SECURITY NOW!

Transcript of Episode #199

"The Geek Atlas," IPv6 & Non-VPN

Description: This week Steve and Leo explore three different topics: a terrific new book of interest to geeks or non-geeks alike; the still-questionable future and operation of IPv6 (the next version of the Internet protocol); and Steve's novel idea for making secure TCP connections across the Internet without using a VPN tunnel.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-199.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-199-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 199 for June 4, 2009: "The Geek Atlas." 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 secure - privacy online, SSL, VPN, all the acronyms. Here he is, ladies and gentlemen, from another acronym, GRC.com, the Gibson Research Corporation. It's Steve Gibson. Hey, Steve.

Steve Gibson: You know, this week we could also say that it's all things geeky because it certainly is. But we have something especially of interest to geeks this week.

Leo: Oh, good.

Steve: And I want to cover - I want to do something we haven't done before, and that is talk about three sort of different things, sort of smaller things, rather than just one central topic. So...

Leo: I like that. Kind of a geek potpourri, a security potpourri.

Steve: Yeah, that's - interesting things, just some things that have sort of been on my radar recently. We will have some security news, a little bit of errata, and then we'll get into it.

Leo: Very good. So before - let's see. Yeah, let's talk about the security news first.

Steve: Okay.

Leo: I want to send you a link to a great conversation, hacker conversation. I'll send it to you for next week. About security.

Steve: Oh, okay, good. But we do have a number of things that have happened since we last spoke. You may have been aware or may have heard that Apple had a QuickTime flaw which they have now addressed.

Leo: They patched, yes.

Steve: I just, when I turned my Mac on, there were three things it was updating. I don't remember whether QuickTime was one of them because my attention was distracted because that band program, whatever the heck it is, oh, Garage Band, was 108MB, and I don't even know where it is or what it does. But apparently it needed some updating. So I guess you don't want your banjo to have a - to blow up on you or something. I don't know, who knows. So my Mac has the current version of Garage Band, for all the good it's going to do it. And presumably this Apple QuickTime flaw.

There was, in the .PICT handling, there was a flaw in a - there was a 16-bit length attribute that was mishandled that will allow people to generate a malicious QuickTime file which can, naturally, take over your machine remotely. So you want to make sure that you get yourself updated on that. Once again, PDF is in the security news. This time BlackBerry. They've had, like, three or four problems in, like within the last year, BlackBerry has, with their PDF handling. And there's a new one which allows you, or allows bad guys, to malform email containing a PDF file, send it to a BlackBerry, and take it over remotely.

Leo: Wow.

Steve: So updates are available. Before they were available, BlackBerry was very forthcoming and said, look, we want to acknowledge this vulnerability. They got some kudos for being really so upfront about it. And now they've got it fixed. So if anyone is, like, using BlackBerry in corporate environments, they want to make sure that their corporation has updated the server, which apparently is involved in the parsing and the generation of PDFs as they're sent out to the BlackBerries. So that's been fixed.

We have also a zero-day Microsoft DirectX vulnerability in DirectShow, which is a component of DirectX, the DirectShow QuickTime parser. Now, there is no fix for it yet. It does not affect Vista and Server 2008 and Win 7. So it's only the down versions of Windows - Windows 2K, XP, and Server 2003. Microsoft doesn't have a fix for it. Oh, and because it's zero-day, remember, that means it is being actively exploited now. And again, it means that the way it was discovered was people's machines were crashing or getting code executed on them remotely. And so this is one of those things where you just visit a website, and DirectX can trigger, that is, the JavaScript in the website can trigger the playing, without any user interaction, of a QuickTime file which executes this code which is vulnerable and allows somebody to do malicious things on your machine without you taking any action other than just viewing it through your web browser. Now, the good news is, Microsoft has deployed their "Fix it" system. I think I've heard you and Paul talking about this, kind of this Microsoft "Fix it" thing where...

Leo: Yeah, it's kind of cool. If you go to some of their support pages, their knowledge base pages, there's like a button that says "Fix This." And usually, I think, I suspect, it's just a registry patch. But they might do other stuff, too.

Steve: Well, and so this is a perfect lead-in. Thank you, Leo, for this. If our listeners go to support.microsoft.com - you can do this right now, Leo, and you'll see.

Leo: Okay.

Steve: Support.microsoft.com/kb - as in knowledge with a silent K, knowledge base. So it's /kb/971778. That's the magic number for this. So support.microsoft.com/kb/971778. That will take you to a page acknowledging the problem. And there are two big buttons there. One is the disable the QuickTime parser, which is where the problem is, for DirectShow. And the other will reenable it if you want it to be enabled. So you can also download these files rather than executing them. For example, if you had a computer that was temporarily disconnected from the Internet or for whatever reason you couldn't go to the site, you can download them as files and then run them standalone on other machines.

Leo: Interesting, they're COM files.

Steve: Yeah. So anyway, so this is a problem that exists now, has not been patched yet. One imagines, here we are, what are we, we're just shy of the second - the second Tuesday of the month.

Leo: But no, it's a QuickTime problem. So is it really an Apple patch or a Microsoft patch?

Steve: No, Apple does not have the problem, this problem.

Leo: Oh, I see.

Steve: So it's a Microsoft implementation. It's like...

Leo: It's DirectShow is the problem.

Steve: Yeah, it's DirectShow parser that parses QuickTime content.

Leo: Got it, I see, I see.

Steve: So Apple does not have it. So, let's see, the first Tuesday of the month was, you know, based on the timing of this, this podcast will be aired on June 4th. So that would have been the 2nd. So that means next Tuesday is the second Tuesday of the month. This has all just happened. So it's not clear to me, I mean, we'll see whether Microsoft is able to fix this in time, get this into their patch update. They don't have much lead time to make that happen. So it's not clear whether they will or not. So I'm, you know, certainly you do run across QuickTime videos on the 'Net. And once you do this you will be disabling QuickTime playing for your browser on the 'Net under Windows, which you can choose to do or not. You can certainly reenable it afterwards. I imagine that Microsoft's update, when they do it, will probably reenable it for you. They may take responsibility for undoing the disabling that they're now recommending. But anyway, again, support.microsoft.com/kb/971778 to disable this as an interim measure because there is no fix for it. It is a zero-day exploit being actively exploited to take over people's machines.

Leo: Very interesting.

Steve: And then the final thing, I'll bet you've probably had this run across your radar, Leo, as a big kerfuffle has arisen. I think maybe Brian Krebs, who I talk about from time to time, who writes a security column for the Washington Post. He may have been the first person to bring this up. And that is that Microsoft was found to be surreptitiously installing a Firefox add-on for .NET.

Leo: Oooooooh. Oooooooh. Oooooooh.

Steve: And get this. The Firefox add-on that they installed as part of their regular monthly patch, that is, the second Tuesday of the month deal, when they did, they introduced the .NET Framework 3.5 Service Pack 1, which was back in February, just without telling anyone they slipped this into the Firefox add-on list. I have seen it for months...

Leo: Yeah, me, too, yeah, yeah.

Steve: ...because it's been there for a while. It's like, oh, I mean, I've already given up because it's like, okay, either you're trusting what Microsoft is doing or you're not.

Leo: Or you're out of luck.

Steve: Exactly. I mean, if you don't, then go to Linux or a Mac. But it literally, this add-on establishes in Firefox, get this, "The ability for websites to easily and quietly install software on your PC. So the problem here is this is why you're using Firefox, is that you don't want websites to have the ability to easily and quietly install software on your PC. You've moved to Firefox because you don't want to be using the most historically vulnerable browser, Internet Explorer, in the industry. And yet Microsoft has reached over and added this feature to your Firefox browser without your knowledge or permission to do just that. Now, what makes it even problematical is that the uninstall button is disabled.

Leo: Oh.

Steve: So no one can remove it. Now, now Microsoft is saying, oh, well, I mean, talk about double-speak. I'm going to quote what Microsoft says on their site because they've been slapped so hard and it's raised so much concern that they've now backed off from that. But they say, "In .NET Framework 3.5 SP1, the .NET Framework Assistant enables Firefox to use the Click Once technology that is included in the .NET Framework. The .NET Framework Assistant is added at the machine level to enable its functionality for all users on the machine. As a result, the uninstall button is shown as unavailable in the Firefox add-ons list."

Leo: In case there's somebody else who's using it.

Steve: Oh, well, yes, exactly. It's at the machine level.

Leo: Oh ho.

Steve: Too sophisticated for you to manage.

Leo: Wow, yes.

Steve: "As a result, the uninstall button is shown as unavailable in the Firefox add-ons list because standard users are not permitted to uninstall machine-level components."

Leo: Okay.

Steve: Even though an...

Leo: So if I'm an administrator I could do it.

Steve: ...end-user running the machine got it installed just by using Windows Update.

Leo: Yeah, no problem, yeah.

Steve: That you're being pounded on to make sure is turned on all the time. So they're saying, "In this update for .NET Framework 3.5 SP1, and in Windows 7, the .NET Framework Assistant will be installed on a per-user basis. As a result, the uninstall button will be functional in the Firefox add-ons list. This update will also make this version of the .NET Framework Assistant for Firefox compatible with future versions of the Firefox browser, whatever that means. To properly update the .NET Framework Assistant, this update must be applied while the extension is enabled in Firefox." And it goes blah, blah; it goes on and on.

So, once again, we have a URL where Firefox users can get this. It is support.microsoft.com/?kbid, as in knowledge base identifier. So it's ?kbid=963707. So again, support.microsoft.com/?kbid=963707. That will get you to a page where you can do a number of things. You can manually edit the registry. They've got all kinds of different ways of rummaging around and making this happen. But there's...

Leo: No automatic "Fix it" button, I notice.

Steve: Yeah, there's not the happy little guy with the tool waving at you.

Leo: Sorry, you can't do that.

Steve: However, what this will end up doing is disabling the disablement, which is to say reenabling as it - now that they've been scolded, you can reenable the uninstall and then say thank you, Microsoft, but I would prefer not to have Firefox able to easily install software in my machine so that I'm not inconvenienced with the question.

Leo: This is unconscionable. I can't believe this.

Steve: Yes. Yes.

Leo: I'm stunned.

Steve: This is bad.

Leo: How dare they?

Steve: Yeah.

Leo: In fact, this is exactly the kind of thing that they've been brought to task for by the Department of Justice and the EU. I mean, it's one thing to say, well, you have to use Internet Explorer, we'll going to include Internet Explorer; anther thing to modify other browsers that you use on the system to make them less secure. Am I correct? This makes it less secure?

Steve: Yes, that's exactly what it does. And which is why Brian Krebs, when he, like, I guess a couple people brought it to his attention, and he said, huh? And he looked at it and did the research, and it's like, oh, goodness, I mean, this is really, really bad.

Leo: That's unconscionable.

Steve: And this is - Microsoft has worked to build our trust in the whole Windows Update facility. I mean, as I said, you either trust them or you don't use Windows because we've given up control. They're downloading code and dunning us and punishing us and with red flashing lights and things if we try to take control back from them. And it's funny because a friend of mine this morning at Starbucks came to visit and says, so, are you on IE7? I said, oh, yeah, for quite a while. And he says, oh, I guess - so you've made peace with it? I said, well, the only time I ever run it now is to run Windows Update.

Leo: Right.

Steve: I mean, I'm completely converted to Firefox with, like, zero trouble. So, I said, so kind of. I mean, it's on my machines; and 8 is sort of filtering in to my machines as I think, well, okay, why not? I mean, I'm not using it anymore, so I don't care if Microsoft wants to push IE8 on my machines. Fine, you know, it has no effect on me except for running Windows Update, which insists on running under IE.

So, yeah, Leo, I agree. This is, I mean, this is a breach of trust. The fact that this was slipped in, that it is a software installation shim for Firefox so that their .NET Framework is able to be more pervasive and to run on more websites. And so that website owners are not going to say, well, I'm not really going to update or start using that because, after all, Firefox doesn't support this. Well, Microsoft slipped this in so that it does, even if it's not what the end-user would want. And so anyone with Firefox, if you look at your add-on list, and you've been keeping your Windows current, you'll see this thing sitting there, and its uninstall button is grayed out, preventing you from uninstalling it until you go here, reenable the button, then you can say thank you, but no, Microsoft.

Leo: Now, what do you lose, just out of curiosity, if you do that? I mean, is there anything that I need this .NET bug for? Is this the Click Once thing?

Steve: Well, that's what it is. And the question is, I mean, you can think of it as, like, super-advanced scripting. The question is, for example, and we discuss this often here, what do you lose if you disable scripting? Well, you lose some functionality that may or may not be something you care about losing in return for increasing your security. So hopefully - I don't know what.

Leo: I mean, is there - okay. Yeah, well, for one thing, yeah. There's no question this was a stupid and wrong thing to do.

Steve: Yes.

Leo: I mean, there's not a question about that.

Steve: Without permission. They could have, I mean, look at all the things we do have to give permission for. Every time Microsoft does something, we're having to reverify our license. Yes, I reassert my compliance to your EULA. I mean, often we're being asked to recertify that, yes, we're going to abide by these license terms. It's certainly not out of the question to imagine that Microsoft might say, hey, we want to - we're updating the .NET Framework. It's becoming more pervasive. It's the future. So we want to bring Firefox, which we happen to notice you have on your Windows machine for some reason, we want to bring it into compliance and make sure that things stay synchronized and the functionality that we hope you've become dependent upon will also be present in Firefox as it is in IE. So do we have your permission to do this? I mean, all they had to do was ask. And then people could have said, oh, yeah, I guess I should have that, or not.

Leo: Do you think this is a case of - clearly what Microsoft thinks, I'm trying to put myself in their head, is this is too complicated for our users. We're just going to make this decision on their behalf. And we're not going to explain it because even explaining it is too complicated. So we're just going to do it. We know what's best. We're not causing a problem here. You've trusted us to run your system, so we're going to just do this.

Steve: I mean, yes, you can certainly say that, hey, you know, trust us or leave.

Leo: Right. I mean, well, you have to. I mean, that's - that's the deal.

Steve: Yeah.

Leo: Oh, I just think that stinks. Now, some people have said this is anti-competitive, as well.

Steve: I have to imagine that there was a conference of some length at Microsoft where they decided to do this. I mean, I hope this wasn't something that they did thoughtlessly. So following your logic, Leo, there must have been the argument made that this is something that was in their and their users' best interests to pursue. I don't know enough about the architecture of Firefox's innards to know whether they had a choice of making this visible on the surface of the UI or not. They may have had no choice. They may have preferred to just sort of slip this in as they do in IE, secretly. But it may be that the architecture doesn't allow them to do that, that they weren't able to just deposit this somewhere and have it take action without being visible on the surface. Or they may have felt, shoot, you know, once that's discovered we'll be in even bigger trouble.

So it would have been nice to be asked, and it certainly would have been nice not to have the uninstall button grayed out. Or, if you click it, have them then present a dialogue that says, whoa. You can uninstall this if you want to, but here's what you lose if you do. Instead it just - it appears magically. It's about installing software into your system without you, making it easier to do that. And we're not going to let you take it out.

Leo: Now, I have Firefox installed on my Vista machine, and IE8 installed, and I'm looking in the add-ons. And I don't see anything. I do see some Microsoft stuff, the Windows Presentation Foundation and Silverlight. I think I installed those.

Steve: In Firefox.

Leo: Yeah. Mozilla Default Plug-in, Java Platform, iTunes. It says .NET in the name of it?

Steve: Yes. And I definitely...

Leo: I've seen it on some of my browsers, I mean, some of my systems. But I'm just looking at my Vista system here, and I don't see it. And a couple of people in the chatroom said, well, I don't see it. So I wonder what circumstances - or maybe you have to download a...

Steve: Okay, I'm looking at it.

Leo: Oh, wait a minute, it's in extensions. I'm sorry. It's not in plug-ins.

Steve: Correct.

Leo: I do see it. Ah ha.

Steve: Correct, it's extensions.

Leo: I was looking in the wrong place. As Click One Support. There's no disable button. There's a disable button, but no uninstall button. So I could disable it, but I can't remove it. Prompt once before running Click Once. Report all installed versions.

Steve: Now, that's interesting. Mine is disabled. I must have done that. I had forgotten. Because I'm seeing my Enable button is enabled, and the little popup toolkit says "Enable this add-on when Firefox is restarted." And so I had clearly disabled it, saying - seeing it and saying I don't think I want this, thank you very much. And so, and I restarted the system, and now it's sitting in there. It's not removed.

Leo: It's not uninstallable, but it is disabled.

Steve: Exactly. I can not uninstall it, but I did disable it in the past.

Leo: Very interesting. I don't - regardless, I mean, I guess maybe Microsoft said, well, you can disable it. But I don't want them installing it.

Steve: Yeah. I mean, and we can be grateful that this came up, that they've certainly, whatever decision they reached around the conference room, the discussion that I hope they had, they may recognize now that they went too far, and they won't do something like this again. So we can hope that they learn from it.

Leo: Yeah. It does seem unconscionable. You know, there's supposed to be - maybe that's expired. But after the terms of the settlement with the Department of Justice, Judge Colleen Kollar-Kotelly I remember required a judge-appointed ombudsman in Microsoft, maybe even a committee, watching what they do to make sure they don't do anti-competitive things. I wonder if these people are paying any attention at all.

Steve: Well, I did pick up a little news blurb earlier this week that the EU is not through with Microsoft. They're gearing up, or teeing up, on Microsoft. Apparently what they're considering is requiring Windows to include competitive browsers.

Leo: They're requiring Windows to include competitive browsers. Wow.

Steve: Literally Firefox and Opera...

Leo: Have it built in, good. I think that's not a bad idea.

Steve: And so what would happen is, when you first turn Windows on, they call it a "ballot screen." It comes up, and it says, which of these browsers do you wish to install, and which do you want to set up as your system default? So the EU would be requiring that users who are first turning their machine on in that initial sort of pre-usage configuration phase are actually given a choice, and the browsers are present, and you can choose to install any of them that you like, and choose which one you want to use. And so they're talking about moving much further than they did in their prior work of requiring Microsoft to unbundle the media player.

Leo: Ken Shepardson's saying in our FriendFeed chat room that to decouple the Microsoft-is-bad stuff from the security stuff, well, here's the security issue. I mean, they're installing something into Firefox that allows a website to automatically install software on your machine; is that correct?

Steve: Yes.

Leo: Okay. That seems to me on the surface of it that that's a security issue.

Steve: And that's why - and my complaint is it's why people left IE.

Leo: In the first place.

Steve: I mean, you have to leave IE. You don't - you're using Firefox because you went to Mozilla.org and got it, and you know why you're using it, and it's a little bit uncomfortable because you have to go back to IE for Windows Update and doing things that only Microsoft will allow to happen under IE. So it's like, this is a conscious choice people are making for some reason, probably because they've decided they don't trust Internet Explorer. So here it's Microsoft reaching over into that decision and saying, eh, not so fast. And then I have one little bit of errata.

Leo: Yes.

Steve: And you probably know this, too. Amazon has indicated that my Kindle will arrive next Thursday.

Leo: Oh, that was fast.

Steve: Yes, June 11th, the new DX, the big...

Leo: Now I'm jealous. Now I'm jealous.

Steve: So I won't be able to wave it in front of the camera, Leo, for next week's recording on Wednesday. But the 10th, next Wednesday, is the release date. And so they estimated delivery for the 11th.

Leo: Wow, that was faster than I thought it was going to be. I mean, I got the impression it would be a few months.

Steve: Yes. I'm excited.

Leo: So now your Kindle will be bigger than my Kindle.

Steve: Then I have a real PDF reader, more than anything else. I mean, for the first time ever. Because I'm busy printing paper all the time, printing PDFs. And that's just - that's not why...

Leo: I have to admit, I'm...

Steve: That's old school.

Leo: I'm intrigued because more and more I'm reading books on the Internet, programming documents, that kind of thing, as I'm sure you are. And that would be really nice to have that slightly larger form factor and the ability to put the PDFs on there.

Steve: Yeah, I mean, what you said stuck in my mind. That is, it's the convenience of the smaller size Kindle, being more paperback size, being able to carry it around easily; whereas this is more of a slate. It's like, yeah, we'll see how that is. Because, I mean, my feeling is I like the larger real estate when I'm reading. Seeing more text, paging less often, maybe scaling things, using a larger font in dim light and still being able to have plenty of text on the screen. So anyway, in two weeks I'll be able to wave it in front of the camera and tell our listeners what I think about it.

Leo: Woohoo.

Steve: And I had a really neat SpinRite testimonial lined up, and I don't know where it went.

Leo: [Laughing]

Steve: I mean, I have a whole file of them. But I thought, ah, well, what the heck, I mean, our listeners all know that SpinRite leaps over tall buildings and is faster than a speeding bullet and so forth. So there's my - we'll do the non-testimonial testimonial. We'll skip it for this week because everyone's probably thankful to have a week off from SpinRite testimonials.

Leo: You're preaching to the choir anyway. We all know it's the best.

Steve: It does work. It does the job.

Leo: I recommend it. I recommended it twice on the radio show this past weekend to a couple of different people. I mean, it's interesting, and I guess this is probably a symptom of the fact that these hard drives are getting so much bigger. But I get more and more hard drive failure-sounding questions. Oh, you know, when I try to run this program it takes a really, really long time, and it finally starts, and little things like that, that really sound like bad sectors.

Steve: Right. So if the drive is being very patient, it's retrying and retrying and going around and around in circles, and is getting a sector at a time, reading very slowly, that's a perfect example of something that SpinRite would just make quick work of.

Leo: That's what I said. I said, well, you just run SpinRite. It'll move. It's more persistent than the operating system. It'll get the data if it's at all possible. Even if it takes days, it'll get that data and move it to somewhere safe and then mark that sector bad and unusable. And that's really what you want.

So we both got this book, and I'm really excited about this.

Steve: Well, I didn't know that you knew the author, Leo.

Leo: I've known him for a long time, probably because of our connection because he's a regular in your forums.

Steve: Well, he is. And what happened was about a month ago I got this polite piece of email from John Graham-Cumming, who I've known through the newsgroups for years. I may have referred to him before, back in the Perfect Paper Passwords era, because he was one of the people who wrote a very nice, high-tech implementation of the Perfect Paper Passwords algorithm for a particular platform. And I've sort of forgotten which one now. But so, I mean, but he's always contributing and knowledgeable, and I've just appreciated his presence.

And so this email said hey, Steve, would you - and I'm paraphrasing - would you mind if I posted a note in the newsgroups about my forthcoming book? And I said, what forthcoming book? Well, it's a book called "The Geek Atlas." And it is, essentially, it is describing 128 - and of course I love that, you know, it's 2^7, it's not 100 or 150, it's 128 - locations around the world of technical and historical geeky interest. And what I really like about it is that, well, in fact what I should do is, let me just read, or, yeah, read for our listeners the review that I wrote which I have posted at Amazon. Because the book is now available. It's called "The Geek Atlas." And it's less than 20 bucks. It's 19 something or other...

Leo: What's nice is it's like one of those guidebooks, you know, it's like a - it has lay-flat binding. I mean, it really is like a guidebook.

Steve: Well, and it's published by O'Reilly, so it's got that standard O'Reilly feel to it.

Leo: Oh, yeah, nicely done.

Steve: And so what I wrote was: "This terrific book first came to my attention when its author politely asked whether it would be okay for him to mention it in the technical newsgroup forums my company hosts. I had known of John through his many years as occasional contributor in our forums, though I knew nothing of his being an author. Little did I know. Now, I have a copy, and I love it. When I take it with me to coffee, other regulars who have seen it before grab it if I'm reading something else. We all love it because it is so accessible. And these are people who are not nearly as geeky as I am.

"Opening the book to literally any page pulls you immediately in. Even if you're not a traveler - I'm definitely not - the book is a compendium of bite-size worldwide technical history of innovation and invention in gratifying detail. No single topic is more than four pages long, so you can read many before your coffee gets cold. And you may be ordering a second cup because this book is difficult to put down. You can read by region or scan the table of contents for anything that looks interesting. The Escher Museum in the Netherlands, the Experimental Breeder Reactor No. 1 in Idaho, the Arecibo Observatory in Puerto Rico, the Mendel Museum of Genetics in the Czech Republic, and 124 other notable places and times where something geeky and technologically important happened. I used to wonder how and where the speed of light was first measured. Now I know.

"John has filled the pages, not only with a discussion of interesting brief historical notes, but also with his own diagrams and explanations of every principle and discovery. He has a direct, straightforward and clear writing style. And best of all for geeky readers like myself, he clearly knows what he's talking about, unlike some authors who are disconcerting because you sense that they're not sure of their facts."

Leo: Right.

Steve: "You won't find any of that here. The technical content is precise and will satisfy the geekiest among us. This book would be a bargain at twice Amazon's price of only $20, so think about getting two. Even if you're not a geek, you'll love this, really. And I'll bet you know a geek who would value this just as much."

Leo: Yeah, I agree that it's much - it's great because it's more than a guidebook because you're learning the science or the principle or how Enigma worked and that kind of thing. And so, I mean, look at this, it's got the ideal rocket equation when you go to White Sands Missile Range Museum.

Steve: I know.

Leo: I mean, that's great.

Steve: And, like, details about Foucault's Pendulum that is swinging from the - is the Pantheon? I think it's in France. And, I mean, there was an impulse rocket engine that I was reading about this morning. And, I mean, anywhere you pick it up, it's like, oh, wow, I didn't know that. Anyway, I just - I think our listeners would get a kick out of it. If it sounds like something from our description that you would like, my guess is you will like it more than you imagine you would because it's, I mean, it's really nicely done.

Leo: It's really, it's fascinating. I don't - you don't even have to go to these places, just to read the book.

Steve: I'm not going. I'm just reading the book.

Leo: Yeah. Although it'd be fun to kind of make a geek tour and go to all of these places.

Steve: Yes. John is clearly a traveler because, as I was mentioning to you before we began recording, he'll say in there that, well, and if you stand on this mound and look to the northwest, you can - it's like, okay, well...

Leo: He was there. He was there.

Steve: ...this guy is traveling around and picking up all of these interesting tidbits. But again, it's not just about travel. It's as much, for me, about history of technology, written by somebody, a serious techie, not a writer who's trying to water it down. I mean, there's - it's full of diagrams and explanations of stuff.

Leo: It's interesting, boy, I wonder if he's been to all these - I guess he has been to all these places.

Steve: It reads like he...

Leo: Yeah.

Steve: It reads like he's been there.

Leo: This is a labor of love. I mean, this must have been a book that he's been working on for years, I mean, to go to all these places. Really cool. I mean, this is one of those obsessive things that only a geek could really do justice to. I love it. I mean, here's Eiffel's diagram of the Eiffel Tower.

Steve: Exactly. I just opened "Galvanic Corrosion and Cathodic Protection: The ability of batteries to make electricity by immersing two different metals in an electrolyte. See page X." It's very, oh, and, see, I have a pre-release proof copy, so they didn't fill the page number in. But...

Leo: Oh, I have it, yeah, yeah, yeah, I have them.

Steve: It's very useful. But "Metal objects can accidentally become batteries and end up corroding. This galvanic corrosion is a problem for everything from the Statue of Liberty to a ship's rudder. And pipelines are particularly vulnerable because of their size. The problem can even occur when only one type of metal is present because of slightly differing compositions of the same base metal." I mean, this is really good stuff, Leo.

Leo: Yeah. I'm going to send a copy of this to my father-in-law. He was a science teacher in high school. He doesn't travel much anymore. But he would love the material in this book, I think,.

Steve: "For example, the Statue of Liberty is made from a copper skin with an iron structure. When Gustave Eiffel built the statue, he anticipated that galvanic corrosion would be a problem. So he insulated the copper and iron from each other using the natural plastic shellac. See page" - again blank. Who knows what that one talks about. "Over time, the shellac insulation gave way, and between the metals an electrolyte - moist, salty marine air - was able to create a simple battery." And it goes on. I mean, it just is, I mean, it's a technical atlas as much as anything else. I mean, and then it all ties in to places all over the world where these things happened. Oh, here's Bletchley Park; and here's Instrument Landing System, ILS, British Airways flight training in Hounslow, England. And so we've got little icons for it looks like there's hotels nearby, and then GPS coordinates for everything, and how ILS works, which I never knew.

Leo: Oh, yeah, that's interesting.

Steve: The localizer and the glide...

Leo: You should know how it works. You've got a couple of ILS computers there.

Steve: That's true.

Leo: You might be called into service at any moment.

Steve: Bunhill Fields Cemetery in London, England.

Leo: This is so geeky. I love it.

Steve: Oh, it's Bayes' Theorem.

Leo: Yeah, because he wrote...

Steve: Talking about Bayesian.

Leo: He wrote POPFile. I used POPFile for a long time. I don't use it anymore because I use Outlook. But it was a great - actually I guess it worked with other programs, too. But it was a great Bayesian antispam filter.

Steve: "Bayes' Theorem gives mathematicians a way of updating a probability when new information comes along. For example, say that 70 percent of the pupils in a school are boys, and 30 are girls. The girls have a choice of uniform, trousers or skirts, and the boys just wear trousers. If a mathematician encounters a pupil at random, then he knows that the chance of the pupil being a girl is 30 percent. Now suppose the mathematician, who is deep in thought and staring at the ground, only notices what's covering the pupils' legs. If he sees a pair of trousers, he can calculate the probability that the pupil is a girl using Bayes' Theorem. He's updated his original estimate, which was 30 percent, based on new information, and comes up with the answer of 18 percent." I mean, anyway, this just - it's, like, full of stuff like this, just really interesting. Oh, here's Hooke's Law and Clocks, which is how the escapement works, where, you know, you have a swinging pendulum. And the escapement, every time the pendulum swings the little disk, the ratchet moves one notch; but in the process it also powers the pendulum? So here it's got - we have two pages on all that. It's just - oh, here's Chernobyl.

Leo: I know, there's some really - there's a picture here of the hammer and feather experiment that Apollo 15 did on the moon, to see if Newton - was it Newton? - was right that - oh, I just love this stuff. I just love this stuff.

Steve: Yeah. And so here on the page after Chernobyl, it's potassium iodide and the thyroid. And so what it is about potassium iodide. "One of the immediate dangers after the explosion was the presence of radioactive iodine in the food system. Radioactive iodine, iodine-131, is produced in nuclear reactors in normal operation as a product of the fission of uranium-235. The uranium-235 breaks apart when its nucleus is hit by a neutron, releasing energy and creating new elements from the split-apart atom. The elements typically created when uranium-235 breaks apart in a nuclear reaction are cesium, iodine, zirconium, technetium, strontium, promethium, and samarium." And this goes on. So, oh, my goodness. Anyway, like I said, it's hard to put down.

Leo: It is.

Steve: Just full of cool stuff.

Leo: What a great idea.

Steve: So I wanted to bring it to our listeners' attention. I did post in the newsgroups to bring it to our newsgroups' attention. And I wrote that review that I read for Amazon since. And John...

Leo: Galileo, not Newton.

Steve: Shortly after I posted he says, "I'm speechless with gratitude." And I said, "John, look at the work you put into this thing." I mean, this is an amazing amount of work. So he's earned...

Leo: Yeah, props to John, absolutely. And it is Galileo, of course, who dropped the cannon ball and the feather off the...

Steve: Off the Leaning Tower of Pisa.

Leo: Leaning Tower of Pisa, yeah. Not Newton. Thank you, Dr. Mom. All right. So that's item one of our trilogy.

Steve: Yes. Number two is IPv6. We're now using, and have been since the beginning, IPv4. And in fact in the first four bits of the header of every packet traversing the Internet gives the version number so that software that receives the packet knows what to do with the rest of the bits in the packet, the idea being that the first four bits say I'm an IPv4 packet. And so then the software says, ah, good, in that case I know what all the other bits in the IP header layout are, and I know how to process them. If the packet came and said, hey, I'm IPv5 or something - which actually is a streaming protocol that was developed related to IPv4, and that's why it's not available; and the guys at the IETF who were standardizing on the next-generation protocol, they had to jump over 5 and call this one 6. If the packet said, hey, I'm v6, today our hardware would go, or most hardware, huh?, and just abort.

Leo: Oh, dear, that's not good.

Steve: Well, no, I mean, because - well, okay. So here's what's happened is people from time to time ask me, and I was asked yesterday, if the DNS benchmark program that I've been working on, laboring on for the last many months, which is always almost done, and we're always closer to being almost done, if it would be supporting IPv6. And it sort of makes me just sigh because it is - IPv6 is still so far away that I want to say it doesn't matter.

Now, there are some instances where it does. For example, it turns out that if your stack in your Windows machine is configured for IPv6, then, for example, it'll emit IPv6 DNS queries. I guess my issue is that this doesn't affect end-users today. That is, the whole IPv6. So, for example, yes, on a server-side platform like the spoofability tester, it's necessary for me to be able to accept queries in IPv6 format. But this is not affecting end-users today.

And it's not clear to me, I mean, ever is a long time. But the whole reason IPv6 was created, the fundamental motivation was a concern for the depletion of IP space. We know IPv4 has 32-bit Internet addressing, 32-bit IP addresses. And that gives us 4,294,967,296 different IPs. Now, what's significant in my mind is that today, in 2009, fully 40 percent of those IPs are not even in use. They're, like, remember Hamachi uses the five-dot block of IPs? Because they're non-routable. No one is using them. Somebody has them. But they've never been used. So there's, right there, 16 million IPs that are available, but no one's using them. And there's a whole bunch of other Class A network regions that are the same.

And, for example, there are universities who were early into the game, who got themselves a Class A network, meaning 60 million IPs. And they're not using them because they're assigning their students NAT-based IPs, Network Address Translation-based IPs. But they're jealous of this IP space. It's sort of a nice thing for them to have.

They'd rather not give them back. So instead we're basically changing everything on the Internet.

And, I mean, the reason I'm of two minds about IPv6 is, first of all, it has no effect for anybody today. It's not clear to me that in another 10 years it will. It's already 10 years old. This spec was ratified in 1998. And so it's been 10 years. And there's some deployment of the so-called "6bone," the IPv6 backbone. Normal people have no access to it. There are services that will allow you to create a tunnel through IPv4 to get to it. So you're able to, if for some reason you want to mess around with IPv6, access that 6bone. But there's not much there at the moment because it's not necessary.

I mean, so what happened was that we started this when there was a concern for, like, exponential growth of IPs on the Internet. But the solution turned out to be NAT. So what's happened is that the emergence of Network Address Translation has essentially solved the problem completely, in a way which is compatible completely with everything we have now. For example, many people have networks where they've got a little switch. You know, it used to be hubs, then we went to switches. Well, none of those can work on IPv6 because the switches, as we've discussed in ancient history on this podcast, build a table of IP addresses, and they memorize which IP is connected to which port of the switch. Well, all of that is in hardware, and all of that is IPv4. So that won't work. Maybe you could flash, someday flash the firmware on your router in order to be v6 aware. Right now, routers mess up IPv6 packets completely. They drop them, or they don't even return an error message typically. They don't know what to do. And that causes connection delays for anybody who's receiving these wacky packets, if you're even able to get them.

So when someone says, oh, well, does this - and I know I've kind of got myself worked up into a rant here. But I get the question enough that I just thought it was worth taking some time to say, wait a minute. I recognize I'm a little bit of a Luddite here. I never went to Vista. I'm still on XP. It'll be a year before I go to Windows 7 because I want to let it mature. When IPv6 matters, then it would make sense certainly for the things I'm doing in network space to be IPv6 aware. Certainly when I'm writing CryptoLink I'm going to anticipate that hopefully, over that product's long life, people may be in some place where they need IPv6 compatibility. So I will design 128-bit addresses, that is, an architecture that supports IPv6 from the beginning for that product. But for any kind of application today, IPv6 is just not here. Some statistics are interesting. Believe it or not, Russia has the highest percentage of IPv6 penetration.

Leo: Really.

Steve: And they're at 0.76 percent. So less than 1 percent of machines in Russia are IPv6. Then falling down from there is France at 0.65, the Ukraine at 0.64, Norway at 0.49, and the U.S. at 0.45. China is even, that is talking about being a big IPv6 deployer, is at 0.24. Because there are so many machines, their numbers are high. But as a percentage it's less than a quarter of 1 percent. So it's just not happening.

Now, I know that our listeners are very privacy and security aware. So get a load of this, Leo. The 128-bit address is divided into two 64-bit pieces, a network number and a host ID. Now, we've talked about how IPv4 addressing works. Once upon a time you had this notion of Class A, Class B, and Class C networks, where the idea was you have a 32-bit IP address which is four bytes. So in a Class A network the first byte is the network number, and the other three bytes is the host within that network. So somebody who had, like, a four-dot IP would have four bytes, and then all the other bytes below that in a Class A network would be part of that network. So you could have, as I mentioned, 16.777 million hosts under that Class A network.

Or you could divide the bytes differently. You could have the first two bytes would specify the network, and the second two bytes the host. That's a so-called Class B network. And there you can have, since you've got two bytes or 16 bits, we know that that's 64K or 65536 hosts, and the same number of Class B networks potentially if all of those were available.

Or you could have a Class C network, which is, frankly, well, for example, everyone who's running behind a NAT router, a typical consumer NAT router, you know, the 192.168.0.something, that's a Class C network because it's the last byte. So where you have 256 possibilities, you lose the first and the last IP within the network, so it's 254 actual machines can be within that network. So that's a Class C network.

Well, the reason we had to sort of fudge those so that there's now what's called CIDR, C-I-D-R, Classless Inter-Domain Routing, what that allows is for the boundary, instead of just being on byte boundaries, to be on any bit boundaries. So, for example, my connection at Level 3, I have a small block of 16 IPs and another one of 8 somewhere else. But so I don't have a block of 256. They slid that boundary down, so I've got the last four bits of my IP block that specifies different machines within my little 16 IP space. Bizarrely, here at home I've got 64 IPs, thanks to my history with the guys from Verio, who then moved over to Cogent. So but then again, it's not 256, it's 64, so that barrier is able - you can slide it up and down in order to economize on the allocation of IPs.

So, for example, if we only had the division, for example, of a Class C, then the minimum network, the minimum block of IPs you could give someone would be 256. Instead, by sliding that boundary down lower, I can have my 16 IPs on my little network. Somebody else can have 16 right next to mine. And we're not on the same network. We're on separate networks. So that's the advantage of the so-called Classless Inter-Domain Routing, CIDR.

Well, when we went to IPv6, all of that problem went away because now we've got - or when we go to IPv6, or when we went to the specification of IPv6, I should say, because it hasn't really impacted the world yet. Now we've got 128 bits to work with. So they say, oh, okay, let's dispense with all of this, the whole Class ABC nonsense. That's gone. We're just going to chop it in half. The high order, the highest 64 bits is the network. The lowest 64 bits is the host on the network. Now, understand that 64 bits for the host means you've got - we used to have 32 bits for all the IP addresses on the Internet. Well, we do now under IPv4. So this is that number squared, four billion squared hosts.

Leo: That's enough.

Steve: That's plenty.

Leo: Somebody, I think it was Vint Cerf, said that would be an IP address for every, what is it, molecule in the universe or something like that.

Steve: Oh, it's ridiculous. It's like...

Leo: It's plenty.

Steve: I read somewhere it's 2^54 IPs for every star in the sky.

Leo: Well, there you go.

Steve: So, I mean...

Leo: That's enough.

Steve: Again, these numbers get big when you start adding bits to them. I mean, they get big fast. But there's a little problem because the guys who designed this protocol said, well, how are we going to assign IPs to machines? And they said, well, we'd like to make them unique. So why don't we base it on the MAC address? So...

Leo: That seems sensible.

Steve: Well, it's sensible except that the MAC address doesn't change. Remember the MAC address is a 48-bit quantity. The high 24 bits is the manufacturer of the network adapter's ID. The low 24 bits is a serial number for that manufacturer, that is, within that manufacturer. So you concatenate those, and you get a 48-bit quantity which is guaranteed to be unique in the world. So they said, well, that'll just be perfect. We'll just use that as the host identifier. The problem is that doesn't change. And that's your IP which you reach out over the Internet with whenever you use IPv6.

Leo: Okay.

Steve: So it is a super cookie.

Leo: Right. It's really identifying you.

Steve: There's a huge, yes, it's identifying that machine indivisibly. So everyone said, okay, that's a problem. So there's an RFC called 3041. And I've got it in front of me. I'm going to read a little bit of it just so you can understand what the problem is. So reading from RFC 3041, it says:

"Nodes use IPv6 stateless address autoconfiguration to generate addresses without the necessity of a Dynamic Host Configuration Protocol" - DHCP that we've talked about often - "server. Addresses are formed by combining network prefixes" - that's the first 64 bits - "with an interface identifier. On interfaces that contain embedded IEEE identifiers" - that's the MAC address - "the interface identifier is typically derived from it.

"The division of IPv6 addresses into distinct topology and interface identifier portions raises an issue new to IPv6 in that a fixed portion of an IPv6 address (i.e., the interface identifier) can contain an identifier that remains constant even when the topology portion of an address changes (e.g., as the result of connecting to a different part of the Internet). In IPv4, when an address changes, the entire address (including the local part of the address) usually changes. It is this new issue that this document addresses.

"If addresses are generated from an interface identifier, a home user's address could contain an interface identifier that remains the same from one dialup session to the next, even if the rest of the address changes.

"A troubling case concerns mobile devices ... that move topologically within the Internet. Whenever they move ... they form new addresses for their current topological point of attachment. This is typified today by the 'road warrior' who has Internet connectivity both at home and at the office. While the node's address changes as it moves, however, the interface identifier contained within the address remains the same.... In such cases, the interface identifier can be used to track the movement and usage of a particular machine. For example, a server that logs usage information together with a source address is also recording the interface identifier since it is embedded within the address." It's the lower 64 bits of the address.

"Consequently, any data mining technique that correlates activity based on addresses could easily be extended to do the same using the interface identifier." So it's like everyone on the Internet had a permanent IP address that never changed as you moved around, and it was used to identify you. So anyway, continuing, it says:

"This is of particular concern with the expected proliferation of next-generation network-connected devices (e.g., PDAs, cell phones, etc.) in which large numbers of devices are in practice associated with individual users (i.e., not shared). Thus, the interface identifier embedded within an [IPv6] address could be used to track activities of an individual, even as they move topologically within the Internet.

"In summary, IPv6 addresses on a given interface generated via Stateless Autoconfiguration" - which is the default case, that is, not using DHCP - "contain the same interface identifier, regardless of where within the Internet the device connects. This facilitates the tracking of individual devices (and thus potentially users). The purpose of this document is to define mechanisms that eliminate this issue in those situations where it is a concern."

So what RFC 3041 does is introduces a way of randomizing this lower 64-bit identifier that would otherwise be static. The good news is, it exists in Windows and is enabled by default. But a command can turn it off. And if it's turned off, you don't know it. You just start generating a fixed cookie, a fixed IP address. You know, "net sh" is a command, "space int" for interface, "space IPv6," and then it's "set privacy disabled." You give that command, or something in your machine gives that command, and your IPv6 system starts now using an IP that never changes, rather than changes normally.

And, although I haven't looked at it closely, because again this isn't a big concern for me yet, apparently other operating systems which are IPv6-enabled do not have this RFC 3041 support enabled by default. Windows, thankfully, does. But other OSes don't. So there are concerns associated with IPv6 that users are going to have to address. And notice that it says this is a problem when we don't have, for example, DHCP. Well, when you do have DHCP assigning addresses, for example, in a NAT network, none of this is a problem. And IPv6 is unnecessary.

So anyway, I just sort of wanted to address the issue. It comes up from time to time, people saying, oh, you know, as if it's this great solution to, I think, a problem that we don't have; and, due to the way it's been implemented, it introduces some interesting new problems for security and privacy.

Leo: Yeah. I had no idea. You know, I've been - Vint Cerf came on The Screensavers many years ago, really pushing v6. He's been one of the people who's been pushing v6 very hard. And at the time, of course, we were very worried that we were running out of addresses. And as you pointed out, this has all changed thanks to DHCP and widespread use of routers.

Steve: Yeah, exactly. The ability to use NAT, which is 100 percent IPv4 compatible, it's completely changed the game.

Leo: Yeah, right.

Steve: So that this just isn't a problem. And even now we're still using - we still have huge chunks of IP space unused. I mean, all we have to do, and I'm sure there's mechanisms for doing it, is to go to people who are squatting on unused Class A networks and say, okay, give 'em up.

Leo: Right, right.

Steve: We're de-assigning these to you for the good for the world. Because you're not using them; and, guess what, you don't need them. So the world has changed since back then.

Leo: All right, Steve. Non-VPN, what the heck are you talking about, dude?

Steve: Well, okay. You know I've been spending a lot of time at Starbucks, where our listeners come by and say hi and introduce themselves, which is always fun. And I've been using OpenVPN to post updates to the DNS benchmark at GRC, and sometimes to get to my network here at home if I need a file. And one of the problems is that, if I've got an established connection - for example, I've got my news reader that's connected to the news server at GRC, and it's connected through the Starbucks hotspot to the Starbucks router and out to the 'Net. Well, if I bring up the VPN tunnel, the news reader starts to complain because I've disconnected it. I've basically put an enclosure around myself saying that I'm now suddenly participating in this remote network.

Well, I've understood that that's a problem. And so I've always had in my own planning for CryptoLink this notion of what I call, for lack of a better term at the moment, my working term is "full enclosure" or "partial enclosure." The idea being that the way OpenVPN, for example, and as far as I know all other VPNs work, is that the machine is completely enclosed and is participating as a peer on the remote network. Which is useful, but sometimes it's not what you want. And so for CryptoLink I've sort of identified a feature that I believe will make it unique, which is a partial enclosure where you can designate specific remote locations whose traffic will be routed through the VPN tunnel, and others that won't.

Now, in theory you can do that with OpenVPN because you're able to modify the routing table to determine what's routed through the VPN interface and not. But that gets into some pretty hairy rocket science. And of course I'm going to make this a lot easier. But the other problem is there's still the weight of this sense of, like, bringing up a tunnel, like doing something to establish a connection that all VPNs have. And what's occurred to me, and I just sort of wanted to share conceptually with our podcast listeners, is there's an alternative approach which I'm really intrigued by and which I may implement, I'm not sure, which is a non-VPN solution which is every bit as secure as a Virtual Private Network solution. And it solves a bunch of problems in a way that I'm intrigued by because I'm coding everything in Assembly language, implementing protocols myself. I really - what drives me is lightweight solutions, minimal solutions, something that works, doesn't get in the way, and there's no baggage. There's nothing extra.

So what I'm wanting to do when I'm at Starbucks, for example, and this applies to any or many telecommuters, certainly there's an application where you need to be on your corporate infrastructure, that is, you need to use a VPN to participate as a peer on your - like to be as if you were at your desk in your corporation. For example, a corporate mail server that maybe you only can get to if you are on the inside, and the VPN allows you to simulate being on the inside. But for many users, like Windows users, for example, something that works very nicely is file and printer sharing. I know that all kinds of people, for example, in their networks at home, they're mapping drives, as is the term, from one machine to another, so that you just - you map a drive on a different machine to a drive letter on your machine, and it's like you're there. You're able to open it, to explore it, to browse around in it and so forth. Well, all you're really doing is you're making a connection between your machine and the other machine over the Windows file sharing port, which is port 443. Is it 443 or 445? 445. 443 is SSL.

Leo: 443 is SSL, yeah.

Steve: Yeah. So all you're really doing is making a TCP connection. Now, the question, of course, is why is that dangerous to do from Starbucks? And the answer we all know, I mean, the answer is legend, is for me to do that I would have to have port 445 open and facing the Internet on the machine I want to connect to. So say, for example, I wanted to access my C drive at home when I'm at Starbucks. Well, I'd have to have file and printer sharing exposed and, in my case, mapped through a router to that machine, which would be incredibly unsafe. No force on earth could make me do that.

But if there were a way for that to be safe, then I have the simplest of all possible modes of access. I'm not having to mess with a VPN. I'm not having to raise a barrier which cuts off other things I'm doing. And also, for example, routes all my traffic through there and then back out onto the Internet, which can result in lower performance. I mean, I'm at Starbucks, I've got a direct connection to the Internet now. Why force all my traffic up through my connection to home and then back out the same connection, which introduces delay and bandwidth constraint? But of course it's not safe.

Okay. Well, imagine that there were a way to make it safe. Oh, the other thing, the other problem is many ISPs are now blocking these unsafe ports, like they'll block 137, 138, and 139, which were the old-style NetBIOS over TCP/IP file and printer sharing ports. And they're also blocking port 445 because of all the problems that Windows has had historically. Well, it's good that it's being blocked because bad things can't get into it. But it's bad if it's me because I want to get to it. So we want to avoid ISP filtering. We also want to avoid there being any security vulnerability.

Well, so I had this idea, and I call it sort of like a junior VPN. It's not a VPN. I don't know what to call it. But imagine that there were device drivers, network drivers, at either end. And that at my home end, this network driver was listening to other ports, not 445, but other ports, and strongly encrypting and authenticating any packets which arrived there with a symmetric key, a secret symmetric key. And at Starbucks I have a network driver which is sort of doing the complementary thing. Windows thinks it's sending packets out toward 445. But when they come to this driver, it says, oh, this is going to home, so we port shift. We shift the destination port up to one or multiple ports, because there's nothing to say one packet can't come in and seven can't come out, aimed at different ports, in order to make sure that we get through to the other end. And that same strong symmetric encryption and authentication is applied so that what the packets contain is absolute gibberish. I mean, as strong as any VPN. And that's all that a VPN really is, after all, is just symmetric encryption and authentication using a key that no hacker can know.

And so this packet leaves, gets shifted to a different port, goes out, passes the ISP's filter, comes in, gets decrypted and authenticated, meaning that only my machine at Starbucks is able to generate a packet which decrypts correctly and authenticates. The port is shifted back down to 445 so Windows is happy with it. And I'm able to map my C drive or whatever drive or resource that I want to from home remotely with no overhead, no tunnels, no routing, no encapsulation, I mean, there's other complexities which introduce the performance problems that people using VPNs often experience. This dispenses with all of that. Far as I know, this has never been done. And it's the first experiment I want to try as I begin coding CryptoLink for my own use.

And it occurs to me that it would be useful for lots of instances. When I was reading - the thing that sort of spurred me to mention this is I was reading the mailbag last week for last week's Q&A. And we had a listener, Steve Hiner, who was having fun. He said, "At the primate exhibit, writing VB with all the monkeys." And so he said, "I wanted to make a suggestion for a CryptoLink feature. As a software developer, I sometimes need to be able to set up communication between one of my programs and my server. Back when I found out about Hamachi through your show I found myself wishing I could use a Hamachi DLL" - like a lower level Hamachi component - "to temporarily create a secure link to my server. That way my programs could communicate with my server without having to deal with the security hassle of running a web server. With a goal of trying to reduce the surface area for an attack, it would be great not to have to expose a web server, but use a CryptoLink connection instead. What do you think?"

Well, I mean, he's talking about sort of a variant of exactly what I'm saying because the other thing about this is, I mean, it's perfect for program-to-program communication. In addition to, for example, me manually mapping a drive, there would be nothing to prevent you from connecting to a file using the Microsoft-style UNC, the standard format for generating those sorts of links.

So anyway, it's - one of the challenges I face is creating the feature set that I'm looking for in CryptoLink and not having it be confusing to people, but at the same time wanting to offer features like this that have never been done before, but which, I mean, create a compelling capability. I just - I love the idea that, again, wherever I am, I would have access to the drives I have at home in a way that's supported by Windows and that in no way mucks around with or messes with or creates any other problems for my normal use of the machine.

Leo: So this is going to be in CryptoLink, you think.

Steve: I don't know.

Leo: Can you try it as an experiment, and then see how it works, and then maybe put it...

Steve: I'm absolutely going to. I mean, I know it'll work. It's a simple means for allowing, well, there are two things. First of all, it only really works this easily over TCP, so it's for TCP things. But that's what Windows file and printer sharing are. That's what desktop, remote desktop is. So it would require TCP things. But again, there's all kinds of other problems associated with tunneling that this completely sidesteps. It's just - it's a simple solution. And it's just - to me it just seems very compelling and absolutely as safe as a VPN because nothing but - no one but I can generate these packets. And there isn't even a port or ports open that can be scanned because one of the things that the technology I've got does is it allows you to authenticate - it allows CryptoLink to generate self-authenticating TCP SYN packets. So you're completely stealth. Nothing can see that you've got multiple ports open. And only you are able to connect to it. And it's just as easy as mapping a drive or printer on a remote location. Anyway, I'm intrigued, and we'll see where it goes.

Leo: This stuff is hard, though, because you just, well, you know. But I find this stuff difficult because it's hard to think of all of the ramifications, isn't it, and all of the possible holes and so forth. I guess this is pretty straightforward.

Steve: It's what I find so compelling is it really is very straightforward. And we've got a really good team of developers and testers hanging out in the newsgroups who are smart guys like John Graham-Cumming who wrote "The Geek Atlas" and others who will check me on this stuff, too. I mean, they've done it in the past, and I'm sure they will here. Anyway, that's just - I wanted to share it because it's been on my mind. It was brought up by this Steve Hiner in last week's mailbag. And I thought, you know, I'll just give it to our listeners to think about because I think it's really compelling.

Leo: Good. This is how you test something like this. You ask a lot of people. You ask security experts. You try it out. You see what happens. Very good. Steve, some great, fascinating stuff. I thank you for a potpourri today.

Steve: You know what next week is, Leo.

Leo: What's next week? Oh, our 200th episode.

Steve: Number 200.

Leo: That's kind of amazing. We should celebrate.

Steve: Not a week missed.

Leo: I'll drink champagne on this side; you drink champagne on that side. Or you can have a nice burgundy, whatever you prefer.

Steve: Cabernet.

Leo: Cabernet, that's right. By the way, his place is GRC.com. That's the place to go on the web to find Steve's show notes, the 16KB versions of this show, the transcripts. We have transcripts for each and every episode. But also Steve's other great software: SpinRite, which is absolutely the one and only, the best disk recovery and maintenance utility, the one you've got to have; all his free stuff, like ShieldsUP!, Wizmo, DCOMbobulator, all that great stuff. It's all at GRC, Gibson Research Corp., GRC.com. And we will be back here in your iPod or iPhone or Zune or Kindle or whatever you listen to us on, maybe on the web, next Thursday and every Thursday, pretty much for the rest of our lives.

Steve: I think so. At the rate we're going, that's definitely the case.

Leo: Thanks, Steve. Have a great week. We'll see you next time on Security Now!.

Steve: Thanks, Leo.

Leo: Bye-bye.