabilities in your chip. Yet it's not easy to know one way or the other whether you do. And of course our listeners will remember that that's exactly what SecurAble tells you, is it makes - you just run it and, bang, it shows you a little presentation of what features your chip has. So it's been picked up by PC World and Tech Republic and, you know, all over the 'Net people have been saying, hey, just go get SecurAble. And downloads have spiked as a consequence of that. So I wanted to make sure that our listeners knew that I knew that had happened.

I wanted to correct something I misstated about the Kindle DX. I referred to its pixel pitch as being higher than the prior two Kindles, and in fact it's a little bit lower.

Leo: Oh, it's lower, okay.

Steve: Now, the total resolution is higher, in fact nicely higher. But the actual dot pitch, that is, the spacing between individual pixels, is a little lower. On the first two Kindles, the original Kindle and Kindle 2, it's 167 dots per inch to yield a screen resolution of 800x600 at that pitch. On the DX it's 150 pixels per inch. On the other hand, the screen, because it's physically larger, ends up at 1200x824. And 1200x824 is very respectable resolution.

Leo: 1200x824, yeah, that's bigger than a VGA screen.

Steve: Exactly. Well, think about many laptops that are 1024x768. 1024x768 is a standard laptop screen resolution. So this is more pixels in each direction. And it's a physically smaller screen than your typical laptop screen at 1024x768. So the actual visible resolution ends up being higher. So I've got high hopes for it being a nice screen for viewing PDFs. And it does have a rotation sensor in it, and it will go into portrait mode. So you get the advantage - it'll be interesting to see how they handle that because you can't scroll eInk. So if you switch it into portrait mode - I'm sorry, I meant landscape mode. If you switch it into landscape mode so that you get the whole width of the - what would normally be the height of the screen becomes the width of the screen to get even greater resolution, you'd have to do a paging operation to move up the screen.

Which actually one writer or reviewer wondered why they weren't doing columns, like they weren't doing multicolumn text. Well, you really don't want multicolumn text in a situation where you can't easily scroll because then you'd get down to the bottom of one column, and you'd have to scroll back to the top of the same page in order to start reading from the top of another column. So it makes sense that they've stretched it all the way across. So anyway, we won't really know much more until I get mine. And then we will. And I did want...

Leo: And you may get yours someday. It's not imminent, yeah.

Steve: And I did want to mention that a guy posting to our newsgroups this morning noted something disturbing about his D-Link router firmware update. And that is that D-Link is now going to begin - they haven't actually started yet. But they're going to begin taking over DNS by default. And that's something I really think crosses the line, especially when the DNS that they're going to be routing people to is that kind which doesn't give you an error message, but instead gives you their own helpful, friendly, marketing-driven interception page. They say, we're providing search engine services for our customers. Turns out that it's some sort of a direct-marketing company that they're using. So they're...

Leo: It's not OpenDNS, it's some other weird - because I'm a big fan of OpenDNS. I could even see maybe putting it in the D-Link router by default.

Steve: Well, I'm not a fan of anything that doesn't give you an error. And OpenDNS does not give you an error.

Leo: No.

Steve: OpenDNS takes you to their own page.

Leo: Yeah. That's how they monetize, exactly. I mean, they say it's an error. But they say, perhaps you meant this, and there's some ads there. It's a surf page.

Steve: Well, but an error means something specific. An error means that you receive a bad name error response in response to a query at the DNS level. It does not mean that your browser receives a happy page of other links. And so that breaks DNS. This is fundamentally broken. I understand they're monetizing, and I'm being a curmudgeon about it. But, I mean, you ask for a name that doesn't exist, you should get an error. You should not be given an IP that will satisfy your request by taking you to a page for marketing purposes.

Leo: Yeah, well, you certainly shouldn't do that without - see, because most users of D-Link routers won't know that that's what's happening.

Steve: Precisely.

Leo: This will be the new default behavior. I don't have so much of a problem if people choose to use OpenDNS, and in fact I recommend it.

Steve: No. And in fact you do need to deliberately configure your system to do that.

Leo: Right.

Steve: And I'd agree. I think it provides lots of good services, so long as the user understands...

Leo: Yeah. You did it yourself, and this is a conscious choice. Problem is, putting it in the router, most people don't ever change the router settings.

Steve: Well, and what's really become annoying, and we'll be talking about this in the future when we're talking about DNS performance, as I will be when we talk about this benchmark that I'm in the final throes of, although I've been in those final throes for some time, I realize, but it's coming out pretty spectacular. It turns out that many routers are not forwarding the ISP's provided DNS services to the computer. Instead they're saying, I'm your DNS server. And so they're intercepting your computer's or your whole LAN's DNS queries and managing them themselves.

The problem is, it turns out that this is being done poorly. It's often slower than asking the actual DNS servers that your ISP is providing, and in some cases it creates a security vulnerability because some replies can crash your router. And there may be buffer overrun possibilities there. So when you set your computer to "Obtain IP address automatically," what you traditionally received was a local IP from the router for any machines on the network. And then you received the router basically resending, rebroadcasting the DNS servers it had received from the ISP so that your computer would be making queries to those DNS servers. Now the router is giving its own gateway IP as also the DNS IP, the single DNS IP for the whole network.

The other reason this is a problem is that DNS is pretty fancy about how it falls back in the event of a server being down. The reason there's typically a minimum of two DNS servers being listed, I mean, the formal reason is if one is down, you can fall back to the other. Because if you've got neither, you don't really have the Internet, unless you're really good with memorizing IP addresses. And none of us do that. So who knows how good the error-handling logic is in the router, which you're now depending upon if you only have one IP address, and that's the router's IP.

So I'm a little bit down on this whole notion. And unfortunately this is the default setting. And so what we're learning as we actually have really accurate benchmark results is that changing your router and telling it do not proxy my DNS is going to be probably the recommended setting. And it also solves some potential security vulnerabilities which we're finding and discovering in a surprising number of routers.

Leo: Very cool.

Steve: And lastly...

Leo: I mean, not cool, but very good to know. "Cool" is the wrong word.

Steve: Yes. And lastly, in the spirit of our Q&A, I actually have a question from a

SpinRite user that I would share because it's an interesting one. He said, "Steve, I've been using SpinRite a lot at work with a four-copy site license - thanks - to both recover corrupt drives and files as well as maintenance on drives. My question is, after running SpinRite Level 4 on a drive, is the drive safe to use? One Western Digital drive, a 500GB SATA, was giving us errors and trouble. So I ran SpinRite at Level 2. It fully recovered a bunch of sectors. After that I was able to get all the data off that I needed. And then I ran SpinRite again at Level 4. No bad sectors a second time; some ECC corrections, but I don't know what is common or too many. Is this drive safe to use and trust again, or should I just toss it? Thanks for the great podcast and great product. Nathaniel."

Leo: Very cool. Very cool. That is cool.

Steve: Nathaniel L. in Minneapolis. And you know, that's a really good question to which I don't really have a hard-and-fast answer. I would say no drive should ever produce an error like SpinRite always finds. So that's never a good thing. On the other hand, it's difficult to blame the drive if there hasn't been maintenance going on. I mean, SpinRite is maintenance. You run SpinRite on a drive, and it's able to show the drive that there are sectors that are going bad before they have gone bad. And that's a critical distinction because these drives are so dense now, and they're relying so heavily on error correction to fix reads which are not perfect, that a lot of error correction is going on all the time. It's when too much is required, more than the drive has, that it says I can't read this sector. Which is then when you bring SpinRite in, and it checks it for you. So if maintenance were going on, then this would probably have never gotten as bad as it had. But maintenance isn't being done on most drives because most drives don't have SpinRite being run on them all the time. Only people who are clued into this understand that.

So I would say keep an eye on the drive. If you have some way of maybe putting it in a less mission-critical place, like make it a drive in a RAID array so that, if it goes belly up, you've got redundancy, and you're not relying on it completely. I would say maybe store less mission-critical data on it. Or if nothing else, make sure that you do run SpinRite over it from time to time and kind of keep an eye on it. On the SMART display screen, SpinRite is showing you dynamically the rate at which error corrections are being required, even behind the scenes, something that nothing else shows you, the actual rate in terms of corrections per megabyte read. And if you see that in general going up over time, then the drive is, I mean, SpinRite is a more sensitive early warning device than anything I've ever seen because it's able to show you the rate at which error corrections, which are still fine, are being relied upon by the drive as it goes about doing its work under SpinRite. So if that makes a jump or a change, or if it's high compared to other drives of the same make and model - that's the other thing you can do is, if you have other Western Digital drives, see what their ECC rates are as shown by SpinRite when it's running. And if you've got a drive that's, like, way off bell curve, it's like, okay, this really does seem like it's an early warning for a pending problem.

Leo: Well, it's nice that people have this option to write in to you and ask the author what's going on and what you suggest. I think that's a great - I've always had that feedback from you, but it's nice that others can do it, too.

Steve: Yeah.

Leo: Go to GRC.com/feedback if you have questions for Steve. Hey, by the way, Colleen handed me a keyboard she found that has buckling springs. That's what makes your Northgate so good; right?

Steve: Well, no. Maybe. I know that what was called "buckling spring keyboard" was the original keyboard from the IBM PC.

Leo: Right, the Model M.

Steve: And it had a very stiff response. I joked in InfoWorld columns that you could just, like, load up a spitball in the curve of one of the keys and, like, let it go, and the spitball would get shot across the room because, I mean, it really - or I think I remember once saying it would bruise your fingertips if you didn't pull them away quickly enough because as soon as it wanted to snap back, I mean, it really did with a vengeance. And I think that's too stiff. I think that's more than I want.

Leo: So these aren't buckling springs then in your Northgate.

Steve: Well, I don't know what the technology is. I know that...

Leo: That's what Dvorak has always said. But, you know, who knows.

Steve: Well, I'm really careful about technology or terminology. And I know that the original IBM keyboard was known as the "buckling spring keyboard." And they're still available. I ordered one. Then they told me, oh, by the way, it's going to be maybe two or three weeks. So I'm still waiting for it. I think I'm not going to like it, and I'll probably have to put earplugs on in order to use it. Mine are definitely - they have a nice snap action. They're not that, you know, the reverse sheet dimples that you're typing on, which is what all contemporary keyboards are.

Leo: Right, the mushy ones. That's actually - I have to use those because the other ones are too noisy.

Steve: Yes.

Leo: They really clackety-clack. I can't use them on the air.

Steve: Yes, and in fact I've got this in front of me, and I've been conscious, sometimes when you're talking, and there's something I have to do, it's like, ooh, can I type on this quietly somehow because I don't want it to get recorded, yeah.

Leo: Well anyway, she made a mention of it. It's the DAS keyboard. It's from Germany, DAS, using buckling springs. So people are looking for that.

Steve: And do you have the keyboard in front of you right there?

Leo: I don't. She found it.

Steve: So you're not able to press the key right now and...

Leo: I can't tell you, yeah.

Steve: Oh, I've just remembered the company, it's CVT is the company that still makes the equivalent of the Northgate keyboard, sort of modernized, and it's got some built-in macro programming and some other stuff. So but it's CVT, Inc., I think is the name of the company. And I have purchased a couple from them. And they were - they didn't quite have exactly the same feel, I mean, I'm being really picky at this point. But yeah, I think that's the name of the group that...

Leo: I also - I bought one for the Mac.

Steve: Oh, yes, exactly. There was one for - I have one also.

Leo: Yeah.

Steve: That is, again, it's a nice, snappy...

Leo: I like it.

Steve: Yes.

Leo: And it washed well in the dishwasher. But it's too noisy, its clickety-clack, and I had to retire it. The new Apple keyboards, you know they're membranes because they're this thin. They're aluminum.

Steve: It's like typing on a piece of paper.

Leo: Yeah. I've washed two of those. One made it, and one didn't.

Steve: What is this washing you're doing?

Leo: Well, you know, with the swine 'flu, you know, the keyboard is filthy. It's filthy.

Steve: You've got a lot of people sitting on your ball?

Leo: I do. We have - because Colleen comes in here. Sarah comes in here. Alex does. And a few other people have used or use this. So I feel guilty having them sit down in front of my grungy keyboard. I am actually using now something called the Silver Seal Antimicrobial Keyboard that is designed to put in the dishwasher. It's for hospitals.

Steve: Get a guest keyboard and just put a big - put a Leo sticker, put your name on yours and get a guest keyboard and just swap it out...

Leo: Maybe that's what I'll do.

Steve: ...when you're going to be away from your desk. Or actually, since they're USB, you can probably leave them both plugged in and just sort of...

Leo: Yeah, you can, yeah.

Steve: ...slide one aside and say, okay, this one's...

Leo: Don't touch that.

Steve: ...got Chinese food all over it, so try not to - don't use this one.

Leo: Are you ready, my friend, for...

Steve: Oh, I think everybody's ready. We're 52 minutes in, and we haven't...

Leo: Holy cow. All right, well, we'll speed through these.

Steve: But we've had a lot of fun so far, and our listeners appreciate that. So it's not just about the Q&A.

Leo: No, of course not. It's never just about the Q&A. This is question number one,

comes to us from Robert Minkler in Prescott Valley, Arizona. And he says: Steve, you skipped a beat. I just finished listening to Security Now! Episode 195, the SSL/TLS episode, for the fourth time. Holy cow.

Steve: Yeah, that was a pretty information-dense episode last week.

Leo: Yeah, yeah. I admire people who really just keep going through it till they get it. Don't forget the transcripts, because that helps. It's a great episode. As usual you make a very complex topic easy to understand. It seems like you missed an important detail of the handshake process, though. See, I wouldn't have noticed that.

Steve: Yeah, I did afterwards, too.

Leo: You mentioned what happens if the session details are already in the cache of both machines during the handshake. You never said exactly what happens when the two machines need to create and exchange a new symmetric key. I know you discussed symmetric key exchange in a previous episode, but seems like you should have mentioned that again. So here's a chance. How exactly do both machines end up with the same symmetric key and keep it a secret? Thanks, Steve and Leo. Security Now! is by far my favorite show.

Steve: That's a great question. And it's funny because, as I wandered off after recording last week I thought, oh, in my notes I just must have skipped over it. Because I talked about how, when the client is connecting to the server, the client, if it sees it has credentials which have been established from a prior connection still in its cache, it offers the session ID of those credentials to the server. If the server is configured to allow reusing of existing credentials, and if it still has them, and it chooses to, it will respond with the session ID, coming right back to the client, telling the client yes, let's use the session ID that you suggested, and implying the use of the same credentials.

At that point I moved past that to other topics. But the question that Robert asks - and I should say that a number of other listeners who were really listening carefully said wait, whoa whoa whoa whoa, what if there isn't a prior session? What if it's the first connection to the server in 24 hours or whatever the cache expiration time is, which is never longer than 24 hours. And they're right, I forgot to mention that. It's pretty straightforward. When the server gives the client its certificate, it sends the server certificate message to the client and then sends the so-called Server Hello Done message. In the case that they need to negotiate a new shared secret key, the client sends what's called a Client Key Exchange message.

What it does, it takes the version number that is it, that is, its version number, which is two bytes, and generates an additional 46 bytes of cryptographically strong pseudorandomness, creating a 48-byte datum. Now, having received the server's certificate, just having received it, it has the server's public key. So it encrypts that 48-byte blob, which is not too long, with the server's public key.

Now, as we know, the public key allows you to go in one direction. The private key, which is never disclosed, is the only thing that will let you go in the other direction. So it

encrypts the server's private key. I'm sorry, it encrypts its 48 bytes of mostly randomness, but also the first two bytes are its own client version. And then it sends that to the server in its Client Key Exchange message. So the fact that everybody who's monitoring this handshake process can see that doesn't help them because, I mean, they see 48 bytes of blob go by. But as we know, that's going to be random noise. I mean, it's encrypted random noise. They cannot determine what the actual plaintext, the unencrypted randomness is because they don't have the server's private key. Only the server has that. So it receives it and decrypts that using its private key.

Now both ends have the same 48 bytes. The reason that client ID is tacked on the front is, cleverly, to protect against a client version downgrade attack. Remember that all of this so far has been in the clear. That is, the client sends its packet of information containing its version number, the ciphers and authenticating codes it supports and so forth, up to the server. The server looks at those, chooses among them, and sends them back.

Well, the client is also sending the server a blob of random stuff. The server sends the client a blob of random stuff. This all happens in the clear. So a possible attack would be to intercept that first client handshake packet that contains the client's version number and edit it down in order to make the client seem dumb and force the server to negotiate an older version protocol that has known weaknesses. So that's prevented by having the client again put its client version, that is, the best it can do, as part of that 48 bytes, which is then encrypted. And of course this is encrypted and authenticated so that nobody can change it.

Then the server receives it, decrypts using its private key, as I said before, gets all 48 bytes, the first two of which are the client version. It compares that client version, which now they've got ciphering running in their connection. Now it can't be spoofed. It can't be intercepted. And it compares it to what the client's first claim of its version number was. If they don't match, the connection is broken. The server just drops it like a hot potato and says "no thank you" and forces a complete restart of the negotiation.

Typically it will compare. Then what happens is they've each generated some randomness in the beginning, which they sent to each other. Then the client generates another chunk of randomness. They each generate, oh, I don't remember how many, I think it's 48 bytes, and they send it to each other. Then the client generates 48 bytes, which it encrypts, sends to the server, which it decrypts. They mix all of that together - the client's randomness, the server's randomness, and their shared secret, which is called actually the "premaster key." That's all mixed together using a common hashing function to generate the final master key, which is then used to generate all subsequent keys.

So that's how that's done. They then both issue a cipher change spec message to each other. And remember that that's the message that says everything henceforth will be done under the ciphers that we agreed upon, using the key we've agreed upon. And then they each send the finished message, which is enciphered and authenticated using what they've agreed on. And upon receiving that, and each end verifying it, they've established their secure communication. And nobody knows of any way to get in there and mess it up. So it's a tremendously secure technology.

Leo: Very cool. Question two. Marv Schwartz at Case Western Reserve University in Cleveland, a very good school, worries about connecting to the LAN side of a router: Hi, Steve. Perhaps listening to every episode of Security Now! has made me

paranoid. Good, it's working. You said that because of NAT, worms cannot cross from the Internet through the WAN side of a router to the machines on the LAN side of the router. However, if someone connects a laptop that has been infected with a worm to the LAN side of the router, then the NAT protection does not come into play, and the worm will try to propagate to other machines on the LAN side of the router. And we've seen that happen with Zotob, among other things.

Steve: And Conficker does it, too.

Leo: Yeah. When this happens, the only protection is the software in each attacked machine. Even though I keep my machine updated, the frequency of security updates, zero-day exploits, and Conficker scares me. So I carry a travel router with me and put it between my machine and a corporate, conference, or hotel LAN whenever I can. Is this a good idea? Or am I just being paranoid? Thanks to you and Leo, a great service you provide so many of us who do not know enough to be paranoid before becoming regular listeners, did not know enough before becoming regular listeners to Security Now!. Thank you, Marv. That's a great question.

Steve: Well, it's a great question. And one of my good friends, Mark Thompson of AnalogX, does that. Sometimes when he's visiting he'll crash here overnight on his way up to L.A. or Burbank or wherever he's going. And he just has a little tiny D-Link travel router. It's a very cool little thing. It's like the size of, like, sort of smaller even than a Mac power supply blob. It's got a little plug built in. You just sort of swing the plug out or slide it out, sort of like an electric razor being deployed, like the sideburn cutter on a razor. And you just plug it into an outlet, and it is a complete little NAT router.

And so what Marv is doing is interesting. You could argue that a properly running Windows firewall is the same thing, that is, if you've got no ports opened and exposed, then the Windows firewall is very much like NAT in that anything from the inside is able to get out, but nothing from the outside is able to get in.

The problem is that the Windows firewall is tricky because there's, like, tabs in the advanced mode where you're able to configure exceptions, you're able to allow things. By default LAN is treated as a trusted domain. But Marv's whole point is that a LAN shouldn't necessarily be trusted because it just takes one bad machine stuck on the LAN. And one of the behaviors I witnessed myself in my own little Conficker honeypot here is it's sending out probes across the entire local network, looking for other machines to grab onto, to infect, and to move itself over to.

So for people who like the idea, getting a little so-called "travel router," and I think everybody makes them now because it's a nice idea, running your Internet connection through that gives you sort of hardware-level protection outside of your Windows machine, that is, that local software cannot configure. You want to do the right thing, of course. You want to make sure you give it a good, strong administrator username and password, change them. You want to disable Universal Plug and Play, otherwise don't bother because Universal Plug and Play essentially disables that protection, and there's no way to really know what's going on. So you want to turn that off. But with that turned off, a little travel router makes a great - basically it's a little hardware, a little portable hardware firewall. And, I mean, I know people who do it.

Leo: Failing that, the next best thing would be just to turn on the Windows firewall. That'd give you some protection, too; right?

Steve: Well, no, that's the point, is that even when it's on it does not provide LAN protection.

Leo: Oh, no, no, wait a minute. If all the machines in the network have the firewall turned on, you are protected.

Steve: No. No. By default Microsoft has filesharing and their client, the Windows client on. And the firewall trusts the LAN. It does not trust the...

Leo: Oh, that's interesting. Oh.

Steve: Yes. And so, like, if you go into the advanced...

Leo: Because this was the recommendation after Zotob was turn on the internal firewall, and that will block Zotob. But so when it sees traffic from the LAN it says, eh, you must know what you're doing.

Steve: Yeah, I mean, a LAN, it's your home, it's your office, it's your friends. And again, it's interesting because there's a...

Leo: So how do you protect yourself if, I mean, do you bring a little router with you even to work?

Steve: Well, you can configure the Windows firewall so that it doesn't allow these exceptions.

Leo: Oh, okay. How do we do that?

Steve: You dig down into the tabs. And, I mean, we could certainly do a podcast to take our listeners through it step by step, but...

Leo: It's probably obvious, though.

Steve: It's pretty obvious.

Leo: So you turn off all the exceptions.

Steve: Although under each of those exceptions - you need to open it up. And you'll see that it says LAN is an exception, and you're able to disable that and say, nope, I do not want LAN to be an exception.

Leo: Holy cow. I didn't realize that. I've been thinking I was safe using the - turning on the Windows firewall. So...

Steve: Not against internal threats.

Leo: They make really good little travel routers you can carry with you.

Steve: Yes, and that's my point, is that everybody makes them. And it makes it a no-brainer. You need to, again, configure it correctly, make sure you don't leave the default username and password because we now know that malware tries, it does brute-forcing against username and password, and that it'll use Universal Plug and Play if it can. So those things you need to deal with. But when you do, you've got quick and easy security. And the other problem is you can't ever really trust a software firewall running in the computer that you're protecting. I mean, except...

Leo: Because it's - a bad guy has access it. But I'm thinking this is of use when you're sitting safe and secure, and there's bad guys outside on the LAN.

Steve: Correct, correct. And so long as you don't click on a link that runs something that your browser has invited in...

Leo: You're screwed if you - once you're infected, all bets are off.

Steve: Right.

Leo: I mean, you know, at that point, okay. We're talking about protecting ourselves. So, okay. Yeah, that's good, okay. I'm going to have to go out and get a little portable - there must be some - because I'm going to China. I know I'm not going to - I'm going to the land of the hacker, baby.

Steve: Oh, baby. I know that the little D-Link, this little cute D-Link that Mark Thompson has is really neat. And he says, I mean, he was saying, he was commenting on how much he likes it. And he said, oh, and it's also got WiFi. So you're able to, like, for example, he's got WPA configured. He'll come to my house, get a hard connection to my cable modem, for example. And then that's all he has to do. The router is permanently configured with his WPA key, big long key. So then his laptop, he's able to roam around the house and use his WiFi connection, his secure connection to his little travel router that is then plugged into my cable modem. Everything just works. So it's like a zero-config solution. It's also very secure.

Leo: Yeah. I think that's a great solution. Okay, I'm going to find one. And of course if it's D-Link make sure you change the DNS.

Steve: Yeah.

Leo: Just in case. We've got two listeners with the same question: Jason Russo, writing through our tech support - your tech support email for GRC.com, and also JD posting anonymously on the security forums, have comments about JavaScript in PDFs. Jason writes: Steve, sorry if this is the wrong email. I couldn't find a good way of commenting on your podcast [GRC.com/feedback].

Steve: Yup. Which is what I wanted to mention, make sure our listeners know GRC.com/feedback gets to me.

Leo: Just a quick comment on JavaScript in PDFs. JavaScript, in addition to Adobe's FormCalc, is used within interactive forms for programming logic in a PDF. This is usually used for calculating fields, input validation, et cetera. But it can also be used to dynamically change the user interface, which is where FormCalc falls short. Unless Adobe abandons interactive forms, JavaScript is a necessary evil for the reader. There is a reason for it, it's just not used by most people. P.S.: Love the show. Well, thank you, that's good to get the clarification.

JD adds: Hi, Steve and Leo. First, let me express to you how much I appreciate your netcast. It keeps me informed and up-to-date on matters of Windows security. For that I am grateful, and I subscribe to Audible because of it. Thank you. Our sponsors thank you. Per your comments in Episode 195, it appears there is a use for JavaScript in PDFs, at least according to Adobe. PDFs have the ability to have Flash content embedded inside. That, by the way, is a really neat feature. You can make a book, a PDF that has audio, video...

Steve: Live things, yeah.

Leo: Interactivity. But herein lies the risk. I've personally seen a demonstration of this in the form of a Flash-based dashboard that contained data represented in the form of pretty gauges. These dashboards could then be sent to an executive or other morons for review. They could then play with the data by adjusting graphically represented sliders and knobs to analyze the effects on the data as it's changed, keeping them busy and out of our hair. No, I'm adding that part.

Following your advice, I switched off the JavaScript and attempted to open a PDF with an embedded dashboard sent to me by a coworker. I got an error message telling me it would not display without JavaScript enabled. Yeah, that's one way that you imbed Flash. Well, I don't want to detract from his comments. Perhaps there are others who could explain this in more detail, but I hope this sheds some light on why JavaScript might be used in a PDF. Best wishes to you and Leo. Keep up the good fight. Thank you, JD and Jason.

Steve: Well, I appreciated both of these guys. It's weird, too, Leo, because wouldn't you know, shortly after I said that, I wanted to submit to the state my request for voting by absentee ballot for this upcoming special election that we've got on May 19th. And so I went to the site, and they gave me a PDF-based form that had auto fill-in stuff. And it didn't work because I had disabled JavaScript, following my own advice to our listeners.

So the good news is it tells you, and I wanted to highlight that, that JD mentioned that, when he tried to run this, the PDF said wait a minute, this needs JavaScript. So I continue to suggest that normally running with it disabled is the better thing because you'd rather be notified that malware or not malware is wanting to do something than to have it be able to do it without your permission or your knowledge. So but I thought it was important, since I had just told all of our listeners go turn this off because it's dumb and you don't need it, well, okay, whoops, there are some places where you do.

But the good news is you will be told if you do. And so then you can decide, again, very much like NoScript in Firefox, you can decide, does this matter to me? Do I think this is probably legitimate? In which case you turn it on, and then you get the full functionality that JavaScript provides. I'm not saying it's not useful, I'm just saying it's so prone to abuse that it'd be nice to have that little intermediate step of saying, okay, this time I'm going to turn it on. Instead of going to a site that is downloading a PDF without your knowledge and using it to take over your machine. That's not what you want to have happen.

Leo: SCS online is saying in our Stickam chatroom that the IRS also uses these interactive PDFs for tax forms. That's - there's two things you really don't want to go together, insecurity and tax forms. But, you know, yeah, it's an issue of functionality. I understand that.

Okay, got a long one. I'm going to try to skim through this one. This is from Adrian Oliver in Chiang Mai, Thailand. Wow. Chiang Mai? He cautions about getting "embedded" with Microsoft: Hi, and greetings from Chiang Mai, Thailand. Following one of your recent Security Now! programs about the power grid in the U.S. being compromised, I thought I would share the following: I used to work with a large industrial automation systems manufacturer, which for the last 30 years has been designing, building, and installing big and small control and monitoring systems from nuclear power stations down to small factory systems. Throughout all those years the critical control systems have always been based on either homegrown proprietary operating systems or a commercial real-time OS like VxWorks.

Ten years ago Microsoft were trying to convince us and similar companies to adopt Windows Embedded. When asked the question of vulnerability issues, they stated they would guarantee a patch within two weeks of discovery. The control system could then automatically download it from the Internet, install, and reboot. Obviously the concept of rebooting the control system of a nuclear power plant did not worry them, but it sure wasn't feasible for us. Given that the default for Windows is to automatically - first of all, it's online; right? Problem number one.

Steve: Yeah.

Leo: Given that the default behavior for Windows is to automatically download, install, and reboot at the same time, it would be likely that all such Windows-

powered control systems would reboot - and potentially fail to boot - at exactly the same time. Aagh [laughing].

Steve: Brings a new meaning to the notion of the second Tuesday of the month.

Leo: Talk about unclear on the concept. In the 20 years I've been in this industry our company never had any control failures in the dedicated control systems due to viruses, attacks, or hacking attempts. Yes, it is true that almost all the supervisory systems are now Windows based, which are as vulnerable to viruses as any home computer. However, they are normally operating as supervisory only, reporting, monitoring systems, hopefully never part of any control loop. I hate that word "hopefully" in there. All critical control systems are designed to operate and continue operating, even when the supervisory system fails for any reason.

Fortunately, I know that most European operators and manufacturers are extremely careful with what they allow to run their systems. One pharmaceutical company near where I used to live in the U.K. manufactured penicillin. Because the manufacturing process of penicillin produces a very fine dust, the explosion hazard is extremely high. The estimated blast radius, should an explosion occur, is one and a half miles. The local, permanently manned fire station is two miles away. Consequently, the people who run the plant were extremely particular - no pun intended - about what software was used to control the plant, as their lives depended upon it. It sure wasn't anything from Microsoft.

On a different but related note, NASA's Mars Rover is running VxWorks, as well. Still running. Had they been designed with Windows XP, we would have had to ask the Martians to reboot several times by now. Hey, great email. Wow. Thank you so much, Adrian Oliver in Chiang Mai, Thailand. That kind of says it all, doesn't it.

Steve: Yeah, I mean, this Windows Embedded is another one of these bad ideas. We've seen it before. This is what Microsoft does when something comes along that they don't really have an answer to. We had UNIX on the Internet for years. Then the Internet became important. And Microsoft said, oh, hmm, we really didn't expect that. We were going to do modems with MSN. And so they said, oh, wait a minute. We'll just stick a network adapter in our Windows machine, and it's now a networked operating system. Just as good as that UNIX. We know how well that worked out initially. Then of course they were unhappy with PDAs because there was the Palm and the Scion and the various PDAs. And they said, oh, gee, batteries, that's a problem. Oh, wait a minute. We'll just - we'll do Windows CE. And of course we know how well that worked initially because it was a huge power hog and space hog. And again, Microsoft just sells what they've got in whatever package it needs to be crammed into.

Leo: Yes. And I don't blame them. That's their job. But you don't have to use it.

Steve: There are solutions, however, which are intrinsically wrong, and Windows always seems to step its foot there. And then finally the embedded application market happens, that microprocessors are being used in dishwashers and microwaves and all kinds of consumer products and, god help us, cars. And Microsoft says, oh, hmm, what have we got? Oh, we've got Windows. Well, let's call it Windows Embedded. And the good news is

apparently they have not succeeded in the nuclear reactor market so far. Which we'd really rather have them just stay out of.

Leo: Well, not so fast, Steve Gibson.

Steve: What?

Leo: Question five from Daan Dingjan - I don't want to say his name "Don Dingdong," but that's what it sounds like. Daan Dingjan in the Netherlands spotted Windows in a nuclear power plant: Steve and Leo, love your podcast, listen to all the episodes. I live in the Netherlands, so I have to wait till Friday morning to download it, just in time for any weekend trips in the car. That's good. That gets you - this one will get you all the way across Holland.

Anyway, the last two episodes I've been hearing your worries about Windows being used to run critical systems. I'm sure you'll be happy to know that, yes indeed, a nuclear power plant in the Netherlands is run on Windows. During a tour through the control room of the facility in Petten, I recognized Windows on the monitor. From what I could tell, it was NT4. Of course I asked whether this was actually used to run the plant or just for administrative tasks. Without any trace of worry they replied it was used to run the plant. When I expressed my concern and skepticism, they said, oh, don't worry, it's been running a long time. It's completely secure. If it ain't broke, don't fix it, was the gist of their reply. Oh, dear.

Steve: Well, the good news is it was NT. I mean, the original NT, which is actually 3.5, I think...

Leo: Yes, that's right, yeah.

Steve: And even 4, those were very strong, bulletproof systems. Now, you don't want to open metafiles on those.

Leo: Well, and they came out really before the Internet was huge. They weren't really designed to protect from the Internet.

Steve: Right. And, I mean, I'm sure that Microsoft has nothing that will update NT4 dynamically.

Leo: No, no.

Steve: On the fly. So again, that kind of sort of pre-Internet, really, I mean, the architecture of NT4 didn't have the bells and whistles and fluff and three different APIs...

Leo: It was a real business operating system.

Steve: ...browser du jour and everybody fighting over it the way Windows does now. So I would say please don't change it. Don't go to XP or Vista or Windows 7, you nuclear power plant people. Just stay with NT4.

Leo: And don't connect to the Internet.

Steve: Exactly.

Leo: And they probably don't need to patch it at this point because that's it. You're done.

Steve: Yeah. And I actually - I keep several friends back on Windows 98 because it's so old now nothing infects it. It's like a different DNA. It can't get the diseases that Windows does.

Leo: Really. Is that true?

Steve: Yeah. Yeah, none of these things - all of these things are all about the 32-bit environment and the new architecture. They don't affect 98 at all.

Leo: Well, unless it's like a metafile exploit or something like that, that happens to take advantage of code that has been running since '98.

Steve: Good point.

Leo: And of course then you know it's not going to be patched, either. I don't know. I think Windows is a great choice if you're an accountant. If you're working in a business. We use Windows here on our office machines. It's a great office operating system.

Steve: Well, it's a consumer operating system. It's fantastic for...

Leo: But it's not a good home operating system.

Steve: It's for consumers. It's just not for industry. And it's gotten used for industry, as we're going to see in some of the follow-on questions here.

Leo: Oh, here you go. This is one from, let's see, Lucas Qualls in Jonesboro, Arkansas. You started something.

Steve: Yeah.

Leo: He says: I work at WalMart while I'm attending college, where I'm hoping to graduate with a degree in computer information technology this summer. That's great, Lucas. We need more smart people listening to Security Now!. I'm writing because over the past few episodes I've heard you and Leo talk about Windows being the underlying technology in kiosks. Well, I just wanted to let you know that, if you have ever seen the self-checkouts at a WalMart, they're running Windows XP. My store doesn't have them anymore, thank god. They were a total pain. But when we did, I saw them reboot several times. And it's, yeah, sure, it's Windows XP booting up. And then a program loads at boot time to run the kiosk. You just drag it into the Start menu. An interesting side note is that they run a terminal emulation program that allows them to access the proprietary system that the other WalMart cash registers connect to. Yeah, it's often you see kludges like that where it's kind of a text-based system.

Also recently they took out the reliable, quote, "old-timey" ATM that we had in the store - you know, the kind that has the green text and a screen and no graphics, the kind that just works - and replaced it with the WalMart Money Center Express. We got word that it was supposed to be the most amazing thing in the world. That's how this new technology is always sold.

Steve: Uh-huh.

Leo: It has an ATM built in, but it also allows customers to buy money orders, purchase and reload gift cards and so on. I've seen them that sell stamps. Well, it'd be a great thing if it actually worked. However, you guessed it, the new system runs, when it does, on Windows XP as well. And not only that, the thing doesn't work at all half the time. Things on it are constantly breaking. It's always needing somebody to come out and fix it. It has to be serviced by NCR at least once a week. I personally never used it because I just don't feel comfortable using my ATM card at a terminal I know is running on Windows, and not even running well. So this is just another example of how stupid some people can be when designing things. Feel free to use this on the podcast. I don't even care if you mention my name or location. He's in, like, the home office practically. They could come out and get you.

Steve: Well, again, I'm at the point I understand I'm beating a dead horse. I want to get this out of my system. Our listeners need to know that we won't keep this up. But mark my words, this issue of Windows being used in mission-critical embedded applications where it absolutely should not be used is going to be something we're reading about in the future. This is bad behavior. There are fundamentally sound places where Windows should be used, like on all of our laptops, and absolutely places where the ease of use is, like, the first concern. Because sometimes you ought to have a certain minimal level of expertise before you're allowed to implement important systems that can affect all kinds of people. And anyway, I'm - good thing I don't have a cuff hooked up to my arm measuring my blood pressure when I talk about this stuff. Ugh.

Leo: Barry Burton in Scotland wonders about WiFi PSK disclosure rules: Hi, Steve. I have a question about how a computer knows it's safe to disclose a preshared key to a WiFi access point. Now we're talking. Here's a good question. No more Windows.

What made me think of this is on my Linux machine - I use Arch Linux - I define the WiFi connection I want to use by creating a brief text file - this is how you do it in a real operating system, kid - a profile in /etc/network.d. That profile is used by the netcfg program to connect to an access point. In this file I list three lines: SECURITY="wpa"; ESSID="myssid"; KEY="mypsk". So in this profile I'm identifying my access point by an SSID string and nothing else.

Say that my access point's down for whatever reason. What's to stop someone else creating their own access point with the same SSID, then recovering my secret PSK - preshared key - when a computer of mine tries to connect? If they recovered this information, then they presumably would have full access to my access point whenever it came back in service. In other words, could you spoof the SSID and get the preshared key?

Steve: Yup, that's what he's asking. And the cool thing - I liked this question because it evidences a little bit of misunderstanding about the nature of preshared keys. But it's also the thing, the reason that they are so strong and the reason I like them so much, is that it is not the case that the client machine provides its preshared key to the access point. It's specifically called a "preshared" key, meaning that it's been shared with the access point prior to the connection. As part of the configuration process at each end they both receive the same key. Thus they are preshared.

So the access point knows the key ahead of time that it shares with the client. And they don't even bother with anything except both of them encrypting everything they send using that key and attempting to decrypt everything they receive with the key. So no part of the key, none of the key ever goes in the air. It's only the use, the result of using the key that is sent back and forth. And the nature of ciphers, contemporary ciphers, are such that none of the result of the cipher leaks information about the key that was used to produce it. That's certainly important as you wouldn't want to be able to reverse engineer the key from the results.

The only way to do that we've talked about over and over and over. It's the so-called brute-force attack, where you just guess keys, checking the results to see whether you guessed correctly. And the strongest ciphers are those that have long enough keys that guessing is the only way to obtain any knowledge about the key. And getting close doesn't count, that is, if you're off one bit in the key, it still produces something that looks just as bad as if you're off all the bits in the key. So there are criteria for this. But all of our contemporary symmetric ciphers meet all of those criteria.

So, I mean, preshared keys are so strong, I mean, stronger even than asymmetric keys, stronger than public key technology. That's one of the reasons, for example, I'm not going to use public key technology in my VPN, because something like a breakthrough in factorization we've often talked about, that breaks public key technology. It does not break symmetric key technology, which is fundamentally simpler. But of course it's got the problem that you need to preshare the key. Many situations, like with a VPN, that's practical. But situations, for example, on the Internet, where you're connecting to a remote server that you've never connected to before, you can't have a preshared key.

So that's why we need public key technology, to allow a temporary and ephemeral key to

be shared. So there's a need for public key technology. It does something that private key technology, symmetric key technology, does not do. But it's not quite as strong as just having preshared key technology. And in the case of a WiFi network you're able to just go manually configure the access point, give it the knowledge to decrypt whatever it receives, and so that way at no point is the key being exposed.

Leo: It's never transferred in the clear.

Steve: Right.

Leo: But so what are you going to use, if you're not going to use public key in your VPN?

Steve: Just preshared.

Leo: Preshared keys, okay.

Steve: Yup. And they're arguably stronger.

Leo: A preshared key could be a one-time pad, too; right?

Steve: Well, one of the things that you never do is you never actually use the key itself in order to do the encryption.

Leo: Right.

Steve: That is, you create the notion of session keys.

Leo: Right.

Steve: And so the master key is never itself exposed, but rather you come up with a process of creating session keys. In fact, the CryptoLink tunnel is going to be continually rekeying. It will always be rekeying. Probably no more than, like, every minute it will automatically negotiate a key, and both endpoints switch to it. That provides something else called Perfect Forward Secrecy, where even if you - it's not possible to decrypt it. But even in the event that you did, you wouldn't get all of the session. You would get just a chunk of it. So it's, you know, a little going overboard, but why not?

Leo: Yeah. Very interesting stuff. I love crypto. It's one of my favorite topics.

Steve: And it's just spectacularly great.

Leo: Yeah. Little too much math for my limited...

Steve: Well, that's why I wanted - I want to do CryptoLink so I get to play with this stuff.

Leo: Yeah, I bet.

Steve: Deliver some value at the same time.

Leo: It's a little arcane, and it's also very, I mean, you could tease it all out logically. But boy, it would be hard to do it in a vacuum.

Steve: Yup.

Leo: You really need to know - you need to learn something, too. An anonymous listener wonders something I wonder every day. Why is the Internet still up? Thank you for the informative and entertaining podcast. I have a question with regard to Conficker, botnets, and the risks they present to cloud computing and the Internet in general. If a criminal organization has control over that many machines, don't they have the ability to take down any website on the Internet? And if so, wouldn't extortion be another method of gaining money? Has this happened? And is there an effective countermeasure to a massive DDoS attack? Continued success in your endeavors.

Steve: Well, certainly. It's happening all the time.

Leo: It's happening today. Right now.

Steve: DDoS attacks, yes, DDoS attacks are now at an epidemic level. And it's because they make money. It's because it's no longer the case that script kiddies, back ten years ago they would DDoS their friends in order to knock them off of an IRC server in order to sort of take over and become the IRC operator, sort of playing in an electronic version of king of the hill.

Now it's all about money. It's about extorting, for example, gambling sites are often extorted because it's critical that the site be up during a horse race or a boxing match or whatever the site is focused on and accepting bets for. So the extorters will say, look, we're going to knock you off the 'Net during this important time unless you pay us this ransom. And oftentimes the first time the gambling site will say no. They're knocked off the 'Net. It costs them a huge amount of money. The second time they say, well, gee, can we get a discount for a quantity? A quantity discount?

So, now, one of the problems is following the money because, in order to get paid, money needs to be transferred. And that's of course the Achilles heel of this. It's very easy for attackers to be anonymous in their attacks and in their threats. But somehow money has got to get from the source of the attack or the target of the attack who's said, okay, I'm going to pay for, quote, "protection," unquote, back to the people who are doing the extorting. And that's sort of the weak link in them maintaining their anonymity. And there are various vehicles that the bad guys have used to try to get around it.

But the Internet as a whole is still up because, big as these botnets are, the Internet is even bigger. And the whole notion of a Denial of Service attack is a focusing of bandwidth on a single spot. I've likened it in the past to taking a magnifying glass, and first you put your hand out in the sunlight, and it's fine. Feels warm, but not a problem. If instead, though, you take a magnifying glass and hold it the proper distance over your hand, the same amount of sunlight that is being collected by that lens that was fine when it was falling diffusely on your hand, it is now focused into a very bright spot of burning flesh.

Leo: Yow.

Steve: And that no longer feels good.

Leo: No.

Steve: So a DDoS attack is the same thing. It's a whole bunch of bots, individually not very strong. Collectively, aimed at a single website, that site is down. Nothing can defend against contemporary bot attacks. But while they're attacking that site, they're not attacking any others.

Leo: In theory, though, they could attack the Internet as a whole by attacking the name servers, the master name servers.

Steve: Yes, there have been attacks against the Internet's own infrastructure, that is, for example, the DNS servers is a key. However, many things make that more difficult. Even though there appear to be 13 IP addresses for, for example, the root servers, there are actually many more machines than just 13. And they use complex routing technologies in order to route traffic to the nearest root server, even at a single IP. So when bots think they're all attacking, even if they were all attacking a single IP, their traffic is automatically diffused across the Internet, so they're not all actually attacking a single location. So that's one thing.

Also, all of this information in DNS is cached, often for a day or more. So an attack for an hour would hurt a little bit, but not tremendously. And there's lots of root servers. So you really need to hold them all off the 'Net for a long period of time. And that, due to the distributed nature of the root servers and the fact that we've got caching and that there are 13 IP addresses, all that need to be under attack, that's a huge job. And it's not clear really what benefit there is. I mean, if you had pure malice, then you could say, okay, we're going to try to do that. But really these botnets are now profit centers. And so they're looking for someone that they can extort in order to get money.

Leo: There's a movie plot there somewhere. You know, a plot to take down the Internet. I mean, that would be - the old days the bad guy, the Bond villain would be aiming a nuclear weapon at New York City. But in the modern days the extortion is I will take down the Internet unless you give me one million dollars.

Steve: Well, I heard you talking during - I don't know if it was another podcast, or I think it was something with Amber recently. And you mentioned that your son was home, and he called you in a panic, and he said, Dad...

Leo: The Internet's down.

Steve: "The Internet is down. Come home now." I mean, we're dependent upon this. Increasingly, every single day, we are depending upon this connectivity more and more.

Leo: Yup. He freaked out. As would I. I'm completely sympathetic. However, I had to finish the podcast before I ran home to save his butt.

Marv Schwartz at Case Western Reserve in Cleveland, another one. You know, that's a great technical school. Great computer science program. Wonders if we're not just choosing the wrong languages. He writes: Hi, Steve. I'm an avid Security Now! fan. You've never discussed how the choice of language in which a system is written impacts the reliability and therefore the security of the system. In fact, since you are an accomplished Assembly language programmer, perhaps you inadvertently promote Assembly language programming. Assembly language, C, C++ all put a huge and unnecessary bookkeeping burden on the programmer and lead to mistakes. And they all provide unnecessary opportunities to clobber registers and memory through bookkeeping errors, bad pointers, subscripts out of range, buffer overruns, and so forth. They require the programmer to allocate and release memory.

Language design that promotes writing reliable software is at least 40 years old. These languages are strongly typed. This immediately eliminates Assembly, C, and C++. Although I've never used it, I remember a colleague returning from a stint at Xerox PARC and commenting that when a MESA program compiled, it would run. So if we're serious about writing reliable software, which is a prerequisite for secure systems, shouldn't we be using languages that help us do that and avoiding languages that invite us to mess up? Is this worth a session on Security Now!? Are you going to write CryptoLink in Assembler? And if so, what are you planning for a UI? Thanks again for a wonderful program. Cheers, Marv. Remember Ada? That was a...

Steve: Ooh.

Leo: Ooh. The Defense Department created that language - it had a horrendous spec - for that express purpose. Ada...

Steve: I don't think anyone ever actually created an ADA compiler.

Leo: It died.

Steve: You talk about the horrendous spec. And it's like, the spec was so horrendous that you couldn't actually make it go.

Leo: But the point of it was to write a provably reliable, secure program.

Steve: Yup. Yup. And in fact MESA was an early ALGOL-like language that PARC designed. And it evolved through several levels. And in fact it was - there's parts of it that were - is the basis for Java. Okay. So Marv's right. There's nothing more dangerous than Assembly language...

Leo: Because you're programming to the bare metal.

Steve: Yeah. You can do anything you want to. And sometimes you do those things inadvertently. The same is with C and C++. The reason programmers like C and C++ is the fundamental power that it offers, the fact that it deals with pointers with abandon. And, you know, you can point at anything you want and do pretty much anything you want with it. It is very powerful. There are definitely languages which wrap the programmer in layers of prophylactic protection. And programmers don't like that because they don't feel robust and powerful, and that's what programmers want to feel like.

Leo: Well, they feel like the compiler is nitpicking all the time.

Steve: Well, and it is. But it's also keeping them out of trouble. It also is generally introducing enough overhead that they're not going to have the speed that they would if they were able to program to the bare metal. So I completely agree that this is a tradeoff. I even agree that you could argue we're making the wrong choice. The solution, however, for example, take NASA, where the code that's being used in the shuttle absolutely has to be bulletproof. I mean, it absolutely can't suddenly be crashing and needing a reboot in the middle of a launch sequence, when it's up in the sky and under thrust. Their solution is inspection. Well, testing and simulation and inspection. They just pound on it and pound on it and make changes very carefully and really look at it carefully and test it and perform regression analysis, rather than using languages which are unable to create some sorts of problems, because they recognize, well, there's still other problems. I mean, sure, maybe it's not a buffer overrun. But it's a plus that should be a minus. I mean, if the programmer makes a mistake, it's still a bug. And so they're working on pounding it out just literally by sheer force of will and brute force and a methodology that allows them to be as sure as they can that this thing is working. The problem is, oh my goodness is that expensive. It is phenomenally expensive to produce software that way.

Leo: How do you do it?

Steve: Well, I mean...

Leo: A lot of human checks, or...

Steve: Yes, it's a lot of people involved, and really good people who realize their reputation and the lives of the astronauts hang in the balance.

Leo: And they just comb through the code by hand?

Steve: Yeah. And they simulate it, and they look at - they've got a whole procedure and a structure for making sure this is as good as it is humanly possible to produce. And the problem is it's insanely expensive. And it's why commercial code is not produced that way.

Leo: Well, I mean, there are some little things you can do even with C, like don't use strcpy, use strncpy and things like that, that eliminate some buffer overruns. But you made a great point, which is that you cannot eliminate all programmer errors with any language.

Steve: Right.

Leo: And so it gives you a sense of false confidence to use a so-called "safe" language because you still have to check it.

Steve: Yup.

Leo: So that's a really - it's an excellent point. I can also see why Microsoft, having spent years building a code base of C and C++, is not very likely to convert it all to MESA. That's called rewriting 50 or 100 million lines of code.

Steve: Yeah.

Leo: Charles Palen in Norwood, Massachusetts works in the museum industry and explains why they use Windows: I listen to your show every week. I've been a long-time listener. Thanks for your hard work, blah blah blah. Writing in response to Episode 192 where you discuss the inherent problems with running Windows on kiosk and museum systems. I work as an interactive developer for a company called Boston Productions. We actually build and install museum and visitor center exhibits. I have previous work experience in corporate and small business IT.

It was always a mystery to me why museums, signs, and kiosk systems run on Windows until I started working in the industry. There are several reasons, including total cost of ownership, IT support, dual display support, driver support. My boss

recently wrote an elaborate article about this at backroom.bostonproductions.com. To summarize, the major reason we use Windows is touch screen driver support, which is terrible even in Windows across different overlay vendors, and multiple display support. How easy do you think it would be to configure a touch screen in a 1366x768 vertical resolution with dual monitor support so it can be viewed on a KVM from the exhibit machine room if the machine were running FreeBSD?

Although I'm a longtime Linux user and utilize FreeBSD with dummynet to simulate network lag when doing network programming, most of the museum and creative design industry are Mac users. It's unfortunate, but the vast majority of the people working in our industry simply don't have the computer skills needed to use a stripped-down OS for exhibit deployments. Everyone in our office uses a Mac except my boss and me, who are the programmers in our company. We utilize Windows Vista on our development machines because we need access to the Flash IDE, multi-monitor support, and many other features.

So what he's I think really saying - by the way, Macintosh supports the Flash IDE and multi-monitor support. But what he's really saying, I think, is that they want to use cheap, off-the-shelf components in these devices, and only Windows supports all that.

Steve: Yes. And that they want development to be easy. And Windows...

Leo: Yeah. And developers know Windows.

Steve: ...is easy to develop for. So I liked this, and I wanted to share this with our listeners because it does represent sort of an alternative view, that is, the good news is, this is not nuclear reactors. And you can imagine that a museum doesn't have an unlimited budget. And they're probably being asked to do a lot for not much money. And if the machine crashes when someone presses a corner of the screen that they didn't anticipate, okay, whoops, pull the plug out of the back, count to ten, plug it back in again.

Leo: That usually works.

Steve: No biggie. So again, here's an application where I don't have a problem with a consumer operating system being used because the hardware is cheap; the expansion is available; it's going to be compatible enough; it's going to do the job. And if it doesn't, if it stops working, someone'll say, oh, look, the screen froze. It's like, okay, fine, pull the plug, plug it back in again.

Leo: Reboot.

Steve: No problem.

Leo: So, yeah. And it's not - it's an aesthetic thing to see a Blue Screen of Death or a Windows error message on a kiosk or a big billboard in Las Vegas. But that's only an aesthetic concern. I mean, there are other concerns, as well.

Steve: Right, exactly.

Leo: I understand that. I'm glad you wrote, Charles, because that does explain it a little bit. Dan in San Diego wonders about the security of VPN solutions. How dare he? I really enjoyed your discussions on SSL, or I'm sorry, TLS. One question, though. How do government organizations, or individuals for that matter, spy on people who use VPNs? I know it's possible. Do they need the assistance of the company providing the VPN service to that customer?

Steve: This was a great question because it deals with one of the fundamental issues of security that I want to continue to remind our listeners about. And that is, you need to appreciate and hopefully understand what it is that your security is protecting you from. A perfect example from earlier this hour, Leo, was this issue of the Windows firewall. You know, the assumption is, oh, turn on the firewall.

Leo: Right.

Steve: But whoops, well, wait a minute, it's not protecting us from LAN-based threats.

Leo: Right. Had no idea, yeah.

Steve: Only from WAN-based threats. So similarly a VPN, a Virtual Private Network, like SSL or one based on SSL, or even my CryptoLink forthcoming product, what it does is, it is protecting the link. It's protecting the connection between the two from attack, modification, eavesdropping, or any leakage of information of any kind. But, for example, if a protected - if a machine using the protected link had a keystroke logger on it, then that's not the VPN's job. That's not the SSL connection's job. The keyboard could still be monitored and filtered and checked for whatever, completely separate from the fact that the connection to the Internet, if that's what the SSL VPN is being used for, is secure.

So again, the way government organizations or individuals could spy on someone using a VPN is to install a little hardware keystroke logger in the connection between the keyboard and the computer. And it will gleefully suck in and record all the keystrokes and even get the protection of the VPN when it tries to send them back to the mothership. So again, it's the case that you can rely upon the security technology, whatever it is, given that it's been properly implemented and designed, to do what it's supposed to do. But it's also equally important to understand what it doesn't do, what it wasn't intended to do. And important not to assume that you get total protection for something that never said that's what it was going to offer.

Leo: I never said I'd do that. Finally, the last question, and it comes from the

islands, from Jamaica. Andre says he knows something about Windows and ATMs: Hi, Steve and Leo. I can relate to the surprise Paul experienced upon discovering that ATMs at his bank ran Windows XP. I always assumed they ran a special, robust, embedded program or operating system, or at least a customized flavor of UNIX on these machines. Needless to say I was very surprised last year when I started working as a software engineer for a company that sells and supports ATMs, only to discover that the majority of ATMs run Windows XP. It turns out that the manufacturer whose ATMs I work with used to run OS/2 on them. When OS/2 went under, they moved first to Windows NT, then to XP. They sold that move to Windows based on ease of software development, among other things.

On the security side of the equation, these ATMs run in a somewhat isolated environment network-wise. They're not connected to the internet. No personal data is stored locally. There are many levels of encryption, including hardware-based for really key stuff as well as on communications. And for the record, no, Leo, we don't write ATM software in Visual Basic. There are also many pieces of software that manage failures, crashes, Blue Screens of Death and the like.

That being said, personally I don't believe anything or anyone should run Windows except perhaps the Death Star. I was going to say the Borg, but they wouldn't be that stupid. So yes, it's a bit unnerving that Windows pops up in places which are obviously bad ideas. It's getting back to that same thing. We want to run a commodity hardware. We wanted developers to know how to write for it. He didn't say what they use, but I presume it's Visual Studio and C-Sharp or C++ or C.

Steve: You notice one thing we have not seen is anybody saying we're running on Vista.

Leo: That's surprising to me. I wonder why that is?

Steve: Oh, goodness.

Leo: Well, they're just slow, they're always slow to move to the new OS. But XP is now seven years old.

Steve: Please be slow. Please, well, and of course what'll happen is, as we know, Microsoft is not going to be letting people use XP forever. Microsoft is shutting the door, and they're going to stop doing security updates, and they're going to abandon it, basically, and force everyone to move forward. So at some point all these systems that are running XP will probably be migrated to - I hope they skip over Vista. Maybe they'll go to Windows 7, and not soon. Please, please, please let's give Windows 7 a year to settle down and get a first few service packs under our belts.

Leo: Yes.

Steve: Okay. That's the last word on embedding Windows where - application and, well, application-prudent, application-proper use of Windows. We will move on. But again, mark my words listeners, this is going to be something that bites us at some point.

Leo: Not the end of the story by any means.

Steve: And you heard it here first.

Leo: So what is next week, Mr. G?

Steve: I don't know.

Leo: Ah, how exciting.

Steve: See what comes up. I've got a whole bunch of list of things. I'd love to talk about the DNS benchmark, but I don't think I'm going to quite have all the documentation and everything pulled together by then. So probably three weeks from now we'll do that, and we'll do something next week. Not sure what.

Leo: You know, I should mention that in a couple of weeks, on May 24th, we're going to do a TWiT with Dan Bricklin, the creator of VisiCalc. And you're welcome to join us, if you'd like.

Steve: I'd love to.

Leo: I think it's going to end up being a computer history TWiT.

Steve: Dan wrote something called DBD, Dan Bricklin Demo.

Leo: Yes.

Steve: Or, shortened, also known as just Demo. The original one was just a text screen slideshow where it was basically a text screen editor where you could edit characters and colors and so forth, and then you were able to step through. And the idea was that back in the day, for example, a program like VisiCalc was also text based. VisiCalc ran on a text screen. And word processors did, and many programs did. So he wrote this - basically it was just a textual slideshow to show someone, to demo - oh, it also had, like, macros and things. So you could kind of wire it up and make it sort of look like it was a live-running app, when it fact it was just the screens. I used it to design SpinRite.

Leo: Really.

Steve: The basis for all of my screens. I have DBD files for many versions of SpinRite. All along I've used Dan Bricklin's Demo as - basically I never made it run. But I designed the UI because the UI expressed all the features the program would have. And then I

basically wrote SpinRite behind the screen in order to make all those things that I designed visually, make them come alive.

Leo: Well, that's kind of the idea behind RAD, Rapid Application Development, and Visual Basic is you design the screen first. You design the UI first, with no wiring. And then you lift up the hood, and you go, I'll make that button go to there, and this and this and this.

Steve: And you hook everything up.

Leo: You hook it all up.

Steve: Yup.

Leo: So he was kind of ahead of his time. He says he's now doing open source programming in, sorry about this, JavaScript. But JavaScript's actually a great language. It's a very, I mean, well, I wouldn't say it's a well-designed language. But it's a very useful language because you can do so much with this client-side browser stuff.

Steve: Yeah.

Leo: So if I were a young guy starting out in programming, I'd absolutely learn JavaScript and jQuery.

Steve: I think so, too. I mean, it is - well, and the other thing, I mean, we know for example that Java and JavaScript sort of hand in hand, as I understand it, it's the platform for the Pre also; right?

Leo: I don't know about Java. But JavaScript and CSS and, yeah, and this was what Apple did first with the iPhone. They said, well, you don't have to write applications. You can use JavaScript and CSS to create applications. That's how dashboard widgets and Windows gadgets and a lot of the Yahoo! widgets are all created the same way. You can do a lot with this.

Steve: And so essentially you have a universal client base.

Leo: Exactly.

Steve: You've got, you know, everything understands it because everything has a browser or a browser component.

Leo: Exactly, yeah.

Steve: And that means everything can run what you write.

Leo: It suffered initially because the spec was so poorly written, and it was implemented differently on every browser. And you had to write lots of code that said, well, if it's this browser, then do this. If it's that browser, then do that. I think that's gone away. And then use of libraries like jQuery have made it a lot more kind of orthogonal language and easier to understand, easier to learn. I'm, you know, I've been playing with it. I love it as a language. I mean, it looks just like C. If you use C, you kind of know JavaScript. You just have to learn the DOM API, and you're set. But it can also be the source of much pain, as we have learned.

Steve, next week, who knows, a Security Now! episode that could cover anything. You go to GRC.com, you will find of course SpinRite there. That's Steve's incredible program for disk recovery, disk maintenance. It's a must-have. If you have a hard drive, you need SpinRite. You'll also find lots of free stuff there. Steve has so many great free security applications like ShieldsUP!, DCOMbobulator, Shoot The Messenger. Also some fun tools like Wizmo, and soon some new stuff coming. Of course transcripts of the shows are there. You'll find 64KB and 16KB versions for the people who don't have all the bandwidth in the world. It's all there at GRC.com. I didn't ask you, did you go see "Star Trek" yet?

Steve: I'm doing that right now, as a matter of fact. This afternoon. This is the first of the Star Trek movies that I have not done the whole stand in line, see it on opening day routine. I guess I am getting a little old. Actually I just - I didn't want to fight the crowds. And I knew that I'd be able to go Wednesday afternoon...

Leo: Perfect.

Steve: ...after recording, when the theaters are quiet, and slip in and really enjoy it. And boy, I mean, the reviews have been great. It made $80 million in the first long weekend of opening. And they made more money on Saturday than they did on Friday, which is...

Leo: Always a good sign.

Steve: That's a good sign because it's growing because word of mouth is so strong.

Leo: It's probably the best Star Trek movie, certainly one of the best. And it's just really a pleasure. So I think you'll love it. I'll be very interested in your reaction next week.

Steve: I'll tell you next week.

Leo: Thanks, Steve. We'll see you then.

Steve: Thanks, Leo.