SECURITY NOW!

Transcript of Episode #194

Listener Feedback #65

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-194.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-194-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 194 for April 30, 2009: Listener Feedback #65. 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!, Episode 194, the show where we talk about all your security concerns. We haven't yet done locks...

Steve Gibson: Actually, we'll give you security concerns, if you don't have any already.

Leo: That's right. Yes, exactly. That's it. And if you don't have them, we'll provide them for you.

Steve: That's right.

Leo: That's Steve Gibson of GRC.com, the creator of SpinRite, the world's best hard drive maintenance and recovery utility, also a security guru, creator of ShieldsUP!, the discoverer of spyware, and our host. Hello, Steve.

Steve: Yo, Leo. Great to be with you this week. Again, we're approaching the end of our fourth year.

Leo: Whoa.

Steve: I'm excited.

Leo: Whoa.

Steve: That's going to be - that's going to be very cool. We're...

Leo: That's hard to believe.

Steve: Yeah, oh, I know. It's like, four years, wow.

Leo: Unbelievable. Amazing. Well, it's been fun, and we never run out of topics, unfortunately. In fact, you couldn't pick a...

Steve: Wait, unfortunately?

Leo: Well, I guess fortunately.

Steve: Oh, yes, for security reasons, yes.

Leo: And unfortunately. But you couldn't really pick a better subject than security right now.

Steve: A fertile medium, yes.

Leo: Holy cow.

Steve: We have sort of a fertile topic, certainly. And our listeners, who've been anxious for the how SSL protocol works in detail episode, will - given that nothing really phenomenal and important happens during this next intervening week, that's our topic for next week. Today we've got a Q&A.

Leo: Excellent. Your questions, Steve's answers. Before we do that, though, any news to report? An updates on anything?

Steve: Yeah. It's been relatively quiet. I did want to mention, I loved - I was watching, was it you and Dick or, no, it was one of the many feeds you have where they mentioned that there's now a - well, actually I didn't ever look for it. But there are several star date calculators. So I just thought it was...

Leo: Isn't that funny that star - oh, yeah, we were talking on the TWiT about "Star Trek." And I said, well, what's the - I can't remember how it came up, but how would you know what the star date is?

Steve: Yeah, and we are, today, this podcast's date is -314327.28. Just so...

Leo: You looked it up.

Steve: Of course. Absolutely. And it turns out you just put "Star Trek calculator" in.

Leo: It's pretty easy, yeah.

Steve: In fact, there's a little guesser thing that's guessing what you're looking for. It even had it on there. So just wanted our listeners to know that we are -314327.28.

Leo: That's great. That's great.

Steve: There were - I only have three notes in security news. The Firefox updates are coming fast and furious lately. We just got updated to 3.0.9, which fixed a whole bunch of problems. And then just two days ago, three days ago, we went to 3.0.10, which just fixed one more problem. So anybody who's got Firefox is probably already up to speed on that. If you haven't seen it, there is an update from .9 to .10. And that's critical, by the way. It is a buffer overflow problem in some text rendering. So you want to do that.

I did also note that Microsoft's out-of-phase update - remember they do the important security updates on the second Tuesday, and they do, like, two Tuesdays later, they do things that are sort of, well, you know, we just thought we'd throw these in. These aren't super critical. Don't worry about them. But I noted that IE8 has now been added to Microsoft Update. So I was - actually I was at Starbucks, and I got this notification that they wanted to send me IE8. It's like, eh, well, I guess they must be confident enough of 8. I mean, my feeling was it wasn't ready yet for primetime. I don't use it anymore except to go to Microsoft for updates. I'm completely converted to Firefox. So I'm not vulnerable to many security issues because - which I would imagine by default IE8 will have a whole new slew that'll now have to be found and hopefully won't be biting too many people until they are. So anyway, IE8 is out.

And something we don't see very often, there was a critical Blackberry PDF vulnerability which was announced. So Blackberry users, I just - we're all used to getting updated moment to moment by our Windows apps. But I did want to make sure that people who were using Blackberries knew that opening a PDF, until they update their Blackberry after this problem, there are exploits that will allow your Blackberry to get taken over, which is of course not what you ever want.

Leo: So it would be you would download a PDF? Or you'd...

Steve: It would be viewing a PDF on your Blackberry is, I mean, the standard, I mean, PDFs have become unbelievably complex. I mean, they've got JavaScript support. And the state-of-the-art free Adobe Reader is now hundreds of megabytes in size. It's like, I don't know what it's doing; but whatever it's doing, it's a lot of it. So hard to imagine that there aren't problems when you've got that much code.

Leo: No kidding.

Steve: Which is, you know, taking in something and interpreting it. So there is that in the Blackberry. Just wanted people to know. And in the spirit of the Q&A, as I was going through my mailbag to collect our today's 12 questions, I ran across a SpinRite question. So I thought, well, I'll do that as my little SpinRite mention of the week. This was from Jose Cerna in San Diego, who wrote to ask, he says, "I use SpinRite regularly to keep all of my drives in top condition. But recently I bought a laptop with an SSD, a solid-state drive. So I was wondering if SpinRite would be okay to run on it, and does it need it. Thank you for the great podcast.

P.S.: I sometimes fix computers for others, and I have used your product to fix their drive and/or recover files. I know that this goes against your EULA, the End User License Agreement. But what I have been doing is I have them purchase SpinRite in front of me, then I download it for them and install it into a Flash drive that they can then use to maintain their drives from then on. I have done this with three customers already, and they are very happy with SpinRite." And of course I'm very happy with Jose for...

Leo: Thank you, Jose. A few yabba-dabba-dos right there.

Steve: That's some yabba-dabbas. Anyway, to answer his question about solid-state drives, I can see absolutely no possible benefit to running SpinRite on a solid-state drive. We know that solid-state drives, good ones, are very reliable. I just bought a 64GB SSD for a new tablet that I own because I'm just - I'm so conscious of the mechanics of a hard drive and what kind of problems people have. Naturally, it's been my bread and butter for more than 20 years. And this thing, I checked it out, has an MTBF, a Mean Time Before Failure, estimated - this is a Samsung 64GB SSD - estimated at two million hours.

Leo: Whoa.

Steve: So theirs is good technology. It is possible to buy crappy solid-state drives. People are invariably going to do that. There are multilevel storage as opposed to single-level cell storage. You can get 64GB, for example, in a thumb that's for a couple, maybe a hundred bucks, probably. But you get what you pay for in SSD technology.

Leo: You know, I just bought a - we've covered this a little bit on our PC Perspective show because Ryan's very interested in this. And he has a great guy, Allyn Malventano, who has become an SSD expert. He's testing all the drives and stuff. And he also likes the Samsung drives. They recommended a Corsair model, which is Samsung parts, I believe. MLC, though, not SLC.

Steve: Ooh, okay.

Leo: Yeah, no, they said that's fine, that in fact very few drives are using SLC. Those are really only the enterprise drives at this point. They're extraordinarily expensive. This was still expensive. For 128GB it was 330 bucks.

Steve: Okay, see, I paid $800 for 64GB.

Leo: You got an X25.

Steve: Yeah.

Leo: The Intel. I don't know if the Intel's SLC.

Steve: No, no, no, I got a Samsung. It's...

Leo: Oh, you got Samsung, okay.

Steve: It's a Samsung, but it is an SLC...

Leo: It is SLC, okay.

Steve: Right. I mean, it's, again, this was - I've done something weird, Leo. I've been spending many hours at Starbucks coding by bringing a copy of my work, my regular home keyboard, because it's only on this full-size Northgate 102-key keyboard that I can type at full speed without even thinking.

Leo: See, I can't use those. They're so loud I can't use them in the studio. They clatter.

Steve: Oh, exactly.

Leo: It sounds like a teletype machine in here.

Steve: I finally figured out that that was the problem was, you know, when I'm hunched over my laptop, I just - it's like, okay, where's the delete key on this? I mean, it's a compressed keyboard for a laptop form factor.

Leo: Right, right.

Steve: That was my problem. So I thought, okay, I need a tablet that has no keyboard, with a nice size screen, and then I'll get an old-style XP, I mean, XT to PS/2 converter, then a PS/2 to USB. So I have a chain of sort of dongles that convert this thing into something that runs now USB. And it's just fantastic. I have the tablet propped up on a little bookstand in front of me, the keyboard there. And you'd be surprised how many looks I get from people who walk by, go what the heck is that keyboard doing here? Anyway, so, but I'm getting - I'm hugely productive now. I've done five ten-hour marathons on the last five days in a row.

Leo: Yeah. And that prevents carpal tunnel, too, I think, because there's more motion, more travel...

Steve: It's fantastic, yeah.

Leo: It's more physical, yeah.

Steve: So anyway, yes. No need to run SpinRite on an SSD. SpinRite is all about mechanics and magnetics, neither of which exist in, by design, in an SSD. That's what you want to get rid of in order to get reliability up. Thus SSDs are extremely reliable.

Leo: I have to tell you the speed on the Corsair is amazing.

Steve: Yeah.

Leo: Load times particularly. But, you know, I did some transcoding, too, because there's a lot of reading and writing in transcoding video. Like twice as fast. I mean, remarkable differences.

Steve: Yeah, yeah.

Leo: So mostly access time benefit, I guess.

Steve: Yup. And I just, for me it's that, I mean, I'm just - I'm so twitchy about any portable machine with a hard drive because they're inherently - they're moving around. They're going to get some bumps and things just inadvertently. And it's like, okay, this is now my main development platform when I'm away from the house. And I care about its reliability. I mean, everybody cares about its reliability, so.

Leo: Yeah, yeah, yeah. Well, I'm excited about it. They're still pricey. But I just thought, I'm going to give it a shot. Both Ryan and Allyn pushed me to do it. And I put it in a MacBook Pro. And it's just - I love it. I can't wait till the prices drop. I'll put them in everything.

Steve: Yeah, I'm not that excited for them to drop.

Leo: Why? Oh, it puts you out of business.

Steve: No one needs SpinRite.

Leo: Whoops. Well, you know what, I think here's what's going to happen. Certainly the usage I plan is you'll use it for your boot drive, your applications, the stuff where you do a lot of I/O. And then you'll have massive spinning drives, these terabyte or two-terabyte drives, for data storage.

Steve: It is the case that hard drives will not disappear overnight. I mean, they're just not going to. That technology is so mature, and they are so cost effective that - and, you know, SpinRite is recovering media for people now. You don't use SpinRite to get Windows back because you can always reinstall that. You use it for your priceless photos, your Ph.D. thesis, your huge music collection of borrowed CDs that you don't have.

Leo: Or movies, frankly. I mean...

Steve: Yeah, exactly. So, yeah.

Leo: So that's - and I think that makes perfect sense. Why use Flash for something like that, where speed is not important, it's storage. I bet you we will see these different classes of storage for some time to come. So you're all right, Steve.

Steve: It makes sense that there would be a spread. Well, and I've got my next product on the way. CryptoLink is nascent, but going to be next.

Leo: Well, you'll always have a home on TWiT, too. I mean, you're the security guru. That's the good news. There's always going to be a need for that, I don't care what kind of drives we use. What else you got?

Steve: That's it.

Leo: That's all?

Steve: Yup.

Leo: I have the questions, Mr. Steve. Are you ready for me to read?

Steve: Yes, indeed. And we have a special first question.

Leo: From Elaine.

Steve: Yeah.

Leo: Oh, that's so cool. Elaine, our illustrious audio transcriber, asks this week's first question, and it's appropriately about Conficker.

Steve: Yeah. There were a bunch of questions that I ran across in the mailbag. There were some things I had - I sort of glossed over, or I didn't cover. So we've got a - we'll have some clarification of the issues from last week.

Leo: But do start, if you haven't listened to last week's show, Episode 193, the whole show is on Conficker, in great detail, how it works, what the dangers are, how it's spreading. She says: This Conficker thing's been giving me a headache for weeks, ever since you first mentioned it. I don't need help with spelling any names in the podcast - often a cause of concern. She says: Instead, I need help with my own head. Please, what I don't understand is, when you keep saying it's, quote, "generating domain names," end quote, and you say it's gone from generating 250 or 500 to 50,000 domain names, what does that mean? Don't those generated domain names need to be registered in order to be used by the worm? I must be misunderstanding something at a really basic level because it looks to me like it's costing someone half a million bucks a day just to run the worm. I usually try to keep my ignorance to myself, but this is really bothering me from week to week.

Hey, Elaine. That's great. She's paying attention. And I should have asked that question. That's a great question. What's the deal, Steve?

Steve: It is a great question. And I don't think I was clear enough in explaining why this is so clever. The idea is that the worm generates a huge number of "potential" domain names, and at random chooses from among those. So, and it's based on the current UTC calendar date. So everybody knows what the algorithm is. The bad guys that designed all this, they know what the algorithm is. The white hats that have reverse-engineered all this know what the algorithm is.

And what's clever about this is that this succeeds even with full reverse engineering. That is, we know we cannot protect the secrets of anything that needs to run remotely, just like the DVD or Blu-Ray and HD guys were unable to protect the secrets even in their most fancy encryption/copy protection technology. You just can't. Because if a device has to run in order to display it, then you can reverse engineer what it's doing, no matter what the people who try to prevent that, whether they're good guys or bad guys, no matter how much they try to be clever to keep that from happening.

So we know that everybody understands the algorithm. So this succeeds even in the presence of that complete documentation of the algorithm. Every day a set of possible domain names that has increased in size after April 1st from a few hundred to 50,000, every day a set of those are possible. From among those, the worm chooses only 500 to actually attempt to go to. So the way this works in a large population of worms is the good guys on our side, the white hats, if they wanted to block all of the possible domains from which on any given day all the Conficker worms could get updated, they would have to preemptively register those, all of those 50,000 domains. Whereas the clever black hat, the clever author of Conficker, only needs to register one or several on any given day.

Leo: He knows ahead of time what the domain's going to be.

Steve: Everybody does. The good guys and the bad guys. But my point is, to preempt the update from getting into the new Conficker peer-to-peer network, to preempt it, all of the domains have to be blocked. To get it in, only one has to be registered because some of those worms will contact, choosing randomly, that one domain. And that will be enough to get them updated. And then they use the peer-to-peer network to update all the other worms that contacted domains that were not registered or were registered by the good guys to block it.

So that's what's so clever is that, I mean, every single day 50,000 new domains are going to be probed. And this is now in 110 top-level, like not just .com and .edu and .org and .net, but now all these obscure, 110 top-level domains. Which makes preemptively registering them basically impossible. And all the bad guy has to do is a few days ahead, just camp out, put a server on one of the domains that he knows in a couple days some small percentage of Conficker will visit. And so he needs to register one. The bad guys need to register all of them.

Leo: The good guys.

Steve: I mean, sorry, the good guys need to register all of them if they're going to block it. And all of them every single day. It's just not possible. So it really creates this dramatic asymmetry of effort, which is why I think this was - and this was extremely clever. And you can imagine that this will now be the paradigm for bad guys keeping in touch with their botnets is this kind of algorithm because it is simple to implement. It is very clever. And it is - there's really nothing you can do to prevent that asymmetry from existing.

Leo: Very interesting. Thank you for asking that, Elaine. And I apologize for not asking it myself. Doug Curry in Houston, Texas asks: What have I missed? Steve and Leo, I was just listening to 192 where you mentioned that the Conficker worm uses public key encryption digital signatures to protect itself from being spoofed with updates from someone other than the original authors. Again, a nice, slick thing to do. You said that even if you got the public key there was no way to reverse engineer the private key. Of course that's the point behind public key encryption. Fair enough.

But help me out here because I've got to be missing something. I'm clearly no cryptographer. But if you know the public key and encrypting algorithm, couldn't you put some set of known data through the algorithm, using the public key, obtaining the encrypted output, attempt to decrypt that output with the same algorithm and a sequence of successive potential private keys - there's the issue, we'll get to that - comparing the encrypted output to the original, in other words a brute force attack against this public key? And given that a brute force attack would require massive amounts of computing power, could you not use a distributed computing methodology like SETI@home in order to get a large number of computers all working on small sections of the problem simultaneously to come up with a solution in a reasonable amount of time? Well, you could do that, couldn't you, Steve.

Steve: No.

Leo: No.

Steve: No.

Leo: Oh, oh.

Steve: Not even close. Now, we've never discussed brute-force attacks against public key encryption, which is the reason this question caught my attention. We've discussed extensively brute-force attacks against symmetric encryption.

Leo: Right, right.

Steve: Now, okay. There's two reasons. First of all, remember that symmetric keys are - they used to be 64 bits. Now you probably need 92. But people are at 128 or 256. Okay, there are enough combinations in even a 128-bit key that brute forcing a symmetric encryption is completely infeasible. And what do we know about symmetric versus asymmetric encryption? We know symmetric is fantastically fast. And asymmetric, that is, public key encryption, is agonizingly slow. It is so slow because of the math involved - all kinds of exponentiation and higher level curve elliptical functions and things. It is so slow that nobody uses it to encrypt anything but a symmetric key.

So as we've discussed before, the way you use public key encryption to encrypt a block is you choose a random number with a very good random number generator as your symmetric key. You use that to encrypt the block of payload because it's so fast. Then you only use the public key technology to encrypt just that one little 128-bit, 256, whatever it is, symmetric key because public key encryption is so slow. Now, not only is public key encryption slow, meaning that, okay, brute-forcing attacks are just infeasible for that reason, but due to the different nature of asymmetric encryption versus symmetric encryption, the key lengths are much longer, like 1,024 bits, or 2,048 bits. And you need that kind of a key length to get the equivalent level of protection that you get with a much shorter, like 128-bit symmetric key.

So on one hand, the algorithms are very slow, and the key lengths are eight or 16 times longer. So that's why we've never discussed brute-forcing public key encryption. Although the fact that we hadn't explicitly ever discussed it, I thought, well, this is a really good point. Doug's question is something that we haven't covered before. So ne now we have.

Leo: So the deal is the raw number of possibilities makes a brute-force impossible?

Steve: Two, two things. The raw number of possibilities, that is, already 128 bits is so many possibilities with a fast algorithm that there just isn't time in, like, billions of years.

Leo: But now you're using 2,096 bits.

Steve: Yeah.

Leo: And it's slow.

Steve: And, exactly, and every single one you try is incredibly costly in terms of time. It's just nobody ever attempts to brute force public key encryption. They do try to find mistakes in the algorithms or other short...

Leo: Yeah. That's your back door. That's your hole. Yes, yeah.

Steve: Yes, yes. Other shortcuts of some kind. But just not brute force because it's very clear to anyone who begins to try, oh, gee, let's see, we've tried six keys in the last 30 seconds. How many more millennia? I mean, the universe will expand and contract and expand and contract many times before you have a chance to get that to happen.

Leo: You just can't do it. That's just the bottom line. You just can't do it.

Steve: Yeah. Consequently, the Conficker guys do not need to worry about anybody replacing their payload.

Leo: No, no. Russell Gordon in Houston, Texas shares some experience and wisdom with Windows, using Windows in process control environments. I'll find out what that is in a second. I just paused the Security Now! 192 I was listening to so I could write and thank Steve for his response to Phil in Montreal. I have worked in the process automation industry for 19 years, programming PLCs - Programmable Logic Controllers - and designing the operator screens called HMIs - the Human Machine Interface - that allow the operator to interface with the control system. While the PLCs are not running on Windows, the software to program them is. And the HMIs are usually industrialized PCs running Windows. Of course.

I recently had to chase the Conficker worm out of a brewery because it was running rampant on their HMIs, which are unfortunately all connected to the corporate network. We also have Windows PCs used in the control systems running pipelines, and chemical and pharmaceutical plants. Luckily these are not connected to the Internet like the brewery. I guess a brewery is not considered essential technology in the same way that pipelines and pharmaceuticals are. Luckily these are not, as I said, connected. But all it takes is a rogue laptop to bring something into the network. Of course. Thank you, Steve, for acknowledging the current state of Windows as it is. I, too, understand how it got this way by supporting backward compatibility. But I also know how much better it could be. Russell Gordon.

Steve: Yeah. So that was essentially, here's another person's direct experience with Windows and non-Windows systems, or also networked and non-networked Windows systems. Here he's not saying that Windows itself was a problem, but letting them communicate is a problem because, as he said, you bring a rogue laptop into the network, and suddenly your off-the-net system is prone to be infected by something that's been brought into the network. So, yeah, just not a good idea.

Leo: Yeah. Question 4, Paul Rudy in Astoria, Queens shares his Windows in the Machine story. He writes: I was just listening to #64 Listener Feedback, and it prompted me to send you this message. I recently went to my local TD bank so I could withdraw some cash. I looked at one of the three ATMS, noticed it was sitting at a Windows XP Professional logon screen. That is not inspiring confidence, I'm afraid. CTRL+ALT+DEL. It's bad enough that my online password is limited to relatively simple rules. But now I see the ATM that controls my money is running Windows XP. I just pray they keep the OS up to date. I would hate to think that Conficker or some other lovely worm was on this machine, keeping track of everyone's ATM card and PIN number. This cannot be normal.

Steve: No, Leo, that's my point. It is. This is what's so disheartening is that - and I'm not going to say anything about what language that ATM software was written in.

Leo: It's probably Visual Basic.

Steve: But we don't know. But again, sometimes you see these systems unmasked, as this guy did.

Leo: That's depressing.

Steve: One of the three ATMS was waiting for someone to login by CTRL+ALT+DEL. Which means there's Windows underneath. And it is truly, truly wrong that that kind of application has a consumer, I mean, this is for consumers. This is not for industry. Notice that the brewery we discussed before, they do use Windows for their UI.

Leo: Just their interface, yeah.

Steve: Yes. But the actual process control systems are non-Windows. That's the stuff that's turning the valves and running the motors and making sure that beer comes out the other end instead of, you know, lord knows what you would get if those valves and pumps were being set incorrectly. So it is - what I'm seeing is evidence of this creeping use of Windows in inappropriate instances, in places where you absolutely should not use Windows. I mean, and there are places. I mean, certainly consumers are using it. I'm using it. I'm sitting in front of it right now. But not...

Leo: But nothing's going to go wrong if it crashes.

Steve: Right. I mean, because I'm not controlling a nuclear reactor with Windows.

Leo: Oh, I hate to think that that's happening.

Steve: Oh, yeah, well...

Leo: But I bet it is.

Steve: Let's hope not.

Leo: [Sighing] Mark Davis, Sandston, Virginia takes issue with Steve's Microsoft bashing. Okay. I was hoping we'd get a rebuttal. Let's hear what he has to say: I've been a long-time listener to Security Now!. I'm not sure when I began listening, but I enjoy hearing about what is happening with computer security. But I take issue with always making Microsoft the bad software company. I agree they could do a better job, as all software companies and writers of software with security bugs should do. The issue I take is that most issues - okay. I'm just going to rephrase this.

Most issues with infected computers come from the users themselves. That's what I take issue with. I find they're either too lazy with patching the machines, or they turn off security in their software, or they never read the manual or instructions about how to secure their computer equipment. The moment software starts making people do the things they should is when they start complaining the program's not user-friendly, and why do I have to click three times to tell a program to do something? Well, that's a good point.

We all know that software has bugs in it and there are people out there who will spend time looking to find those bugs. The reason certain software gets targeted is because it has most of the market. And when someone else has more of the market we'll hear about how bad their software is when the bad people decide it's worth targeting. And yes, I have SpinRite, which is the only tool for those times when the hard drive says, "Captain, I don't know how much more I can take."

So he has a point. I mean, I hate it when people blame users. Look, I do a radio show for normal people. Users. And the first thing I say is you shouldn't have to be a security expert. And because you do have to be a security expert to use Windows, I encourage them to use something else.

Steve: Okay. I think I've probably made myself clear already.

Leo: Yes.

Steve: But I want to make sure that Mark and anybody else who feels I was unfairly bashing Microsoft understands. Now, okay. it is the case that I called Windows a big steaming pile of crap a couple weeks ago. So there's that. But my point is not that. My point is the misapplication of Windows, as I was talking about before. I mean, Microsoft, all they've got is Windows. If Microsoft created a little real-time operating system kernel that was designed to be lean and mean and run breweries, I would bet they could do it, as long as they didn't use summer interns to do the programming, as unfortunately they do on a lot of the stuff that we're running. The stories of the creation of NT - I read several books about that - are just horrifying, about the critical systems that some random summer intern was given to work on. It's like, okay, fine, looks like you did a good job, good luck with your junior year.

Leo: There is embedded Windows, I think, and there's CE and stuff.

Steve: Yeah, but all that is is stripped down steaming pile.

Leo: Right. That's a good point.

Steve: It's not rewritten from scratch. And in fact what Microsoft has done, unfortunately, is they have made decisions over time which they felt were warranted then, but which negatively impacted the stability of the system. There was once - the original design of NT was, remember, it was a multi-API kernel. You could run POSIX, and you could run 16-bit Windows and 32-bit Windows. It sort of had this plug-in architecture. It was sort of a microkernel approach. And, gee, we need more performance. So let's move GDI into the kernel. Oh, shoot. Now that means that a buffer overflow displaying a JPG is a kernel overflow rather than out in the user space where it can't do any damage. And so you could see the decisions Microsoft has made over time for expedience sake have ended up having dramatic security implications.

Okay, I guess I'm not supporting Mark's rebuttal here. But I understand that software is difficult to create. I don't blame ever, ever, Microsoft for having bugs. I blame their policies. I blame them for the things they deliberately do, like putting scripting in email, like once having scripting in Windows metafiles, apparently on purpose. And those sorts of policy decisions, or moving GDI, the Graphics Device Interface, into the kernel. That's what you do not do. And Microsoft does it over and over and over due to a lack of respect for the consequences of these things. But the other issue separate from this is there are places you should not use Windows. Anywhere...

Leo: Anywhere.

Steve: Yeah, well...

Leo: I'm sorry. I had to say that.

Steve: Anywhere that you absolutely need security and reliability, that's not Windows. We all know that's not Windows. And you could argue, okay, it's not Commodore 64 or Mac OS X or Linux or any of the big popular consumer OSes. And I wouldn't agree - I wouldn't disagree with that at all. There is a whole separate class of industrial-strength operating systems that consumers have never used because they don't have all the fluff on them. They don't do all the things that we want. They may not even have network connectivity like we're used to. But they are what you want running your ATM at your local bank and the nuclear reactor in the next county. That's what you want, not Windows.

Leo: And there, you know, there's secure, I think, BSD, what is it, OpenBSD? Which one is the one that you like that's the most secure? NetBSD?

Steve: I'm using FreeBSD.

Leo: FreeBSD, that's it.

Steve: Yeah.

Leo: I mean, there are operating systems designed from the ground up to be secure, that are full consumer operating systems. I mean, it is doable, if that's your intent.

Steve: And that's a very good point. And Microsoft has not chosen that path. Microsoft, as I said, has chosen, over and over, expediency. And here's the real problem is that, now that hardware has caught up, you really don't need GDI in a kernel so much. But back at one point you did, and that's where it's living now.

Leo: Well, and there's where the legacy, the compatibility is biting them because they can't take it out.

Steve: Right.

Leo: Even if they know better now, they can't. Bill in Walnut Creek joins the operating discussion with a comment about the NeXT cube: Steve, I see we're in agreement with just about how steamy a pile Windows is. I've used many operating systems - Windows, Mac OS 9, Solaris, Red Hat Linux, HP/UX, IRIX, AIX, Be OS, et cetera. If I have any criticism of my past work on technology, it was most of the industry completely missed what NeXT was doing. Think about this. It has been ported to these CPU architectures: the Motorola 68030, Intel/PPC, Intel/ARM. All of these transitions were completely seamless. It is a tribute to the flexibility of the microkernel architecture that NeXT has maintained despite some cost in performance. Just imagine if Microsoft had to move Windows to a new CPU architecture. Oh, that's not going to happen.

Steve: No.

Leo: Oh, that's not going to happen. The big problem is that Microsoft has packed almost everything into the kernel space - oh, this is what you were talking about - to give them the performance advantage in the '90s. But now that rooster is coming home. I really think Apple has an edge on everyone else thanks to the heritage of NeXT, that is now the heart of OS X. Time will tell, but so far so good. The developer environment is second to none. It's a nice hybrid of proprietary and open source software. I could go on, but I'm sure you either already know or can research it yourself. Yeah, it uses the mock kernel. What is a microkernel?

Steve: The idea with a microkernel is, I mean, as it sounds, you have a very small trusted environment that provides the minimum set of services, so literally a very small core. The beauty of that is that it's much easier to design a trusted, privileged, high-security environment which is small than it is to, like, do something like Windows in that kind of environment. And then the idea then is that everything else that the operating system needs is an external non-trusted module which uses that set of core services. So things like memory allocation and memory management, process creation and thread creation, the scheduler that jumps the processor around among all the processes that are running, you know, those fundamental services of the operating system are the microkernel. And then everything else is - and that's trusted. That has to work. That has to be bulletproof and provably, knowably secure.

But then you design the system so that all the other things that the OS provides, even though they are traditional operating system services, they're provided as sort of add-ons outside of this microkernel. And because they're outside, problems in them cannot affect the rest of the system. So there's a much better sense of containment. And the bloat that all of these systems end up acquiring over time, the bloat is not kernel bloat, it's user space bloat. It's outside of the kernel. So if there's mistakes in the bloat, as there inevitably is, they can't hurt you nearly to the degree that mistakes in a bloated kernel will.

Leo: Linux uses something called a "modular kernel." Which is I guess kind of a little bit of both. You have a kernel, but you can add modules, compile them in to add capability. I guess it's not a microkernel really.

Steve: Hasn't Windows 7 - isn't there something modular about Windows 7's architecture?

Leo: Yeah. They claim to have a modular kernel now and a modular system.

Steve: Well, but they have that buzzword, then.

Leo: Yeah. You know what they're doing, which is interesting, I'm sure we'll cover this in future shows, they've decided with Windows 7 to build in compatibility basically in virtualization. So you know they've announced that they're going to include with the professional versions...

Steve: Yup, XP.

Leo: XP Virtual. So maybe this is their way out of this corner they've painted themselves into is, well, we could still be compatible. We'll just do it in virtualization.

Steve: Well, remember that - I think it's in Windows 7 that all support for 16-bit code goes away. And I think this demonstrates that Microsoft is just - is phenomenally unwilling to break anything. And so they're saying, wait a minute, I mean, no doubt there is still 16-bit code somewhere that has to run. And I'm sure that there are some corporate customers that are saying, look, we've got 16-bit code written by people that have wandered off so long ago their forwarding addresses no longer forward to a valid address. And so someone somewhere is saying we have to have 16-bit compatibility. And I'm sure that Microsoft said, uh, okay, let's - we'll give you virtual - we'll give you XP in a VM because XP still runs 16-bit code. And that way you'll be good.

Leo: It's a good solution actually.

Steve: Yeah.

Leo: It's how I run XP is in virtualization. Andy in Latvia and, if that were not enough, Peter Katt in Syracuse, ask good and obvious, in retrospect, questions about Conficker. Andy writes: Hello. I was wondering how hard it is to trace the actual people who have registered the domains for the Conficker server used by the Conficker clients? That's where those updates occur. And Peter asks: Steve, in last week's Security Now!, quite informative as always, wouldn't it be possible to track down the person or people behind Conficker by finding out who registered the domains it's using? Yeah, there are a lot of them but I think someone would be following the money, even if they're using stolen credit card numbers to pay. The registration orders must be coming from a computer somewhere. Couldn't its IP address be recorded?

Steve: These two questions represent many that I received. People saying, hey, wait a minute, why not just track down the guys that register the domains? Well, now we've got 50,000 domains, only one or two or a handful of which need to be registered. But more importantly, unless you are registering or, for example, acquiring an SSL certificate where your identity needs to be covered, and there is no SSL dialogue going on with Conficker, it uses standard unencrypted connections even though the payload that it is passing back and forth itself is encrypted.

It turns out, I mean, it is very easy to completely anonymously register a domain. We know, for example, that the TOR network, The Onion Router network, is very good about anonymizing IPs. So I wouldn't be at all surprised if whomever is setting up these IPs is bouncing through TOR, or maybe has their own network, or may be using their own existing Conficker fleet as an anonymizing service, going through somebody else's machine that they have control of and doing that work there. And there are lots of ways, I mean, there are so many registrars around, especially when you're now expanded to 110 different domains, that it is entirely possible to forge an identity and register a domain with absolutely zero credentials.

Leo: Well, and I don't know if this is germane in this situation. But GJ Dunga says in our Stickam chatroom, remember, you don't have to register the domain, you just have to infect the computer that's providing that - serving that domain. In this case because they're random domain names I think you in fact have to register that domain. It's unlikely that anybody's going to have Q37914988873219.

Steve: Yes, very good point. And, yes, yes.

Leo: But other worms could take advantage of that fact by pointing you to a site that they know they can compromise. In fact, that has happened in the past.

Steve: Yup.

Leo: Bill Everson, Green Bay, Wisconsin brings news of an IronKey update. Remember that's the USB thumb drive we really liked that had all those security features in it.

Steve: And it's now got one new very cool one.

Leo: Oh, I'm liking this. Steve, in case no one has told you about this yet - I'm going to order one right now - IronKey now supports VeriSign's one-time passwords. I downloaded the update for my IronKey - oh, you don't even have to buy a new one. It just updates.

Steve: Right.

Leo: And I'm now able to access my PayPal account with either the football or the IronKey. Yes. Since I have both activated, the IronKey can't logon automatically since PayPal brings up a screen that asks me which token I want to use. Yeah, that's - me, too. But it is a simple matter to select the IronKey token and have it manually generate the OTP. That is cool. So now you can carry a one-time - this basically makes it like a YubiKey; right?

Steve: Yes. What's happened is VeriSign has opened up the SDK. They have created a software development kit for people who want to do this. VeriSign - oh, and also, Leo, there's now an iPhone app from VeriSign.

Leo: Oh.

Steve: So you can install this on your iPhone and...

Leo: Even better, so it can generate the passwords, too.

Steve: Yes. And they've got a whole bunch - RAZRs, and there's a long list of them. I've got this on my - I didn't have a chance to follow it up in detail, but I did get news directly from VeriSign because the guys thought that I would - that our listeners would find this interesting. So first we've got IronKey. Now iPhone and many other phones. And I'll have a - I'll do some research about...

Leo: Oh, it's free, too. Wow.

Steve: Yup.

Leo: So it has a credential ID that I would then register with PayPal.

Steve: With VeriSign.

Leo: VeriSign. Okay. I'm already registered using their card.

Steve: Right. And all of the providers who are using VeriSign as their backend authentication, this is then compatible with.

Leo: Now, point of order, you might be a little less secure because you're putting it on your iPhone. If somebody gets your iPhone, they've got your card.

Steve: Yup, good point.

Leo: Yeah. However, that's fantastic. It's called VIP Access. Which isn't that what they call it on the web, as well?

Steve: Yes. VeriSign Identity Protection.

Leo: I'm downloading it right now. That's awesome. I almost bought an IronKey, but I don't even need that.

Steve: Nope.

Leo: Wow. Cool. I'm sorry. I should probably continue with the show before I start installing that on my system. This is Bill Gearhiser in Boca Raton, Florida. He thought the Conficker podcast left out something crucial. Steve, your Conficker podcast was fantastic and fascinating, but you left out the most important part. How do we know if we've got it?

Steve: Very good point.

Leo: I guess so.

Steve: I did make the comment that Conficker has always been blocking DNS access for security-related sites. So, for example, you cannot go to Symantec.com or McAfee or a number of other sites. But I thought this was also a good time to remind people that we all, we all Windows users, have the Microsoft software, the Malicious Software Removal Tool in our systems; and it's always being updated monthly, on the second Tuesday of the month. And remember, running it manually is fun. It has the option of doing a deep scan. And it's as simple as just clicking on Start and then Run to bring up the little Run dialogue box, and put in MRT.exe, MRT.exe. Hit Enter. A dialogue pops up. And among other things, there's a link right there that lets you look at the list of all the things it's aware of. And in alphabetical order, Conficker is among the many things that this Microsoft Malicious Removal Tool is aware of. So you can absolutely just bring up the little Run dialogue under the Start menu, type MRT.exe, fire this puppy up, and give it a deep scan. And then go away for dinner because it'll take a while. But that way you can manually run this test and be sure.

Leo: MRT. And Microsoft's MRT has had that since January, I think.

Steve: Yes. It's been aware of Conficker for months. And it is being updated. Oh, also, when it pops up, make sure that you've got the current version. We did run across a Q&A some weeks ago where someone was saying, hey, I popped it up and it said November of '08, and this was in March of '09.

Leo: Oh, yeah, yeah, yeah, yeah.

Steve: I just popped mine up, and sure enough it says April 2009. So make sure you've got the current one because that would be an indication that Windows Update is not updating.

Leo: Yes.

Steve: And we've seen that happen, too.

Leo: Yup. Very important. D. Larson in Portland, Oregon and Todd Boring in Houston, Texas both wonder what Conficker is up to. D. Larson writes: Love the show, guys, especially the last show about Conficker. I'm not sure what you could call - I am not what you could call an especially technical person. My question is, what does Conficker do once it's on your computer, and how can you tell if it's on your system? Sounded like right now Conficker is lying in wait and isn't actually doing anything besides updating itself, infecting others, and staying alive. Is it logging keystrokes? Is it tracking traffic to send home later? Close the loop for me.

And Todd says: Steve and Leo, thanks for the great podcast on Conficker. Fascinating stuff. Makes me wish I were a programmer working for the NSA or something. That's true. It's really an interesting field. You talked at length about how Conficker propagates and how it updates and defends itself. But I'm still wondering, what does it do? It seems apparent that it has botnet potential as its owners could send an update command for all infected machines to, say, DDOS a particular website or corporate Internet linkages. Any proof of it gathering, then delivering data to the author from the infected system? What does Conficker do, Steve?

Steve: Well, these are great questions because I sort of implied but didn't explicitly say that, until E, which is where we are now, Conficker version E, Conficker itself, as D. Larson said, basically was just establishing itself. It was creating a beachhead of this infection among, well, 10 million machines, if you count all the ones Microsoft has removed it from. That is, the MSRT we were just talking about has been effective in removing it from, not tens of millions, but more than 10 million machines. So there's been this battle going on. Versions A, B, C, and D, they pretty much just existed to exist. It was this technology, as D. Larson suggests, where is it just sitting there infecting other machines, updating itself, and staying alive? Yes.

Now, E is behaving differently. It's installing - there are two different reports. There's a spam botnet called Waledac, which is well known, which is by the same people who did the very famous Storm botnet. We remember the Storm worm from times past. And so this Conficker E is installing this Waledac spam botnet. And a different report indicates that it is also installing some malicious nagware, Spyware Protect 2009 specifically, which is one of those things that pops up and says, oh, we've just scanned your computer, and it looks like you have an infection. Anyway, it duns you, just bothering you to death with these popup dialogues until you pay $49.95 for a download that does nothing. So it's basically just sort of bribery ware, I guess.

The other change is that the E variant, which has now been analyzed and looked at since we talked about C in detail last week, it has restored the original vulnerability, that is, it's using the original unpatched machine problem to spread. So whereas the later variants of Conficker were not using the original, they were not attempting to exploit the original unpatched vulnerability that Microsoft fixed in October of '08, E is doing so. So it is now emitting probes, looking for newly non-patched machines to take over.

And the sense is, in the security community, is that some of the later behavior, which seemed to be less aggressive, was in fact resulting in a decrease in Conficker population, that the author may have been getting a little worried that some of the games he was playing with changing its survival and replication strategy weren't working. And so he's gone back to traditional .A style spreading, which is what really made Conficker as potent as it was.

Leo: Very interesting.

Steve: So the bottom line is, the answer is, Conficker until recently, well, Conficker itself is still just a delivery mechanism. It's a worm that exists to exist, to communicate, to build itself. Yet, because it has the ability to install and to acquire updates to itself and also other third-party packages, it has now begun to essentially generate cash. And ultimately that's what these things are for these days, is until now it was sort of a technical curiosity. Now it's installing a spamming botnet, this Waledac botnet, and this dunning popup Spyware Protect 2009 stuff in order to begin generating some money.

Leo: I think they're like most Internet startups. They're content to establish a base of users before they go for the monetization. Or public offering. Barry, question 11. Barry, working somewhere for the government in Minnesota, writes: Hi, Steve. I've been listening since the beginning of the show, and I'm a fan. But I must take exception to your comment in Episode 193 that you haven't yet seen a smart government person. You didn't say that, did you?

Steve: Yeah, kinda.

Leo: Steve.

Steve: I kinda did.

Leo: Steve, Steve, Steve.

Steve: It isn't what I meant. But let's...

Leo: Of course it's not what you meant.

Steve: Let's hear Barry out, and I will explain.

Leo: Yeah. I know that's not what you meant because I know you know better than that. As the CISO of a large state agency, I have the privilege of working with many very smart staff and colleagues. My agency is involved in the healthcare and EHR world, and I share your concerns about the push toward online medical records and the associated security issues. However, there are clear medical benefits to the consumer, and that's the main driver. You and Leo answered your own questions within a minute or two of your statement. However, you didn't retract that statement.

The push to move a product or service to market and accelerated development timelines drive so much that we in the security industry do, whether in government or business. In the government sector we're also subject to the whims of elected officials, perhaps not unlike those of corporate executives. It's the rare organization that truly builds security in, although we're all united in that quest. This is a common theme in our industry, and one you've discussed at length on your fine show. I suspect we'll find that the compromised Pentagon computers were Internet connected because of a requirement to make them accessible to external contractors. The breach itself may have been caused either by compromising the contractor end point, or the remote access process, perhaps via social engineering or a weak password or a patch that wasn't applied because of a possible incompatibility with the development code.

All of those things happen. I also suspect that appropriate security personnel warned about that possibility but were overruled. So please recognize that those of us responsible for security of government systems, assets, and data are doing all we can, with minimal resources and budgets, to secure that data and maintain citizen confidence. Keep up the great work, minus that one statement. Well said.

Steve: I really, really apologize. I know that I said that because Barry was not the only person to write. I heard from many government people who said, how could you say that all government people are stupid? Or, I mean, I didn't use that word. But I'm - and I'm thinking, how could I have said that? And I know exactly what I was thinking at the time. And I was thinking about the legislators who I see interviewed that are just...

Leo: There you go. That's a different matter.

Steve: ...I mean, clueless about this stuff. I mean, I absolutely meant no slight to the actual people like Barry that are on the ground, as they say now, doing this work. Of course not. It was the people, frankly legislators, who I just haven't seen one that understands this. So, and I know even they, too, are dealing with a bureaucracy that I can't even comprehend, that would just make me shoot myself if I had to deal with that on a daily basis. So they're doing things I can't do, either, in a different way. So anyway, I have the greatest respect for government employees who are working as hard as they can, like Barry, and facing, as he says, resource constraints that are probably much tighter even than the corporate world has. So I certainly apologize for having given that impression. It was not what I meant.

Leo: As do I. Neither one of us believes that, for sure.

Steve: No.

Leo: No. Sam in Alsager, U.K. is smarter than your average monkey and the author of our last question. Hi, Steve. You said in a recent Security Now! episode that Visual Basic, quote, "allows monkeys to program." You know, these things happen. I am a computing student and a technology enthusiast. And I feel that's a bit harsh for someone like me or someone in a similar position. I know you started programming at my age, 16. But times have changed a lot. I've grown up very literate with computers. However, programming is very different. At times I've racked my brain for hours to get a program to do something or have to play around with variables or data types. And so I feel it's a bit harsh to assume that anyone, even monkeys, can program. There's been no evidence that monkeys can program, by the way.

When I started programming with limited help, and with the web being clogged up with useless crap doesn't help a lot, either. Not everybody is an amazing Assembly language programmer like you are, although we would like to be. And I can well understand that, as one further increases their knowledge of computers, we tend to grow farther away from the less technically literate. And I understand less why they may not know something or understand something. So I end up by saying, I understand where you're coming from, but please try to be a bit more understanding of the people just beginning with this stuff. This doesn't change how I feel about the show. I just wanted a shout-out for the people who are getting started or a little less technical. Thanks a bunch for yours and Leo's hard work. Sam. Sam, that's a great letter. And you're not a monkey. But you really should take a look at Python. Okay, I'm sorry. Go ahead.

Steve: Now, okay. I did say that monkeys could program Visual Basic.

Leo: But not that all Visual Basic programmers are monkeys.

Steve: Exactly.

Leo: It was a logical...

Steve: Thank you, Leo.

Leo: ...a logical error.

Steve: Yes. And nor do I think that all Visual Basic programmers are monkeys. I should explain also something that is true, which is buried now in the depths of history. And I doubt that many of our listeners will remember, although I know that Nevet Basker, who was the original product manager for Visual Basic, remembers, I am largely credited, that is me, Steve Gibson, Security Now! podcast, with putting Visual Basic on the map.

Leo: What? I didn't know that.

Steve: Yes. They came up with it. I was writing the InfoWorld column. The product manager, a neat gal named Nevet Basker, came down to Southern California to show this new thing to me. And we were all blown away. I mean, this was the first time we saw anything like this, where you had a little toolbar of controls, and you dragged them over and stretched them out, and then you wire up event handlers for the various things. And before you know it, you have a program. And I'll never forget, one of my developers, because this was when GRC was growing, and we were probably about maybe 20 people at that point, and I thought I could stand having anybody else write code - which turned out not to be the case. Thus we're three of us now. But this guy's name was Millard

Ellingsworth III.

Leo: I love it.

Steve: And Millard made a comment I will never forget. We were standing there, I mean, I was just in awe. And he said, oh, this is really bad. And I said, what? What do you mean? He says, "Anybody can write Windows programs now." He didn't refer to monkeys. But, you know, he talked about humans, any human.

Okay. I was so impressed that I wrote three columns. The next three weeks of my weekly InfoWorld column were about Visual Basic, jumping up and down. I mean, I was just - I thought, this is just fantastic. And I once, in one of my trips up to Redmond maybe a year or two later, I ended up having the occasion to have dinner with a group of Microsoft people, including Nevet. And she said that that following summer, maybe a couple months after I had written the columns in InfoWorld, she was in Europe, doing an European tour to introduce her product, Visual Basic, around to all of their corporate customers and various people there. And she said that every door was opened to her because people were saying we have to see what it is that Steve is jumping up and down about. So by all means, we'll give you as much time as you need to tell us about this amazing new Visual Basic that has Gibson just breathless. So in all fairness, that was then. I was tongue-in-cheek when I said that a monkey could program...

Leo: Clearly.

Steve: ...Visual Basic.

Leo: No ape yet has the...

Steve: Actually, yeah. I mean, it is...

Leo: Although an infinite number of monkeys, as somebody in our chatroom pointed out, typing on an infinite number of computers, would actually be able to write a program that would quote Shakespeare. So there you go.

Steve: Yeah. Okay.

Leo: Thank you, Eric.

Steve: Anyway, Sam, I apologize for the slight. I think it's great that you are programming at the age of 16, as I was.

Leo: That's the best time to start.

Steve: It is. It's a very different era, as you say, now than it was then. It was, you know, Assembly language was what I got into then because that's what was happening. I mean, there wasn't - we didn't have Visual Basic and the ability to drag controls out onto a form and stretch them to size and wire them up. Or I doubtless would have been doing that. I mean, I also think that Visual Basic has a place. It certainly, in terms of just getting the job done, I'm sure that Visual Basic is probably still the most used language on Windows of any because in corporate environments where you're not trying to produce a finished product with all the polish and everything, but you're just trying to create some internal access for your SQL database backend, you want to - people have all the requirements, and they're needing this and that and the other thing. And it's like, hey, just use VB. And there's nothing wrong with it. You can certainly write really good Visual Basic code, which has all the complexity and richness of code in any other language.

Leo: Oh, yeah. Oh, yeah.

Steve: Yeah.

Leo: Coding is coding. Although I still think, I think it'd be great for kids to learn Assembler today. I don't think there's any harm in doing it. It's actually a very easy language to learn. It's much easier than learning VB.

Steve: For me the advantage is that, first of all, I have run across many college references to things like the PDP-8 being the instruction set which is being taught. There are - people have written PDP-8 emulators for the use in educational curricula so that it's possible to learn the simple instruction set and solve some problems. I really believe there's something valuable about understanding what's going on underneath. And those tools are available, although it requires that you delay gratification. And it's certainly the case that you're not going to produce your typical, contemporary, state-of-the-art application quickly in Assembly language because that's just - because we're, you know, the old quote is standing on the shoulders of giants. I mean, there's so much that Visual Basic is standing on, all the way down to the processor, that most people want an application that looks like something that they're able to buy off the shelf. Visual Basic gives it to you almost without trying.

Leo: Yeah, yeah. And there are a lot of great student languages like Python where there's - Visual Basic gives you, as you say, all that UI access. And that's nice. And of course people should learn the concepts of programming before they get too excited about drawing Windows on the screen. Maybe not, I don't know.

Steve: Of course Pascal was designed - it was a language designed...

Leo: It was a teaching language, yeah.

Steve: ...by Nicholas Wirth for the purpose of teaching programming. And, I mean, it's still the one I wish had survived, as opposed to C. It's just, it's a little verbose with having to type BEGIN and END all over the place. I do like left and right curly braces a lot better. But still it produces beautiful code. I mean, educationally beautiful code. Which has a place.

Leo: Another plug for a Mac. It comes free with a development system that allows you to do a lot of the - it has a rapid application development tool, just like VB, so you can draw Windows and add code to it.

Steve: It's that Xcode pack; right?

Leo: Xcode, it's amazing. And you can code in Python, AppleScript, Ruby, all of these come with the Mac, as well as Objective-C. And we were talking about how strong, what a great operating system NeXT was. Part of the reason it was so portable was it was written in Objective-C, which is a really interesting extension to C that just really does a nice job. So all of that comes with a Mac, which makes it a very good computer for people wanting to learn to program. There's a lot of good stuff. There's a lot of good stuff out there. Alice, that Randy Pausch wrote at Carnegie Mellon, Alice.org, great language for teaching kids. Anyway, we've come to the end of our 12 questions, Mr. Gibson.

Steve: Well, and not a moment too soon.

Leo: Yeah, because I have to run. But I thank you so much. Always a pleasure. Next week do you know what we're going to talk about? Or is it a surprise?

Steve: We're going to finally, finally give our listeners who've been chomping at the bit for the SSL protocol episode what they've been waiting for.

Leo: Wow. That's great.

Steve: We're going to take all the building blocks that we've talked about, hashing and symmetric and asymmetric and signatures and all that stuff. We've got all the pieces we need now to look at, okay, what happens with, I mean, this is, as we've said, the number one most used security protocol there is because we all use it. Every time there's an S on the end of our HTTP, that's an SSL connection. How is it formed, how is it secure, and how does it work? We're going to do that next week.

Leo: Thank you, Steve Gibson. We'll see you next time on Security Now!.

Steve: Talk to you then, Leo.