DNS Rebinding

Description: This week, after catching up on all of the post-Black Hat and DefCon conference news, Steve and Leo plow into the detailed depths of "DNS Rebinding." Together they thoroughly explore this significant and fundamental weakness of the Internet's security.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-260.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-260-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 260, recorded August 4, 2010: DNS Rebinding.

It's time for Security Now!, the show that covers all your security needs. And here he is, the king of Security Now!, man who's been doing it now for five years, Mr. Steve Gibson of GRC.com. Hi, Steve.

Steve Gibson: I guess security really is a need. You introduce the show saying "covers all your security needs." And in fact I would argue that you probably can't really get very far these days, or not very safely, without having that need covered, so...

Leo: I think in some ways it's kind of a shame that you have to be a security expert, well, at least a mini security expert, in order to use the Internet and use your computer. And that's such a shame.

Steve: Oh, Leo, I hate that we have to have this podcast. I mean, I love doing it. But, I mean, I'm so annoyed that, well, I'm so annoyed that this is necessary. And today's episode is an interesting indication of a different sort of reason that it's necessary. We're going to talk in detail, as I promised a couple weeks ago, about DNS rebinding, which came back up into the news, even though it's 15 or 16 years old, that is, the problem is. It came back in the news because there was going to be a presentation, as there was, at the recent Black Hat conference, where there was a new approach that allowed malicious remote websites to take over people's local routers. And it used the trick of DNS rebinding. So I thought it was worth looking, sort of revisiting it. I don't think we've ever really covered it in depth, which I wanted to do.

But what's interesting is that this is a problem that is not about anything being broken,

not about a vulnerability, not about anything even being designed wrong. It's just that the system we've built was never, from its original concept, never built with security in mind. And there are ways to abuse technology that works the way it's supposed to, in ways that the original architects weren't defending against. It just wasn't in their mind. And, for example, this is so fundamental to the way DNS works that not even DNSSEC, the next evolution, not even signing the root node, and not even DNS security prevents problems with DNS rebinding.

Leo: Really.

Steve: Yeah. So...

Leo: So this isn't the Dan Kaminsky flaw. This is a whole different thing.

Steve: This is a kind of a "gotcha" with the kinds of stuff we're trying to do. And so it's a consequence of clever people saying, you know, if we did this a little differently, we could make something happen that people have been trying to prevent since Mozilla 2.0. Which is...

Leo: [Laughing] Wow.

Steve: I'm not kidding. Yeah, there was something, there was a technology that we've talked about briefly, but we have to cover it in a little more depth because it's tied into DNS rebinding, the so-called "same-origin policy" that scripting uses, which prevents scripts from sort of being able to go do things to sites that they didn't come from. It's like sandboxing for scripts. And there's been - it was in Mozilla 2.0 that this notion of same origin policy was first implemented because the original Mozilla guys realized when they created JavaScript that scripts were very powerful. Well, we know. That lesson is something that we seem to visit every single week of this podcast.

Speaking of every single week, this is #260. And it's been a phenomenal amount of controversy over in GRC's newsgroups about when it is that we actually finish year five and start year six. And so I looked at the calendar. Actually I looked at our archive of prior podcasts. And what struck me as I was - we were talking about this a little bit before we began recording - is this podcast used to be about 20 minutes long.

Leo: No.

Steve: First one was 18 minutes long. And they stayed about that long for quite a while, and then began to grow in, I guess, covering more current events. I don't think, I mean...

Leo: We weren't doing news in the first few years, I think.

Steve: I think that's exactly right. We weren't covering that so much. It was mostly just

topical stuff.

Leo: Right. And we didn't do Q&As in the first few years.

Steve: That's true.

Leo: So we were just saying, okay, here's how cryptography works, or here's - so in a way it was just the chunk, the kind of the end chunk of the show that we still do, but now we've added a lot of other things. I hope you all like it. I think, you know, remember, this is the second show we ever did on the TWiT network. I started doing this right after TWiT. And half the time I did it in Canada with you.

Steve: Yup.

Leo: We did them on the set; you know?

Steve: I won't ever forget standing on the set in Toronto, and you said, "Hey, Steve, would you be interested in doing a podcast about security?" I said, "A what cast?" Never heard the term. That was in March of...

Leo: 2005. Wow.

Steve: ...'05, I guess, yeah, yeah.

Leo: Yeah. And I think at the time, when we first started, I didn't know how long a podcast should be. And all of the shows have gotten longer. Not only because we're wordy sons of guns, all of us, but also in reaction to the fact that I always hated it that I only had six minutes with you on TV. It was always rushing. We never really covered the subject thoroughly. And audience support for longer shows. I've always been saying, and I'm still open to the idea, is this too long? And as long - I think what people want is it should be the length of their commute.

Steve: Yes.

Leo: No shorter.

Steve: If everyone would please drive exactly 90 minutes...

Leo: No shorter. No shorter. Longer is okay because you just stop, and you pick it up. But what you don't want to do is listen to a show, and you're still in the commute, and now you have to find another show. You want - it's my sense, and I'd

love to get feedback from people. But that's my sense of it.

Steve: It's like finishing a book when I'm in the middle of my stair-climbing workout, Leo. Nothing worse than that. It's like, okay, now what am I supposed to do?

Leo: Somebody in the chatroom is saying, can you tell the story of how you two met? We won't go into great detail. We've mentioned it before. But we met on The Screen Savers. We were covering something called "The Click of Death," which was a problem that ZIP drives had. They'd go click-click-click, and then the drive was damaged in such a way that it would damage every single ZIP disk you'd put in there. So you would destroy your collection as you tried to find a disk that worked.

Steve: Right.

Leo: And Steve wrote a program, Trouble In Paradise; right? Was that it?

Steve: Yup, TIP, Trouble In Paradise.

Leo: And we put you on the show, Kate Botello and I. So it was in 1998. It was the early days of The Screen Savers.

Steve: Yeah. And you and I knew of each other, but we had never...

Leo: Oh, god, I'd read you religiously in your InfoWorld column. I was a huge fan of Steve Gibson. So, as has been the case my entire career, people like Jerry Pournelle, John C. Dvorak, you, you know, meet you, and it's like, oh, I'm meeting an idol. It was really, really exciting to meet you.

Steve: Well, and my favorite memory was when Kate discovered ShieldsUP!; and it was, like, her turn to do a segment of the show. So you were sort of the sidecar for that particular phase. And she said, "Yeah, this is over at GRC.com." And it didn't register with you immediately. And so she was starting this, like, "There's this really neat thing that checks your ports." And you said, "Wait a minute. Steve Gibson? He's SpinRite, you know, he's the hard drive guy." And she says, "And apparently security, now, too."

Leo: Uh-huh.

Steve: So, yeah, that was the beginning of that, which was fun.

Leo: Yeah. And I consider Steve one of my best friends.

Steve: Ditto.

Leo: And so we're celebrating five years of shows, but we've known each other for 12.

Steve: Well, and speaking of five years, I did the math last week where I said, okay, 365.25 days per year because every fourth year is leap year, so we get an extra day, which we divide by four because it's every four years. You divide that by seven days per week because I think we're all agreed that each week has seven days. You never have a six-day week.

Leo: I think we're in agreement on that.

Steve: Never have an eight-day week.

Leo: Okay.

Steve: That would really throw things off. So you divide that by seven days per week. And it does not give you 52 weeks in a year, it turns out. It gives you 52 point whatever it was, 197, or 179, I think. Now, if you then multiply that by five years, so exactly how many weeks there would be in five years, you get 260.8 something or other. So rounding that up you get 261, which would say that next week's episode, #261, would be ending year five. So we'll be beginning year six on 262. And lo and behold, our first episode that we recorded back in 2005 was on August 19th, Thursday, August 19th. And that will be Episode 262 is August 19th of August 2010.

Leo: We'll have a cake.

Steve: So the math works, and I hope the controversy is finally resolved.

Leo: I love it when a plan comes together.

Steve: But I do stand, I stand corrected that for many weeks I was saying 52 weeks a year, 52 weeks a year, and multiply that by five. It's like, no, because there are those little annoying leap years. And they add up, so.

Leo: Well, they only add up when you've been doing a show for five freaking years. That's why they add up. I mean, the first couple of years it didn't matter.

Steve: We'll be leaving that behind soon.

Leo: We've got some great stuff to talk about. We're going to talk about, as Steve mentioned, DNS rebinding. We've got security news. We've got patches galore, of course.

Steve: Oh, and we've got - Black Hat and DefCon were last weekend, and lots of follow-up from that, too.

Leo: Yeah. I'm dying to hear what you think about that.

Steve: Not too much fallout, so that's the good news.

Leo: And, you know, I'm just getting news now that - we heard that India was having trouble with BlackBerry, that it wasn't secure, they were worried. It was banned in India. Now Dubai and the United Arab Emirates banning BlackBerries. And now we've just learned that the European Commission is abandoning its BlackBerries because - and going to iPhones because they don't deem the BlackBerry software safe enough. And they think that the U.S. actually has a backdoor into it.

Steve: Well, yeah, we have BlackBerry discussion actually in our notes.

Leo: We will talk about that, too. Now, moving on to Security Now!, you have an update on this LNK thing. Thank goodness.

Steve: Yes, well, big news, I mean, in terms of security updates. And I'm presuming that all of our listeners know this by now because this happened Monday. As we were hoping, and maybe on the border of praying, Microsoft responded with what they called an "emergency out-of-band" - that term still, it ought to be out-of-cycle, but what the heck - emergency out-of-band patch for this shell LNK vulnerability that we've talked about extensively, so I won't go into it in too much detail. Everyone I'm sure knows about it.

This was the big problem where Microsoft's only solution was to disable the displaying of all shortcut links. If you used their Fix it button, it would turn all of your shortcuts within your entire system into white featureless rectangles. This is also the thing where Secunia had come up with a temporary interim filter to solve the problem in a less UI-disturbing fashion. Microsoft released the patch on Monday. So I would imagine that people would have seen their little yellow shield appear.

In any event, if your system didn't get notified, or if you don't have Windows Update set up for automatic updates, absolutely you want to get this patch installed. You can, after that, safely unfix it from Microsoft's Fix it button. And if you installed the Secunia temporary interim fix, you can remove that using the Add/Remove Programs list in Control Panel. And this little nightmare is behind us. The use of it was going up exponentially. Many other uses were being found, aside from the first-seen attacks. So it's a good thing that this was resolved.

Now, the thing that still bugs me is that Microsoft didn't acknowledge any problem before Windows XP SP3. So in their list of systems affected, they're not even saying that

Windows 2000 has a problem, or Windows XP SP2 has a problem. They both do. As far as we know, NT does. But so I'm annoyed that they're not saying these, like, everything has a problem, but these are the ones we're going to fix. They're just ignoring the fact that earlier systems have a problem, but they're not going to fix it.

Leo: Well, we know nobody uses earlier versions of Windows.

Steve: Uh-huh.

Leo: They've all upgraded.

Steve: We'll be talking about IE6 in the U.K. here pretty quickly. Now, I saw one mention in the SANS security newsletter. Their dean of education, I think is his title, Johannes Ullrich, who runs the Internet Storm Center. He made - he just - there's one little line comment that SP2 is being silently supported by this fix.

Leo: Oh, interesting.

Steve: Now, I've not verified it. I thought, well, okay, I care about that because I'm on SP2 still, with a system that once reacted badly to SP3, so I backed off on that. And I checked Windows Update. It didn't have any happiness for me. So I don't - I have to pursue this a little bit further. I will, and I'll see if I can find it. And if I can, I'll let people know. I ran across some people who were saying, hey, I'm still using Windows 2000 because it works just fine. So it's like, yeah, I understand that.

Leo: Except it doesn't, I mean, at this point.

Steve: Except, yes, except this is a long-term threat. And the bad thing is even the Secunia fix won't fix Windows 2000. It will fix, in the way that they offer, XP SP2. So you can use it on XP, just not any earlier than XP, which is too bad because otherwise it might be all we have, if, in fact, Microsoft didn't fix SP2. I'm not surprised they, Microsoft, would have fixed SP2 silently, if in fact they did, because that would just, I mean, it makes sense because there are people who have known problems with SP3.

Leo: Yeah, I can understand saying, oh, look, we don't want to support software after a certain point. That's understandable. Software after a while gets out of date and so forth. But just to protect the Internet there's a certain responsibility you have. Even, to use another example, if a car is 20 years old, but you discover that the brakes fail, you still have a responsibility...

Steve: Even if it's out of warranty...

Leo: ...even if it's out of warranty to tell the owners, look, we've discovered a

problem, and here's the fix. So just for - you don't have to fix bugs. But you have to fix security flaws. You have to. And I think you have to do it as long as those operating systems continue to be used, not just for the owners of the operating system, but for everybody else.

Steve: I'd buy it. I mean, I could say, I mean, I could even see it being reasonable if Microsoft said, while these OSes are under our security umbrella, we're going to fix them for free. After that, you're going to have to pay for it. Now, of course that would cause all kinds of problems, too. But if it's a matter of buying it versus always having this really bad security problem known, I'd fork over, you know, five bucks to get the patch.

Leo: I can't imagine that the cost of, once you've got the fix for...

Steve: They know what it is. They know what's wrong. They fixed it everywhere else.

Leo: I cannot imagine that it's so difficult to fix it for these other versions. I think it really comes down to we are trying to push people to upgrading. And some of it's to make money, of course. But some of it is just because we don't want to have to support these versions at all. We want them to be gone.

Steve: Well, and to be fair to Microsoft also, we know that they have been doing a good job - now, this is me saying this - a good job in increasing the security of this Windows platform moving forward. We've got address space layout randomization. We've got the execution prevention where they're doing more work with protecting the stack. We've got UAE. I mean, there are many things that they've been doing that are enhancing the security moving forward. So it is in fact in people's own best interest, where they can, to move forward. And I'll be on Windows 7 at some point. I'll just jump over the dead carcass of Vista, happily, and go directly to Windows 7. So...

Leo: You'd be right to do so, I think.

Steve: Oh, yeah, exactly. Again, I think that's a very good point. So one of the things I had my eye on the most was this concern, which we discussed in some detail last week, but we didn't have all the details because it was upstream of the formal presentation at Black Hat of this WPA2 hack or crack, which was supposed to - it generated a lot of press, and lots of people were wondering what the story is. Believe it or not, it's now being criticized within the security community as a publicity stunt.

Leo: Really.

Steve: I mean, it was - it turns out it's exactly what I described before we knew any details last week. And even, like, people in the Wi-Fi Alliance, now, you can imagine that they have some bias of wanting not to believe that this was anything big. They're saying, this is not news. Everyone knows this about the way WPA...

Leo: Was it brute force? What was it?

Steve: It was the idea that, if you were authenticated on a WPA or WPA2 - because it doesn't matter whether you use, for example, a radius server for producing per-client passwords, or you use a single password for the whole access point - if you're a client associated with a WPA-protected access point, then the groupwise temporal key, or temporary key, the groupwise temporary key which all clients of a single access point share, in order for them to do things like broadcast to each other, that allows you to essentially do an ARP spoofing, that's all this turned out to be was ARP spoofing, in order to intercept someone else's traffic.

The problem is that traffic is still encrypted with their private key. I was thinking maybe these guys had come up with something, and this is why I was, like, withholding judgment last week. Maybe they'd come up with some way of changing what's called the "pairwise temporary key," or getting the other client to divulge it to the attacking client. They didn't. They just said, well, we can filter traffic. I mean, like, we can filter traffic that we can't decrypt. And so other security consultants, because I was wondering what the fallout from this was, they were just, like, saying, well, this was nothing. I mean, this is nothing new. This was a publicity stunt.

So there is a clever hack that can be used against a secured access point, which we actually did talk about years ago, where, if you convince another client that you're the destination for its traffic, it will send that traffic meant for you to the access point under its own encryption. The access point, seeing that it's meant for you, will then decrypt it into plaintext, reencrypt it under your key, and send it to you. So you've got the other person's traffic, the other client's traffic that you've received under your key, courtesy of the access point in the middle decrypting it and then reencrypting it.

But that requires something known as "inter-client communication," which is explicitly and by policy normally disallowed on an access point. All access points have an option to - and it's normally defaulted - to prevent inter-client communication. In which case that particular problem, which has been known about for years, thus the reason that an access point will not forward traffic between clients, is normally enabled to prevent that. So again, all these guys could have done was to receive traffic that they have no visibility into, traffic directly from another client, because they'd done ARP spoofing, so the client is sending it to them instead of to the access point. But they don't ever get the other client's pairwise temporary key.

Which is why security consultants universally said, okay, so, what are you going to do? Yes, a denial of service attack. You could use this to cause another client to lose connectivity. That's as far as anyone can see, and now we understand everything that these guys were showing, that's the limit of what this could be used for is causing them to lose contact, another client to lose contact with the access point by redirecting their traffic to you or to somewhere else. And it's like, oh, okay, well, and that's only if you are already authenticated on that access point. So it's an insider deal. It's not somebody on the outside that can do anything because you first have to be authenticated to the access point to get the shared key.

Leo: Well, forget it, then.

Steve: Which you then use for ARP spoofing. I know, it's nothing. I mean, yes.

Leo: You've already got access. So now you can get access.

Steve: Uh-huh.

Leo: Who cares?

Steve: Well, you've got access, so you can annoy somebody else who also has access by causing them to, like, have to reassociate or relink to the access point. Okay, fine.

Leo: Big deal.

Steve: So we're not worried. The other big piece of news was that a mistake that's been found in Apple's PDF rendering engine, when it's rendering Type 1C fonts in PDF files, because Apple has its own PDF renderer, doesn't use Adobe's Acrobat for PDFs. A mistake there has been used to create a vulnerability, or to exploit that vulnerability, to easily, I mean, with remarkable ease, jailbreak just about any iPhone or iPad. And Leo, you've got more experience with this than I have, so...

Leo: Yeah. We did it, it's funny, we did it on Sunday. Brian Brushwood on TWiT was brave enough to do it. I tried to do it on my iPad, and at the time there was a bug that prevented that. They fixed that, so I did it on Tuesday on my iPad. And while there may be some little issues, the jailbreak itself seems to work just fine and be harmless. However, as you're about to tell us, the fact that you can do this is a significant security problem.

Steve: Exactly. So it's weird because the press was carrying this as, oh, look, now it's easy to jailbreak your phone. You'd literally go to jailbreakme.com.

Leo: That's like saying let's all go to hacker.com.

Steve: Yeah, well, and it's interesting, too, because, I mean, I went there with Firefox, just jailbreakme.com. Go with Firefox, and you see what looks like a little entry screen on an iPhone app. And it would be like a "slide here to jailbreak your phone."

Leo: That's exactly what it looks like on the phone.

Steve: Yeah. And you can also check, I think somewhere else I was able to go, like for more information. And on my non-iPad or iPhone browser, that is, on Firefox, you get this long strip of website, or like webpage, that would normally be what you would see on your iPhone if you scrolled along with the iPhone, which explains what's going on. And it's got - it's open source and GPL this and that, and they use compression, and so they're giving credit where credit's due and so forth.

The problem, though, is, well, okay, first of all, so that's what happened. This was also disclosed at last week's conferences. And the concern that everyone has is that you can - essentially you are using this font-rendering bug which is now well known publicly to run arbitrary code which is sort of part of the font. So the code is bundled in with the font. And this mistake in Apple's rendering of the PDF causes the code to execute due to a heap or a buffer overflow. Which is to say, jailbreaking is only one thing this can be used for. This is not a jailbreaking vulnerability. This elevates this Safari to root privilege.

Leo: Exactly.

Steve: It gives Mobile Safari the ability to run as root in your phone, breaks it out of the sandbox, and then lets it do anything it wants to, that is, anything the attacker wants it to do. So the good news is, there could hardly be anything that Apple is trying to fix faster than this.

Leo: You know what the funny irony is, that once you've jailbroken it, there is a fix in the Cydia store which you can now have access to for the PDF vulnerability. So you can - the way to fix, right now, until Apple does an update, is to jailbreak your phone. And as far as we can tell, jailbreakme.com is safe. I'm not, I mean...

Steve: Yes.

Leo: I'm not vouching for it, but I know Saurik, I mean, these people are fine and honorable people, and it's legal to do so, as we know now.

Steve: The DMCA, yes, Apple had been threatening people using the DMCA. And what, about a week or two ago the ruling came down that, no, jailbreaking of your own phone is something you're entitled to do. The DMCA will not protect you.

Leo: Apple says, not unreasonably, we will void your warranty.

Steve: We're still mad at you.

Leo: Don't come crying to us.

Steve: We're still mad at you.

Leo: But we can't stop you. And you should be aware this is a risk, security risk. But on the other hand, this is also the fix, which is kind of funny. Now, we don't know what other holes are created by this, either. So that's another matter entirely.

Steve: So you jailbreak your phone. Then you can use an unauthorized app to fix the vulnerability.

Leo: Exactly, to fix it, yup.

Steve: Yeah, okay.

Leo: Isn't that funny.

Steve: They ought to just put that app in the store and let people fix it until they make it official.

Leo: Gee, what a thought.

Steve: Well, also in Black Hat, which gave us all kinds of material this week, it was revealed that there's some nasty wallpaper which has been downloaded approaching a million times, at least many hundreds of thousands of times.

Leo: Wow. I didn't realize it was that much. Wow.

Steve: Yes, it's approaching a million, Jackeey Wallpaper, from a Chinese site called IMNet. It turns out it's free wallpaper. It presents you with your choice of many copyrighted, stolen intellectual property items. And what was discovered was that it was, in the background, collecting phone numbers, SIM card numbers, text messages, subscriber IDs, and voicemail passwords.

Leo: Oh, boy.

Steve: And mailing them back, sending them back to www.imnet.us, which is in Shenzhen, Guangdong, China. So this is a cautionary note, just in general, about not installing apps that you don't trust. Now, it's true that Android, as we know, is a much more open marketplace than the iPhone store.

Leo: Nobody's vetting it, as far as I can tell.

Steve: Well, and my fundamental problem with the notion of vetting is that nothing prevents Apple from being fooled.

Leo: Right, as they have been. They have been.

Steve: They have, exactly, Apple has been fooled. Unless they receive the source code and study it, they're not going to be able to tell, for example, what behavior an app could develop after a certain date.

Leo: This happens all the time. There are apps in the iPhone store to do things like enable Emojis. They pretend to be flashlight apps, but they enable Emojis. Or they do tethering. And Apple eventually learns of these, pulls them off. But that's exactly the point is that there's no real way to look at these and say, well, what is it really doing? And in fact there was a hack that allowed iTunes passwords to get leaked out through a malicious app on the Apple store. So just any time you put applications on an operating system, I don't care if it's a phone or a desktop, you run a risk.

Steve: Yes. And I was going to also add that, as we know, people have heard me say this many times, even if they had the source code - and of course they don't. But I would challenge Apple's engineers, first of all, they're not going to be able to learn the source code of every app, how many gazillion there are that they're bragging about on the store. But, I mean, as I have often said, you can be staring at code, and you inherently buy into what the code says it's doing. And your eyes would scan right across something that someone malicious had cleverly had the code do that you could just never detect. You would not see it. So that would create a whole new cat-and-mouse game of "Apple's making me give them my source code. Well, I'm going to put something in it just to show them that I can." Which we're not going to have to go to because Apple doesn't get the source.

But so, yes, Leo, I think your summary is exactly right. These phones have evolved into computers. And unfortunately, as we know, the thing that is the source of all of the vulnerabilities we talk about in the guise of full-blown PCs and Macs and UNIX and Linux machines is the connectivity. And phones are inherently connected. That's what they're for is their connectivity.

Leo: Which makes them more desirable, frankly, to hackers. I mean, I think ultimately this is going to be the frontline of security is these phones. There's ways for them to make tons of money just by commandeering the phone.

Steve: Yes. And there's a cornucopia of little apps that you can download, that people are downloading all the time, that do things. And the problem, of course, is that people are also using their phones for storing personal, confidential data.

Leo: Right.

Steve: Text message dialogues and their address books and so forth. And there have been high-profile instances where celebrities got their data sucked off their phone for reasons of, like, a dumb password. Paris Hilton I think famously had that happen to her because...

Leo: She had a dumb secret question that was her dog's name, which everybody knows.

Steve: Exactly. So the problem is these things become repositories where you assume confidentiality that you really can't assume on a connected device. So the lesson is security-conscious people really need to exercise extreme self-control with the apps that

they run. We would like to believe that the sandboxing, which Apple and Android both have deliberately engineered, is going to work. But we're seeing instances where it just doesn't, where mistakes made in the code break out of the sandbox, much as this jailbreakme problem with the PDF rendering elevates Safari to full root access, and then it can do whatever it wants to.

Leo: It raises a really interesting question because I think people have said, oh, well, the Google store isn't as safe, the Android store isn't as safe because it's not vetted. And people have proposed that maybe there should be a vetted and unvetted store. Maybe Google should have Google-approved applications, or the carrier. Actually Verizon does that, Verizon-approved applications. But you raise that point that you cannot ever be 100 percent sure unless you demand source code. And even then it's tough.

Steve: Yes. I would say that there's no better example of what would end up being a false sense of security.

Leo: Yes.

Steve: Coming from anyone saying, okay, we're going to put our stamp of safety. It's like, well, are you going to guarantee? Well, of course not. They would never do that.

Leo: They can't.

Steve: They can't.

Leo: What about antivirus software, that kind of thing? Is that the next thing to do? I mean, there is antivirus software in Android. There's Lockout, something like that, that I've used, that looks like it scans every download. I don't know what it's scanning for, but I don't think that would solve it.

Steve: I don't think so.

Leo: It needs heuristics, wouldn't it. It would need to monitor what's going on.

Steve: Yeah, and we've got - two more bullet points ahead of us is an interesting concept that was released at Black Hat that we're going to spend some time talking about.

Leo: I won't slow you down. Go ahead.

Steve: Sort of like that. In the news, the U.K.'s Information Commissioner's Office, the ICO, formally concluded that Google "did not collect meaningful personal details."

Leo: Thank you.

Steve: So, yes, exactly. So they have said they looked at what Google collected, and they were one of the first people to say we want to understand what was going on. They do. And they've essentially let Google off the hook. They did say they were going to keep an eye on what other countries' investigations uncover. But their preliminary feeling is they were collecting it by mistake, they didn't intend to use it, they didn't use it, and they're sorry. So, I mean, and I think that's all true.

Unfortunately, while the Information Commissioner's Office seems to have a clue, the U.K. government itself seems not to. Well, or they're stuck between a rock and a hard place. They've formally said that they're going to continue using IE6, against mounting pressure to get with a better browser, even 7 or 8 under IE, or maybe Firefox. The bad news is they made the mistake many years ago of commissioning the creation of a large body of custom, government-driving software which only runs on IE6.

Leo: [Laughing] Sorry.

Steve: It is locked to IE6...

Leo: Oh, dear.

Steve: ...platform, and it will not run anywhere else.

Leo: Oh, why?

Steve: So they're saying they cannot leave IE6 without incurring a huge cost. It's like, well...

Leo: Yeah, I'll tell you a huge cost. You want to see huge costs? I'll show you huge costs.

Steve: [Indiscernible] Microsoft stop supporting that. And then good luck to you.

Leo: Well, I see this all the time, especially in line-of-business software that just requires IE. And maybe it's ActiveX. It probably is. It's probably that they require ActiveX; right?

Steve: Yup.

Leo: But it's just, oh. But that should be a red flag for anybody who's buying or

using software. I think.

Steve: Indeed.

Leo: I'm sure you would agree.

Steve: Also, speaking of Dubai and places in the region, the UAE has stated that - they've even given a date. As of October 11th they are going to shut down BlackBerry service within the UAE because they're unable to determine what people are texting and sending back and forth to each other. BlackBerry's crypto is state of the art. It's a properly designed public key crypto system where individual BlackBerry phones have public keys that they use. Each phone contains a certificate that it uses to negotiate a secure connection to BlackBerry's servers, wherever they're located, in Canada presumably. And that establishes a tunnel whose cryptography, whose encryption cannot be broken, as far as anyone knows.

So this has been a problem. As you were saying before, India a few years ago brought up the issue. They were uncomfortable with what RIM was doing. And BlackBerry, the RIM folks said, well, we're not going to drop our encryption. They sent some emissaries over to India to negotiate, and no one's really sure what happened except that BlackBerry, RIM was still able to continue service. So other countries in the region are similarly concerned that these phones are going to somewhere outside of their control, and who knows what's happening? I mean, I don't know what RIM is doing. All we do know is that the technology is so safe that our own government does allow BlackBerries to be used in sensitive situations.

Leo: Everywhere in government. Everybody in government uses BlackBerries. At least the last time I was in DC. They all use BlackBerries. This is when we - this was a few years ago, five years ago, when we interviewed Michael Powell, who was the chairman of the FCC at the time. He lived on his BlackBerry. Now, if the chairman of the FCC feels it's secure, I think it's probably secure.

The European Union Commission has just announced that they're not going to use BlackBerries. The quote is they've evaluated - and it's not just for security. They say for durability and running costs, as well. But you've got to think security is the primary concern. "Following this evaluation, the HTC and the iPhones emerged as the most suitable platforms for voice/mail-centric mobile devices. As a result, the Commission currently supports these two platforms," not BlackBerry.

Steve: Well, and of course those are generic email clients where you configure them to connect to whatever server, wherever.

Leo: So they could run their own, I guess, and not have to worry about going through Canada, the RIM servers in Canada and so forth.

Steve: Yup, exactly. And so that means that those email servers can then be monitored and watched...

Leo: I think that's what's - yeah. That's really what's going on. It's not that they're insecure, it's that they couldn't monitor them. They couldn't watch.

Steve: Well, if you have, for example, you have two BlackBerry phones in the UAE, and they are texting to each other, that sets up two very secure encrypted connections back to RIM. And it's only there that the text is decrypted and then reencrypted and sent back out to the other phone. So nobody anywhere in between, except at RIM, is able to see what's going on. So not only can no other extra government forces monitor the channel, but it is the case that back there at BlackBerry those communications are briefly decrypted as they're being reencrypted and sent to the other phone. So there's a vulnerability there from a state secrets standpoint. It must be, I guess it's possible for corporations and no doubt the government to run its own servers. I don't know what the architecture is from that standpoint. But...

Leo: Yeah, they have these BIS servers that they could run.

Steve: Yes. And I don't know whether that still runs through BlackBerry or how the security architecture works because you would think that governments could do that, I mean, like the UAE could do that, except obviously that's not an option for...

Leo: They say, and this is why the UAE's banning it, they have their own state-run telecom - it's Telesat, I think is what it's called, or eTelesat. And they say that, because it still has to run through Canada, they can't - they won't support it. October 11th they're going to cut off email Internet access. There's half a million users in the UAE. When I was in Dubai, everybody had BlackBerries. But they love iPhones.

Steve: Interesting. Also at Black Hat, an interesting - and this is what I wanted to talk about - an interesting concept called "Blitzableiter," which is German for lightning rod.

Leo: Love it.

Steve: Presumably, I guess it was named because what it does is it's an interesting approach for making Flash secure, the idea that it turns lightning into just a flash or something, so lightning rod. And it's an interesting concept. And from a security standpoint, I really like it. Which is why I wanted to give it a little bit of time and talk about it. The concept is that it's sort of an intermediary which reads a Flash file, a Flash movie, as they're still called, and parses the file into essentially a meta language, into sort of its own intermediate representation, and then builds that back into a Flash file. And this is something, this is a technique that has been around for years. It's one way, for example, in the case of Internet packets, where there's a concern that network packets could in some way be malicious.

The idea is you never let a packet cross from an untrusted area into a trusted area. Instead you have something in between which interprets the packet, basically breaking it down into an intermediate language, into some description of what this packet is supposed to do. You then discard that packet completely; and, using the description only, you build a new packet that does the same thing. And the beauty of that is things you

don't know about, mistakes that are being made, for example, deliberately created in the original packet, they get flushed away by this reinterpretation.

First of all, if the interpreter looks at the packet and can't understand something, well, it's probably been malformed. So it ought to just be dropped. It's like a bad packet. But if everything seems to be okay, the act of - it's as if someone who couldn't lie was being used as a proxy and received some information and then turned around and told it to someone else. Well, if that person can't lie and has knowledge of the truth, then they're a filter. They're going to prevent a lie from passing through them. Similarly, if this interpreter is designed to build benign packets from a description of what an incoming packet does, well, it's going to prevent any sort of badness from getting through.

Well, that's what these guys have done with Flash. It's GPLed. It's on Google, in Google's code base is this - this work is being developed. They don't have a complete interpretation of Shockwave Flash at this point. They've got a large body of it done. But it literally discards the original file. So if you want to - it's available as an installation. At this point I'm not recommending people use it. I think it's too immature. But it works with, it integrates with NoScript running on Firefox.

Leo: Really. Oh, that's neat.

Steve: And so the idea is that, if you ran some Flash, that Flash never hits the actual Adobe interpreter. This thing gets it first and breaks it down into what the Flash is supposed to do, and essentially understands what it's supposed to do, discards that file, the original file completely, and then recompiles a new Flash file from that intermediate description, which is then what the Flash interpreter runs. And so you are, in the same way that I described with Internet packets, you're protected because somebody in between said, okay, this is what this is supposed to do. And this rebuilder essentially, well, you're not using the original file. So tricks like buffer overruns and things that are depending upon particular characteristics of the interpreter to be breakable, end up not making it through sort of this purifying process. So it's an interesting notion. And I'll bet we see things like this in the future because it's a very powerful concept.

Leo: Very cool.

Steve: Somewhere, I think it was on one of the TWiT shows that I was listening to, I heard some guest ranting about - might have been Paul, actually - about reflection from the iPad. In fact, I think it was Paul. I think Paul Thurrott was saying he's on a plane, and the only thing he can ever see in his iPad is his own face.

Leo: Yeah. It's glass. It's a very reflective surface. As you can see, I'm reflecting our lights right back at you. It's very reflective.

Steve: And so I just wanted to make another pitch for this iLuv anti-glare film. I have it on both of my iPads. Every iPad owner who sees it says, oh my god, where did you get that iPad? I say, no, no, it's the same iPad. If I peel this back, you'll see yourself looking at yourself. But, I mean, it is a pain to get on because it's one thing to, like, put an anti-glare film on something the size of a phone.

Leo: Right.

Steve: It's much more difficult, I mean, it is really difficult. And the stuff's not inexpensive, and you'll probably go through some getting it on right. The iLuv anti-glare, you get two per package.

Leo: Oh, it's not - so it's not a sheet that goes over it?

Steve: Oh, yeah, it is.

Leo: Oh, it is. Okay, just but getting it right is hard to do.

Steve: Well, first of all, you've got to get the surface clean. Then the problem is peeling the backing off of the anti-glare film, inherently, you're, like, peeling it apart. That generates static electricity. So then every bit of dust in the neighborhood comes rushing to it. I mean, it really is a pain. And then you've got this big sheet which you have to get exactly aligned correctly. I realize I'm not selling this very well, but I'm wanting to caution people. I mean, it is such a mixed blessing. But the upside is, if you can get it done, oh, it's unbelievably good.

I mean, it's the biggest mistake Apple made, I think, is this high-gloss mirror-finish glass on the iPad. I know that's what Apple likes. Yes, it's sharper and crisper and more wonderful. But boy, it's annoying. When I look at somebody else's iPad - oh, and also, of course, everybody else's iPad just looks like they're finger-painting with their body grease all over it. And so the anti-glare really does a good job of hiding fingerprints, as well. I just can't recommend it highly enough. I wanted just to get it out into the ether after listening to Paul ranting about how tired he is about, I mean, he was really ranting about it. I thought, okay, I've just got to say this anti-glare film really does work, and it is really wonderful once it's in place. And then you never have to worry about it again. Getting it down right is a real pain, but it's possible. I've done it several times.

Leo: i-L-u-v; right?

Steve: And Amazon sells it. i-Luv.com sells it. I think it's i-Luv.com. But Amazon also sells it. And it's less than 20 bucks, I think, for two sheets. And you'll need two because the first one will be a learning experience. But it just - it does work. I'm here to tell you.

Leo: Good to know, yeah. I'm ordering it right now.

Steve: Good. I think it's worth trying, Leo. It really makes a difference. I had a fun story, just because the guy's a little bit over the top. Greg Scheeler says, "Well, I can't believe it. I'm now one of the masses contacting you to tell you what a great product you have," speaking in this case of SpinRite. He said, "Here's the story. I had an upset co-worker. Her home PC would no longer boot Windows XP after a power outage. I offered to help. After identifying the point of failure during boot-up, I realized that something

was wrong with the hard drive. I thought about moving the drive to another PC so that I could check it out, but I didn't have a PC available with the correct connections. And, frankly, I was trying to avoid spending $89 for your product."

Leo: I don't believe that.

Steve: "I finally gave in. There was nothing else to do. I purchased SpinRite 6, not really convinced that it would make any difference. I thought I had just wasted $89. I let SpinRite run overnight. In the morning I rebooted the PC, and it came right up into Windows. I'm a convert. I realized that I just can't be without this tool, in my side business which is PC repair/web design, in my toolbox. I'm now saving up to get the site license. Great job, Steve. Greg."

Leo: Very nice.

Steve: And Greg, thank you very much.

Leo: All right, Steve. DNS rebinding. What's the story here?

Steve: Okay. So the problem has been understood for quite a while. We need to step back a little bit and talk about what's called "same-origin policy," which is sort of a fancy word for "same-site policy." So you can think of same-site policy as what this is really about.

The guys who were doing Netscape Navigator 2.0, who put JavaScript into web browsers for the first time, they realized that scripting was very powerful. They got that part right. And that essentially, when you went to a website and downloaded a page which contained JavaScript, this JavaScript was going to run in the browser, and it could do lots of things. What they wanted to prevent was it doing anything to other websites on behalf of the user. So because, for example, you could query other objects. You could, I mean, the scripting was - it's a language that is very powerful and flexible.

So they said, okay, how do we constrain the script so that it's not going to get up to any other mischief? And they said, well, let's let the script only deal with the same site, that is, the site that it came from is the only server domain name that it's able to access. And so this notion of same-origin policy, that is, the origin where the script originated, the origin of the script is a constraint that all browsers since then have imposed. Some of them do it to different degrees. And different resources have different degrees, sort of like levels of enforcement.

For example, origin is supposed to mean the same domain and port and protocol. So, for example, if you got a document over https://amazon.com, then the script could not do anything to http:// because that's a different protocol, HTTPS versus HTTP. So it's got to be the same protocol. Also the same port, although it turns out IE doesn't enforce the port side. And, for example, cookies don't obey the protocol side. So cookies that you transact over HTTP will also be transacted over HTTPS as long as the domain is the same.

So these things are understood. And this has been sort of evolving along for some time.

The problem is that DNS creates a relatively weak link or, to use the fancy term, "binding," a weak binding between the domain name and the IP. That is, that's what DNS does is it binds a domain name to an IP so that, as we know, you ask DNS on the Internet, wherever it is, what's the IP for this domain name, and some sort of a process goes about resolving that domain name into the IP, and you get an answer.

So it's been understood, though, that there are some problems created with this. And this has also been known since about 14 years. It was in 1996 that the first DNS rebinding attack was first seen. And it was used against the Java Virtual Machine. The idea there was that, and we discussed this briefly a couple weeks ago, when you ask for the address, the IP address of a DNS domain, you can receive more than one IP address in return. This is often used for load balancing. For example, I think if you ask - I'm trying to think, I think like the IP for Amazon, you don't get one, you get, like, three or four, or Microsoft. And every time you ask, the server rotates them so that the one that's sort of first in line is what the browser will use. But the point is, if there's a problem, if that server's overloaded, or it can't make a connection, it'll go to the next one.

Well, what some clever hackers realized was that they could return the actual IP address for a malicious site as the first address, and a local IP like 127.0.0.1, which is the localhost IP, for the second address. So when the browser attempted to get another resource from the malicious server, it would send a reset packet. It would send a TCP reset packet back to the browser, denying that connection on the IP that it wanted to use. Well, since the browser had received a number of IPs, or at least two, it would go to the second one, which in this case was 127.0.0.1, which is sort of universal. It's called the localhost IP. It's always used to refer to that own machine.

And what that did was that gave Java, the Java Virtual Machine, something by design it should never have, which is socket-level, network-level access to your own machine. Which hackers had all kinds of fun with until it was understood that this was causing a real problem. So what happened that got this into the news just recently is that another new vulnerability was discovered in routers that hadn't been suspected before, which was, okay, now get this. The router will obey connections aimed at its WAN IP from the LAN.

Leo: So the WAN IP is the IP assigned by your ISP.

Steve: The public one.

Leo: The public one.

Steve: Exactly. So it's like, whatever, 24.192.345.789. I mean...

Leo: And the LAN IPs, those are the private ones, the 198 or the 10-dot.

Steve: Right.

Leo: Right.

Steve: So the idea is that, normally when you connect to your browser, like you want to do administration on your browser, you typically use its gateway address, 198 or 192.168.0.1 or 1.1 or whatever, which is typically the same IP that it uses for its gateway. It has a web server running in it. And so you point your browser at that IP, and that brings up its admin interface. Well, because rebinding is a problem, there have been protections put in place historically against - by browsers against being fooled by having the script able to use local IPs to access your browser.

So let's step back a little bit and understand how that works first because the idea would be you browse to a malicious site. You don't need to press any buttons, click any links, do anything. You just download a page from the site. Or what's even more disturbing, a web ad is served by a malicious site. So you don't even have to go to a site. You can simply be surfing around benignly on the Internet, and a web ad is displayed which is, after all, a browser document. And we know that those can contain scripts because they often are Flash, which has got some script running to show you a Flash ad. It can also be JavaScript.

So the idea is, when your browser asked for the IP address of attacker.com, it received a valid IP address the first time it asked. Then, in running the script, the script says, oh, I need something else from attacker.com. So what happens is your computer makes another request for the IP address of attacker.com. The reason it does that is that your browser has its own DNS cache, but plug-ins like Flash have their own. So even though your browser knew the IP address of attacker.com, Flash, the Flash plug-in, technically the term is they have separate DNS name spaces. So the Flash plug-in, or Java, or Silverlight or whatever, they're not privy to, for example, Firefox's DNS cache or even your system's DNS cache. They've got their own. So they'll make a request.

When that second request is made, instead of returning the IP address of the site, it returns an IP address that is probably your router's gateway, like 192.168.0.1. So now what happens is this same-origin policy we were talking about, which prevents a script from having access to different domains, now what it has is it says it asked for attacker.com, which it's just been told is 192.168.0.1. But attacker.com is where the script came from because that's where the browser originally loaded it from. Which means that the script came from attacker.com. Now this Flash plug-in believes that attacker.com is your gateway, is your router.

Leo: Ah.

Steve: Which means it has full permission within the same-origin policy to do anything it wants. And so it's able to establish a web browser session, a web connection to your router, login without you knowing it, assuming that you didn't change your username and password. It can typically identify the brand, make, and model of your router from the greeting page, the login page that tells it what kind of router you have, make and model. It then looks up in its own little dictionary the default username and password. And more often than not, about half the time apparently, it's able to log on. And so that's the way same-origin policy is broken.

Now, some browsers and plug-ins protect against this because this, again, has been known for a long time. They block 192.168 dot anything dot anything. They block 10-dot anything anything anything. And the 172.16 through 172 dot what is it, 24? 29? I can't remember what the second byte of that is. But basically the RFC 1918 is where those define. Those three networks, they're smart enough not to allow that. So this problem was believed to have gone away.

It turns out it crept back in, in a different form. And that's what this hacker revealed last weekend at Black Hat, which is, not only can you browse to your router's web browser using the private gateway IP, 192.168 dot whatever dot whatever, or whatever it is; you can, believe it or not, also get there using its public IP, that is, the WAN IP, the public IP of the browser, even if it has been disabled, even if you've specifically configured your router not to allow WAN-side access. The way the stacks are written in the DD-WRT and OpenWrt browsers, these aftermarket firmwares, and some of the standard manufacture firmware, will still allow the browser to respond from inside the network, if you use the IP from outside the network.

And so the next-generation attack that was revealed last week, which I'm sure all of the various firmwares are in the process of scrambling around to fix right now, solves, well, what it does is it gets around the blocks against internal LAN access IPs by using your public IP. And of course the remote DNS server gets your public IP because that's the IP from which the request comes to it. It's emitted by your computer, asking for the IP address of attacker.com. Well, that comes from your public IP. So it's able to return the public IP to the script running in a plug-in, which then knows how to get around the use of private IPs on the LAN to access your router.

So, I mean, this is the kind of complexity we're dealing with in this day and age because we've made our systems so complicated. It's, I mean, it's just like one more little hole that's been found that we now have to scramble around and patch.

Now, I mentioned a couple weeks ago that NoScript had some built-in protection for this, that is, for prior types of attacks like this. They're very, very close to releasing v2.0 of NoScript. I think we're at 1.9.9.96. We're just about to roll over to v2.0. And 2.0, I was looking at the release candidate log at RC7 and then 8. They've just added a new feature which will block this next type of attack, as well.

There's something very cool that I never really looked at in NoScript called ABE, stands for Application Boundaries Enforcer. And if you go into NoScript Options and then Advanced, under the Advanced tab there's an ABE subtab. And there's actually a little rule-based firewall that there's a page of documentation about it. On the 'Net you can go to noscript.net/abe and learn about this. But this is built into NoScript, and it allows some interesting restrictions to be put on what your browser is able to do. They're not normally enforcing very many rules. There's a default rule that prevents this kind of problem, the first-generation type of rebinding attack that your browser would launch against any other machines in your local network.

And by the way, this doesn't only have to be launched against your router. Essentially, what these rebinding attacks allow is your computer to serve as a proxy that's then operating inside your network and, with script running in it, that potentially has access to any of the machines in your network. Even, for example, to Windows filesharing, where behind your router you might believe that filesharing is safe because your router is going to protect you. But if something has set up a beachhead in your browser which then has access to your network, and this is specifically what same-origin policy is designed to prohibit, if that's broken, then you've got something running in your browser that has visibility into your entire LAN. Which is frankly terrifying.

Leo: Yeah, no kidding. They should call it "Honest ABE."

Steve: [Laughing] That would be good.

Leo: Keeps you honest. Or keeps a browser honest.

Steve: So we have essentially this DNS problem which DNSSEC doesn't protect us from. It takes advantage of the fact that our systems have gotten complex, so that they're sort of all doing their own DNS fetches and not taking advantage of the DNS knowledge that different pieces of the system have, which allows the same domain to be known by multiple IPs. And if some of those IPs are within your network, then scripts which have been deliberately restricted against having access to anything but the domain they came from, well, they then believe that the domain they came from is whatever machine inside your network they want access to. This makes it possible. And Flash and Java both have socket-level capabilities, meaning they can open connections, low-level network connections. They're not constrained to just web-based accesses. They can open network-level connections to, for example, email servers within your network, and then use your email server to send spam out, or do whatever it is they want to. And they can have a persistent connection to a remote attacker who's got now persistent access to your machine, essentially using your machine like a proxy into your network. So, frightening stuff.

Leo: It sounds like another reason to use NoScript.

Steve: Yes. We keep coming back to, remember that all of this depends upon scripting. In some cases it depends upon third-party plug-ins, like Silverlight or Flash or Java. But in general I would say the pervasive takeaway is "trust as little as possible," that is, NoScript ought to be there. You ought to not have scripting on unless you know you need it, and then turn it on selectively. As we said at the top of the show, it is really too bad that this kind of real, I mean, you could call it "kneejerk paranoia" is necessary. But here's another example of something that really does work, that was demonstrated, that no doubt people are trying to exploit right now until people get their browser firmwares updated. And by the way, if you are using DD-WRT and OpenWrt, it's very likely that, unless you've updated your firmware recently, you have exposure to this. So keep an eye out for updated firmware that specifically corrects against these latest DNS rebinding attacks.

Leo: And I would guess that DD-WRT variants, like the Tomato, I think Tomato is based on DD-WRT. It probably would also have the same issues. So if you've rewritten your browser firmware, you can check there, too. And they're telling me in the chatroom NoScript 2 is out.

Steve: Oh, no kidding. Because I looked just yesterday, and it wasn't. Oh, yay.

Leo: Well, they said - who knows. They say it is.

Steve: Great. I believe them. The chatroom...

Leo: They're rarely wrong. Although when they are, they're really wrong.

Steve: In that case, everybody, NoScript 2.0 is out. If you're a NoScript user, I would consider jumping to it. It does something really interesting. You might wonder how it could block an IP that it doesn't know about, that is, it has to block access to the WAN-side IP. What NoScript 2.0 does, with the permission of the person running the site, there's a site called secure.informaction.com, i-n-f-o-r-m-a-c-t-i-o-n dotcom. In fact, Leo, if you're in a browser right now, you go https://secure.informaction.com...

Leo: Informaction.com.

Steve: Informaction, slash ipecho. What you will get is your public IP.

Leo: Let's see if it works. Yes.

Steve: Now, NoScript 2.0 does that. Silently, when it runs, it makes a connection over SSL, using that URL. That allows it to know the WAN-side IP of a private LAN, which then with updated rules - it's updated its ABE rule set to include this additional information so that - and I think there's a checkbox. I was told there would be a checkbox, WAN IP "?" local checkbox, which hopefully is enabled by default. I'll know certainly by next week because I'm going to jump and go get NoScript 2.0. And that provides protection immediately against this, no matter whether your browser is vulnerable or not, as long as you're surfing from a NoScript-enabled system.

And in fact there was some interesting conversation that I saw in their forum where they were talking about how NoScript has been incrementally adding so many good features, like the clickjacking prevention that we talked about a while ago, and now these features, that frankly, even if you told NoScript to globally allow scripting, which of course goes against the "only allow scripting for sites you trust" rule. But the point is, it's doing so many other things to protect you that it's still very useful.

Leo: Yeah. Oh, yeah, absolutely.

Steve: Then it's not a problem for people. It's like, you know...

Leo: That's kind of how I use it. I say go ahead, do anything you want. But it's still catching these, clickjacking and now this DNS rebinding, stuff like that.

Steve: And remember tabnabbing.

Leo: Tabnabbing, clickjacking, and DNS rebinding. What a menagerie we have.

Steve: Indeed.

Leo: Steve, always a pleasure. Steve's the greatest. Don't forget GRC.com is his

website, where you can find the fantastic SpinRite, the world's finest hard drive maintenance utility - you must have it - and of course all the free stuff, like ShieldsUP! and all his free programs. He's also on Twitter now: SGgrc is his main chat-with-Steve thing; SGpad for pad updates; and GibsonResearch is the official Twitter account. But you know you can find all that at GRC.com. Next week Q&A, so leave your questions at GRC.com/feedback. While you're there you can get 16KB versions of the show. I see you gesticulating. What is it?

Steve: 261, baby. We've confirmed it. That's the last episode of year five.

Leo: It's been the last episode of year five for a while now. Finally, the end of year five. I've been waiting for this. 16KB versions of the show, transcripts, everything you need at GRC.com. Or TWiT.tv/sn, that's the canonical

web page where you can subscribe to the iTunes feed. We do audio and video now, high-and low-quality video, depending on your device. It's all there. It's all there.

Steve: It's all high quality, Leo.

Leo: It's all high-quality stuff.

Steve: Only a high-quality podcast.

Leo: Nothing but the best content from Steve. Steve, always a pleasure. Thank you so much. And we will see you next week. Now, I won't be - no.

Steve: No, you will.

Leo: I will. We just...

Steve: You're my hero, Leo. You're my hero. You're always leaving just after the podcast gets recorded.

Leo: I am. I'm leaving tonight and getting back Tuesday night. So...

Steve: I really appreciate it.

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

Steve: Thanks, Leo.