DNS - After the Patch

Description: Steve and Leo follow-up on the recent industry-wide events surrounding the discovery, partial repair, and disclosure of the serious (and still somewhat present) "spoofability flaw" in the Internet's DNS protocol. They also examine what more can be done to make DNS less spoofable.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-157.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-157-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 157 for August 14, 2008: DNS Problems Continued. 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!. Steve Gibson is here. He's the security guru. And in times like these it's good to have a friend like Steve. Hello, Steve.

Steve Gibson: Hey, Leo.

Leo: I mean, times like these, I'm not kidding, it seems like the bad guys are winning.

Steve: We have no problem coming up with weekly content, let's put it that way.

Leo: Oh, man.

Steve: There's lots of material. I deliberately didn't tell you something when we were doing our pre-start-of-roll chat.

Leo: You holding out on me again?

Steve: Again?

Leo: What's up?

Steve: First time I've ever held out on you, Leo.

Leo: You know what, it's funny because right before we do this show I do a podcast with Roz Savage. As you know, she's rowing across the Pacific. And she's been holding out on me for six weeks. We knew her electric water maker had broken, but it was all going to be okay because she had a manual water maker. She told me today, well, yeah, actually that broke six weeks ago, too. And it was like, what? She said, well, I didn't want anybody to freak out that I was going to die of thirst. I said, but you're going to die of - she said, no, we're almost...

Steve: What's she been drinking?

Leo: Well, when she rowed the Atlantic, it was a race. And the rules of the race required that you use ballast that was drinking water, potable water. So fortunately she...

Steve: You mean, like, safety rules.

Leo: Safety rules, basically. So they're, you know, plastic, kind of like bota bags. And...

Steve: Clever.

Leo: Yes. And so fortunately she had adhered to that rule, even though she's not racing this time, and had that, and she's been drinking it. But she said it was going to get a little dicey towards the end there.

Steve: Well, and a little stale-tasting, too. I think, you know, drinking your ballast doesn't really sound very appealing.

Leo: No, it doesn't sound - she said, yeah, it tastes a little rubbery. It doesn't sound good. I mean, it's clean, but it doesn't sound so good. Anyway, what's your big surprise?

Steve: My big surprise is just to note that being Episode 157, and being that there's 52

episodes per year, and three times 52 is 156...

Leo: Wow.

Steve: ...this is the beginning of our fourth year.

Leo: Holy moly. Happy anniversary.

Steve: Yay [trumpeting].

Leo: And you have not missed - am I correct to say that you have not missed a week in three years?

Steve: Never one. Never have we missed one.

Leo: He's the Cal Ripken, Jr., of podcasting, ladies and gentlemen. Unbelievable. Wow, that's really great. Our fourth year.

Steve: Yup, we're into year number four.

Leo: I can't believe that. That's really kind of amazing. I think I proposed this to you when we were in Toronto. You were visiting Call For Help as a guest.

Steve: Oh, I remember the moment. I remember where I was standing when - you were over on the set. Sort of if you think of the three locations in sort of a triangle, where you and Amber would do the opening, and then there was the place you went to do the...

Leo: Home base place, right.

Steve: ...Call For Help stuff, yeah.

Leo: And then Andy's set was the other set, point of the triangle, yeah, yeah.

Steve: Exactly. And so you were at the corner of that triangle of three. And you said, "What would you think about doing a security podcast?" And I remember, well, first of all I said, "A what cast?"

Leo: Yeah, because at that time I was only doing TWiT.

Steve: Yeah. And I remember thinking, like, later, it's like, oh, I hope he doesn't bring that up again, you know. Because, I mean, I just had no concept of what it was or what - and I remember thinking we're going to run out of stuff to talk about. I've got maybe three weeks' worth. You know. But here we are in year four and going strong.

Leo: Oh ye of little faith.

Steve: And with a long list of stuff to talk about still. Actually the list is growing in the same way that, like, SpinRite testimonials, there are more coming in than I'm able to read, like on a weekly basis. So it's like, okay, we're not running out anytime soon.

Leo: Well, thank goodness. Well, maybe not thank goodness. But the hackers seem to be keeping ahead of us.

Steve: They are. And frankly, Leo, I mean, this has been - it's been great for SpinRite for me to have a forum where I could tell people about it. I mean, I even notice, as I mentioned to you before, you're using it now as a verb, which really says, I mean, and this has been my lifeblood for the last 20 years, so it's important to me. So I'm more than happy to trade everything I know about security for helping to have our listeners understand that there is salvation when their hard drive dies.

Leo: Well, bless you, Steve Gibson. You know, I ran SpinRite actually last night because I've been having these kind of flaky results on the recording machine.

Steve: Really.

Leo: Yeah. But - and this is not the first time I've run it. Remember I mentioned this before because it's a RAID 0, and it's been giving me these weird error messages. And it bluescreened on me in the middle of TWiT on Sunday. So I ran it all day Monday, nine hours. Didn't find any errors, though. I ran the number four; right?

Steve: And you ran it on the drives separately?

Leo: No.

Steve: Oh, see, you really - I think you have to pull it out from behind its RAID configuration. SpinRite will allow you to do that. It doesn't modify the drive contents at all.

Leo: Okay.

Steve: And I would run it on each drive separately because it could be that the controller is masking the errors from SpinRite.

Leo: That must be it.

Steve: And it's definitely the case that SpinRite's not really talking to the drive controller itself, it's talking to the RAID controller. So there's a bunch of magic that SpinRite does in terms of, like, at the low level of the ATA interface, where like it turns off retries, it turns off ECC, it's able to penetrate and get much, I mean, much more intimate relationship with the drive.

Leo: That's what happened. Okay, good, okay. Because, yeah, what I get when I boot up, the RAID controller says there's an error on the first drive. But I see no errors in SpinRite. Second time I've run it. That's what I'll do. So I'll SpinRite the drive individually. Okay. I'll do that. That's tonight I'll be doing that. I'll let you know how that works out.

Steve: Cool.

Leo: Yeah. See, it's nice to have friends in high places.

Steve: Well, at least knowledgeable friends.

Leo: I did want to ask you, actually, and I should probably ask you this off the show, but I'm going to do it anyway and put you on the spot. You know we're making the ultimate gaming machine. It's one of the things we're giving away, really fun, we've been having such a blast doing it. Have you been following this?

Steve: I've heard some of it, yes.

Leo: Colleen is building this amazing box.

Steve: She's a kick.

Leo: She is. I mean, you know, when you get somebody, a girl who can weld and build, and then she, you know, she's never built a water-cooled system, but she of course understands fluid dynamics very well. And so she's done all the research, and she's come up with a really great design. She had her own skateboard design company. I mean, she kind of knows how this stuff works. I mean, she did the 3D rendering of these aircraft aluminum skateboards. Anyway, but what I would like to ask you is if we could - if I could get you to donate a copy of SpinRite to this.

Steve: Oh, in a heartbeat, of course, of course.

Leo: Because you know why we really need it, we're using VelociRaptors, those new 10,000 rpm drives. They're the fastest drives out there.

Steve: I would just give each of those drives to SpinRite before you even start your install and setup.

Leo: Great idea.

Steve: There are a number of people who run SpinRite on drives just, I mean, even though it's new, they just run it so that SpinRite has a chance to just thrash it to death - hopefully not to death. But you'd rather it then than later.

Leo: Well, yeah, yeah. You know what, I should have done that with the TriCaster because we only had one drive. The new drive that I put in, I should have done that. I will. That's another thing.

Steve: Good preventative maintenance.

Leo: We're going to have a SpinRite party tonight, baby. I'm going to SpinRite everything in the house. But that's kind of the great thing. So thank you, that's a very nice offer. I'm going to try to get all of our sponsors to donate hardware and software for the winner because, I mean, they're already getting an $8,000 computer. But it's nice to, you know, I mean, they need SpinRite. They've got these - and it's going to be in RAID 0. They need it.

Steve: Yup.

Leo: They need it. All right, Steve. Let's take a little look at the tech news. But before we do, what are we going to talk about today? We do have a topic today; right?

Steve: We've got a great topic, actually. We're going to basically talk about the "DNS - After the Patch" is what I would title this podcast. As we all - we've talked for several weeks about this major vulnerability that was discovered by Dan Kaminsky back in February, and secret meetings which were held in Microsoft's campus in March, which resulted in virtually all DNS servers that were on the Internet having the ability to be patched, that is, patches made available to mitigate the window of opportunity of DNS spoofing exploits. Well, there's been a lot of news since then because, well, in fact one Russian researcher has successfully exploited a fully patched, much more secure DNS server, BIND 9 with all the latest stuff, because it's harder to do; but the question is, is it yet hard enough to perform a DNS spoofing exploit, a cache-poisoning exploit, which we talked about in detail two weeks ago.

So we're going to talk about what has happened since and some interesting further steps that can be taken. I'm going to touch on the ultimate, like, potential solution, which is

something called DNSSEC, or DNS Security, which has been churning along for nine years and still hasn't gotten off the ground. I'm not going to go into it in detail because that's a topic for an entire episode all by itself. DNSSEC is very powerful, but it's got some real problems. And then finally we're going to end up with, okay, like, we've used the term "hack" before. What I'm going to describe is a way of further strengthening DNS that is like the ultimate hack. It's like the definition of the word "hack," which we're going to have a lot of fun with, so...

Leo: That sounds cool.

Steve: ...lots to talk about. Lots of security news, as well.

Leo: All right. Well, let's start with the news.

Steve: Oh, yes. Mac OS X got a major, an important update. So I just wanted to mention that to Mac users. When I turned mine on this morning it said, oh, we've got new stuff. We've got an iTunes update and a big security deal. And it had to do the whole shutdown sort of secret repatch the OS level kind of thing. So it was a big deal. So that happened. Also we're recording this Tuesday the 12th. This is the second Tuesday of the month, and this is a mega Patch Tuesday. So by the time our listeners hear this on Thursday this will be old news. But I wanted to make sure, as always, that they pay attention to this. This is a 12-bulletin major security update from Microsoft. Many critical vulnerabilities, remote code exploit, buffer overrun, rah rah rah, you know, standard routine. So don't ignore this one. This is a big one. I haven't seen them all because, you know, we're recording this on the day. I did try, I did ask Microsoft to give me an update try, and they said nothing yet. It's like, okay, fine. Well, later today it'll definitely be something. So I wanted to mention that.

Also, in other security news, we've talked about in various contexts the controversial actions that Comcast was taking, this so-called "deep packet inspection" where they were selectively dropping BitTorrent connections, which got them into trouble. The FCC voted three to two against Comcast's doing this. I thought it was interesting that there were two people who thought this was a good thing. It's like, okay, well, at least we got the majority. It was three to two. And what the FCC ruled was that Comcast violated federal policy by throttling Internet traffic for subscribers using BitTorrent filesharing software. Now, Comcast rebutted, saying that this FCC network neutrality stance is just a policy statement and not an enforceable rule. However, then the FCC went further and said that Comcast had a motive for its action. Comcast was just saying, well, we were trying to, like during periods of high traffic use we were just trying to, like, let everybody have a fair share. But the FCC said, wait a minute, you've got a motive because users downloading video files through peer-to-peer clients, specifically BitTorrent, could be perceived to be taking business away from Comcast's paid video-on-demand service.

Leo: Exactly.

Steve: So it's like...

Leo: It's anti-competitive. What they're doing is anti-competitive. They do the same thing in some, well, a Canadian ISP does it with Skype.

Steve: Right.

Leo: Totally anti-competitive. It's not about protecting the network.

Steve: And so fundamentally the people who have observed this have said that the problem was selective enforcement against BitTorrent specifically. That is, it wasn't that Comcast was, like, throttling back all high band uses of anything. They were targeting the BitTorrent protocol and deliberately blocking that while allowing other potential high bandwidth applications to just move through without any throttling at all. So anyway, so I guess the point is that ISPs have rights to enforce terms of service. But they can't, you know, the Network Neutrality guidelines say you can't selectively enforce bandwidth limitations. You've got to do it across the board.

Leo: Although, and while this is a victory, it wasn't really, I mean, it was - they didn't even punish them.

Steve: Right, exactly.

Leo: There was no penalty whatsoever. So...

Steve: It was just, you know, we're watching you and don't do that please.

Leo: Don't do it. And of course Comcast has said, but we never did.

Steve: Uh-huh.

Leo: We don't do that.

Steve: Now, in the other really big news that came out of Black Hat, of course we've been talking about, pre-Black Hat conference, we were talking about Kaminsky's presentation, which I'm going to go into in detail when we get into the heart of our show. But in other security news I wanted to let all of our listeners know that I'm aware of the presentation by two guys, there were two security researchers, one with ISS, IBM's Internet Security Systems, and the other from, interestingly enough, VMware. These guys have put together a very meaty paper. It's a 50-some-odd page, I think 54-page paper where they demonstrate how to bypass Vista's security, this much-heralded ASLR, the Address Space Layout Randomization that we've talked about before, and DEP, the Data Execution Prevention, which were like the big deals that Vista was adding to beef up security beyond XP. So two weeks from now we're going to - that'll be the topic two weeks from now, given that nothing else more horrible happens in the meantime. And

the title of that one will be "Vista Security Bypass?"

So I just wanted to let our listeners know I'm aware of this to prevent everyone from having to go to GRC.com/feedback and tell me, hey Steve, Vista security has been bypassed. We're going to look at that. There is some - there are two sides to this, and we're going to take a balanced position rather than going all crazy and saying the sky is falling, you know, how serious is this, how does it work, what does it mean, and is this really a problem sort of deal. So we'll take a look at that in depth in two weeks.

Leo: It would be a shame. I mean, Vista has actually been much, much, much more secure than XP for the last 18 months. I'd be very sorry to hear that it wasn't as secure. So we'll look forward to that a couple weeks hence.

Steve: And that's a perfect, a perfect segue to my further saying that, yes, and it's one of the arguments against this being a big deal is that this isn't suddenly - this doesn't suddenly make Vista a lot less secure. Vista's more secure for many reasons, sort of a whole landscape of things that were done better. But the question is, how much less secure does this make it?

Leo: Right, right, right.

Steve: And, finally, I got a really nice note from a Security Now! listener. Neil Abrahams in the U.K. wrote about his experience, actually his whole experience with GRC, which no one has ever really mentioned explicitly before. He said, "Hi, Steve and team. Kudos at the speed, quality, and easy of use of SpinRite 6.0. By day I'm an IT consultant, so naturally I'm looked to by the family to sort out the problems. I got a call from my virtual father-in-law" - and I'm not quite sure what that is. But he's a computer guy, so he has a...

Leo: Maybe he's virtually married. I don't know.

Steve: He has a virtual father-in-law - "saying the family laptop could not boot up, and that they had tried the recovery options such as safe mode, last known good configuration and so forth, those boot-up time options in XP. On receiving the laptop I immediately removed the drive and inserted it into my USB drive caddy so I could access photos and documents that had not been backed up before moving on to fix the Windows installation. I ran into the first problem when my desktop would not recognize the drive. I also checked it on my laptop, and still no drive was recognized. By this point I assumed the drive was physically damaged and that I couldn't recover any data whatsoever. To check that theory I put the drive back into the laptop, the original laptop, and booted into its BIOS to check that the drive was being recognized, and sure enough it was. Having been a ShieldsUP! advocate and then becoming a Security Now! listener when it first began, I immediately thought that if the drive could be seen by the BIOS, then SpinRite could fix it." And of course that's proper logic.

So he says, "I logged onto GRC, purchased and downloaded my first copy of SpinRite 6.0, and installed it to my USB flash drive. I plugged the USB into the laptop and booted that into the SpinRite program. I chose the Level 2 option, and 20 minutes later it had finished. On reboot, Windows started up and I was able to log on and back up the data,

albeit very slowly. The service and quality of the website in terms of" - and he's talking about GRC's website - "in terms of simplicity, ease of payment, and download facilities is what makes GRC a world-class business. In less than five minutes I was running the SpinRite recovery and had the problem resolved in 20. Smiles all around from everyone. Cheers. Neil Abrahams in the U.K."

Leo: That's neat. It's not an easy thing to do, to make a useable website. And nowadays, I'll tell you, the pros get involved in this. And you did it all by yourself. I think that's great.

Steve: Yeah, you know, I wanted it to be mine, so...

Leo: I'm the same way. I'm the same way.

Steve: So NIH.

Leo: Yeah, we're very old-fashioned, you and I, in that regard. There were some other, you know, you could probably have done a whole show on DefCon. I mean, there were some other things. Did you see the Medeco lock hack? That was amazing. You know, these Medeco - they're, like, twice as expensive, these Medeco locks, because they're unhackable. And for some reason there's something about software hackers that they really like a little hardware hacking and lock picking as, you know, kind of a relaxation?

Steve: Right.

Leo: Did you ever do that? They have lock-pick contests and stuff?

Steve: Oh, I can pick a lock.

Leo: Yeah, see, I knew it.

Steve: I had my youth.

Leo: It's funny, I find it fascinating, it's very, very common among geeks that they did safecracking and lock picking, too. I don't know why that is. But anyway, these Medeco locks are - I'm sure these Medeco people hate DefCon. Hate it. So they showed how you could just get a low-res picture, you know, use your camera phone to take a picture of somebody using their key. You don't have to have a great picture of the key. And then they show how you can blow it up and use it to cut out a credit card and actually open the door with this credit card. I mean, it was kind of pathetic. I thought, boy, that's, I mean, this is what's interesting about technology has really advanced the abilities of hackers. And then there was another presentation which

the folks at the MTA must hate, how to hack the Boston subway cards with a cloning attack so you can ride for free forever. And of course...

Steve: It really is those kinds of things and the diversity of them are one of the things that make DefCon so fun is it's just...

Leo: It must be fun. Have you ever gone to that?

Steve: Nope, I never have. I have certainly read the reports that come from it, though.

Leo: You and I should - maybe next year we can sneak in.

Steve: That'd be cool.

Leo: We don't have to sneak in.

Steve: I don't think we have to, no.

Leo: No, we don't have to sneak in. Patrick Norton used to go all the time. He said, I'm not - I said, bring a laptop, and you can call us. He said, are you kidding, I'm not bringing a laptop to DefCon.

Steve: I remember him saying that, yes, yes, yes.

Leo: Yes. Those guys are crazy. I'm not getting online there.

Steve: Unh-unh.

Leo: So thank goodness that two weeks ago, Steve, you gave us an update on how DNS works because then - actually it was more than - that was four weeks ago. Because then we talked about this DNS flaw and how it worked and the presumable fix. Well, now the fix is broken.

Steve: Well, okay. Yes and no. We, as we really covered clearly and carefully two weeks ago, DNS was literally never designed to be secure. Like all the original protocols, it was designed just to work. And back then just the fact that it did work was a miracle. I mean, they couldn't believe that, like, this whole notion of autonomous packet routing was as robust as they designed it to be, where you just stick a packet on anywhere on the Internet with a unique destination IP and it just finds its way there kind of all by itself with the help of routers bouncing it from one router to the next, and it ends up at its destination. So DNS was so insecure - and I don't even like to use that word because it

was just never - security was not part of anyone's thinking back then. And I remember reinforcing that by saying, you know, SSL, the really cool layer on top of TCP that the Netscape guys added, it didn't exist in the beginning. There was no way to encrypt communications on the original Internet. That was an afterthought.

And similarly, DNS had no, virtually no spoofing protection because the servers at the time would open one outgoing port and would use it for emitting all of their queries, meaning that all of the responses would come back to the same port, and the 16-bit transaction ID was just a counter that counted up linearly to help the server determine which, when replies were coming back, to help them match up the replies with the queries. So the fact that it used a fixed outgoing port for its queries and that it used a linearly increasing counter meant that faking or spoofing replies was child's play. So because that was a hole that was just too big, one of the early improvements in DNS was simply to change that 16-bit counter into a 16-bit pseudorandom number. So that value would be jumping all over within a 16-bit space which we know is 64K, or 65536, in an unpredictable fashion, hopefully unpredictable. In fact, there were some pseudorandom number generators that were not sufficiently unpredictable. And so another attack was getting a few queries from a server and using those to sort of lock onto its pseudorandom number generator and again being able to spoof replies and poison the DNS cache and produce all kinds of mischief that we know follows from DNS cache poisoning.

So what happened in - what Dan's realization was - and I should mention his Black Hat talk was everything we expected it was. There was nothing that he revealed that hadn't already been independently reverse-engineered just from the knowledge that there was a problem, which I think is really interesting. All he had to say, as he did in early July, was there's a big problem, we're not going to tell you what it is, wait a month, and I'm going to let everybody know in Las Vegas. Well, a couple weeks later everyone had guessed. And just by virtue of the way the 'Net works, people improving on each other's guesses, they figured it out, built a proof-of-concept, released an exploit, and it was possible...

Leo: It's pretty amazing how the 'Net works that way.

Steve: It really is.

Leo: You can't, you know, people think it's kind of like the old days, you can keep a secret. You really can't.

Steve: No. And in fact, all it took was just people knowing there was something.

Leo: Right.

Steve: And so they thought, hmm, what could it be? And you think it could be this? And someone will say no, because of that. And okay, how about this? Oh, now that's interesting. How about maybe if you made it green, then it would work better. So, I mean, it just, you know, it just happened. And this thing was born with Dan saying nothing. Now, Dan's talk did evidence, however, that he had been thinking about this a lot longer. So, and as sort of the buildup to this, he pushed the limits of what it means

for DNS to be broken, as badly broken as he discovered it was in February. And, I mean, it really was badly broken. It's cool, when you think about it, that this - cool in a certain twisted sort of way - cool that something this bad hadn't been independently discovered until something was known to be wrong, and then it was found. Because, I mean, it turns out that when HD Moore added this exploit technology to his Metasploit framework, and making it very command-line run-able, you could basically poison an arbitrary domain on an arbitrary name server in 10 seconds.

Leo: Now, wait a minute, you just said somebody's name and somebody's Metasploit. What are you talking about?

Steve: HD Moore is the guy, the hacker, who maintains the Metasploit Framework, which is a very mature, sophisticated, growing over time framework that hosts exploits. And so it sort of - it just provides an easy way of, like, adding a small bit of code to do the work side of an exploit where the rest of the foundation is sort of there. And so this is - there are traces of the Internet, sample runs of giving a command to take over, to, like, poison the name cache on a DNS resolver. And you see it running, and you see it saying, okay, doing 10,000 queries, testing; doing 10,000 queries, testing. So what he would do is he would blast a bunch of queries, send a bunch of spoof responses, and then ask that name server, the targeted name server, for the record he was trying to replace, see whether he'd succeeded. And if not, do it again. And it would take, like in one case, like, 13 iterations of this over the course of maybe 10, 15 seconds, and it would say "success." And you would see that that server was now returning the fraudulent, deliberately inserted, replaced record from its cache. So that was 10 to 15 seconds.

We knew that the best we could do was to use source port randomization, as I described two weeks ago, to essentially give us more bits. The idea is if all we can do, if the source port is fixed, then the only thing the attacker doesn't know are the 16 bits of transaction ID. So you just guess a lot. And because there aren't that many combinations of 16 bits, you're going to get a collision. So, and the collision meaning that you guessed the transaction ID of an outstanding query that that server is waiting to have a reply from. You pretend to be the replier, the authentic replier, and your malicious reply is accepted as being valid. So we want more bits.

So what they did was they said, okay, well, we can't change DNS. We can't change the protocol. But we need more bits. So they said, oh, let's use source port randomization. That'll give us a bunch more bits, depending upon the specific configuration, as many as another 16. You actually lose some because many systems can't allocate ports down in the so-called "service area," the first 1,024 ports. And other configurations just have problems randomly allocating UDP ports all over the remaining space. For example, Microsoft's DNS server reportedly now, the new updated version, allocates about 2,500 ports. It, like, preallocates them, and then those it uses. Well, 2K is 11 bits. So that's giving us essentially an additional 11 bits on top of the 16. So we get, what, 72 bits total. So that's a lot more. It makes the attack - it makes the old attack 2,000 times harder.

Leo: But not impossible.

Steve: But, yes, that's the point is we want more bits. We're getting as many bits as we can within the existing framework. So what happened was late last week a Russian physicist, Dr. - looks like - I had to practice his name. I have been practicing it. And now I look at it...

Leo: I had the same issue on the radio.

Steve: E-v-g-e-n-i-y.

Leo: Evgeniy.

Steve: Evgeniy. And it looks like Polyakov, maybe

P-o-l-y-a-k-o-v.

Leo: Sure, that's good enough. Polyakov.

Steve: Anyway, he blogged on Friday that he had used two machines on a gig Ethernet, so closely coupled to the name server, meaning a ton of queries and replies.

Leo: Far more than you would in any normal circumstance.

Steve: Well, and difficult, arguably very difficult across the Internet. But it took him 10 hours. In 10 hours he got a collision on a state-of-the-art, recently fully patched, version 9 of BIND that had source port randomization and strong transaction ID pseudorandom number generation. So a state-of-the-art BIND took him 10 hours. So his point was that, okay, that's hard to induce that across the Internet because of the number of packets required. But a compromised trojan operating inside of an organization or inside of an ISP has that kind of intimate connection to the DNS server, meaning that an overnight or over weekend, meaning sort of like when no one's really paying attention, attack could succeed. And so basically we're seeing a proof of concept that, yes, it's no longer 10 seconds, it's now 10 minutes, and only on an intimately connected environment where you've got a lot of bandwidth available. And the people that have been countering this argument say, yeah, if you saturate a gigabit Ethernet, people are going to notice. You got lights blinking and wires smoking.

Leo: It's not exactly a surreptitious attack.

Steve: It's not very subtle, no. So but that has spawned, then, some follow-on dialogue. I did want to respond to a couple things. I wanted not to forget to tell our listeners that I've got a bunch of links on this episode's show notes. And actually they're online right now, Leo, so you can tell Dane and Tony that they can grab them and move them to your site also. I've got a link to Dan's - the PowerPoint presentation that Dan used, Dan Kaminsky used at the Black Hat conference, and also a link to this Russian physicist's page where he describes it, and all of the follow-on blog replies, which make an interesting read, as well. And there's one other link, I can't remember what it was, but I think there were three there. So I've got stuff up there on our show notes for this week that people may find interesting.

One of the things that Dan Kaminsky said I sort of take issue with, but he has a point.

And that is, he talked about how even SSL, our much-beloved Secure Sockets Layer protocol that we depend upon for authenticating and encrypting our connections, we authenticate inasmuch as we check the certificate of a site we're connected to to verify that this is really them, and we know that it completely scrambles our communications so we're not - no one can - no man in the middle, nobody intercepting our traffic or listening passively to our traffic is able to obtain the data that we're sending in the clear. So Dan, I think trying to make a little more of a deal, big deal about DNS than is warranted, was saying that SSL depends upon DNS. That is to say, everything depends upon DNS.

Well, okay. Where I agree with him is, the point is, yes, that's true. Nobody uses IP addresses. I mean, obviously I don't mean that in an absolute sense because we were talking last week about how using an IP address to access a test to see if your DNS was poisoned would be a good thing to do because you can't spoof an IP address. On the other hand, IP addresses are too static. And so there are problems associated with using an IP address, aside from the fact that they're hard to memorize and nobody - people know GRC.com and TWiT.tv. They don't know our IP addresses.

Leo: Right, right.

Steve: Exactly. So technically he's right. My argument is that, if DNS had been spoofed, then the bad guys who were trying to spoof your connection and to whom you were connecting, if you believed you were setting up a secure connection to a remote site that had been spoofed, well, your certificate protects you from that. That is, the fact that you're able to check the remote site's certificate and verify that it's really them. And we talked about, you know, for example, spoofing sites simply would not bring up an SSL connection during logon, whereas a legitimate site would do so. So technically you are protected from spoofing as long as you're aware that you need to have a secure connection and you check to make sure that the certificate you've received is really theirs.

Now, where Dan is right is that, okay, well, what about the way they got their certificate, or the way you got your updates to your certificates or your root certificates, for example. I mean, his point was since ultimately, like ultimately ultimately ultimately everything is relying on DNS, this notion of root of trust is subverted, the idea being that a certificate authority is a trust root. It's something you - I'm trusting VeriSign or Network Solutions or GoDaddy to really check to make sure that these are valid certificates they're issuing. And so we've talked about certificate chains. Well, that's a chain of trust where at each step you have enforceable trustability. Therefore when you get to the end of the chain, as long as all the links are trustable, then the result is trustable as long as the root is trustable. So he's saying, okay, except that ultimately all of those things that came before relied on DNS. And if DNS is not fundamentally trustable, then nothing else can be.

And so it's like, okay, granted. Except that we have to assume, I mean, I do assume that when Microsoft is sending me a certificate batch update, we know that their updates are encrypted, and we know that they're digitally signed, and we know that that can't be spoofed. So it's like, yes, pedantically everything relies on DNS. Practically, while this is a bad problem, I wouldn't go so far as to say that SSL is compromised by DNS being compromised. That, I mean, it requires a little too much stretching from theory to reality.

Leo: Well, so this - but I have to say these kinds of attacks often are just the precursor to a fast - remember it took for a while a long time to do WEP. And then they got faster and faster and faster. Does it seem reasonable that they might get faster? Or is it just really, I mean, you've got so many bits now. What is it, 71 bits.

Steve: Well, and so that's where we're going to go next with this discussion is the fact that we've done well, the question is have we done well enough.

Leo: Well enough, okay. A clever hack coming up, I take it.

Steve: Well, not yet. We're going to - we have a few other things that are interesting things that people are discussing. So first of all, someone has proposed that queries be - and actually the engineering term is "debounced," which I think is sort of a funny, is a funny term in this case, the idea being that you double query. You make a query, and you get the response. But you don't trust it until you query again. And the idea being that simply querying again and requiring, essentially debouncing or despoofing the query, if you got a different answer because you got the right one either time, either the first or the second time, then you say, whoops, wait a minute, this is suspicious, and then you engage some, like, deeper querying logic where you look to see if there's a lot of replies coming in, or you issue more queries and do some majority voting sort of approach.

And the argument against that I thought was really interesting because it tells us a lot about where DNS is in the world. Okay. Double, just doing two queries would obviously double the amount of DNS traffic. It would double the load on the resolvers, those doing the querying, and on the authoritative name servers, those providing the authoritative records. We don't have a hundred percent spare capacity in our DNS network today. So we don't - we can't even double the number of packets that we're sending. DNS servers are so busy, and their links are so near saturation, that they are more than half full. They are more than half capacity. And so even something like double querying we cannot do. There's just - there isn't the capacity in the existing Internet DNS to allow for that.

So that automatically rules out the next idea, which was, okay, this whole problem is from spoofing, and UDP protocol is infinitely spoofable because nothing prevents you from putting whatever source IP you wish into an outbound packet. Well, okay, not having raw sockets prevents you, but UNIX systems and Linux systems do allow full raw sockets and allow the programmer to build any packet that they wish and send it out on the line. So with those limitations, UDP is absolutely spoofable, where TCP is not. As we've talked about and understood TCP protocol, the whole overhead of that three, the so-called three-way handshake, where we send a SYN packet to the place we want to initiate a connection, they send a SYN/ACK back, which is their SYN and acknowledging ours, and we send a final ACK back to them acknowledging their SYN. So that three-way handshake, three packets that establishes counting for the packets that will follow, then you can send a query and receive a reply. Then you've got to say I'm done now, so you send a FIN packet, and they respond with a FIN/ACK.

Well, that's, if you count all those, three to initiate, two to shut down, plus two in the middle for the query and the reply, you're at seven packets. So you've gone from two packets, a simple UDP query and reply, to a seven-packet exchange. Now, there's actually a way to save one because it's a little-known fact in the TCP protocol is that you are able to send data with your first ACK. So that would cut out - that means that when

the querier is sending its ACK back to the server, it could include its query with that if the stack at the receiving end was fully TCP spec compliant. And it's not clear whether they are or not, so it's generally not done. But even then you're at six packets, which is three times the number of packets required to make your normal UDP query and replay. So that's even worse than, you know, using TCP is worse than just double querying.

So, finally, the issue of DNSSEC comes back up. Now, DNSSEC is the acronym for DNS Security. This is a very complex, very sophisticated sort of next-generation secure DNS which has been bubbling along, simmering for about nine years now. And it just - it never seems able to get off the ground. A couple times the various groups have attempted to ratify it and lock it down and say, okay, this is what we're going to implement. Then they start looking at the implementation cost in the real world and inevitably back off again and say, oh, wait a minute, that means that if the root server changed its private key, it would have to send out 22 million copies of its public key, you know, that kind of thing. I mean, it's like, oh, okay, that's not good, how do we fix that?

So the problem is, DNSSEC is what we're going to end up with ultimately because on the level of pedantic purity Kaminsky is right, that everything relies on DNS ultimately. And if we don't have a strong foundation of trust, nothing we build on top of that non-trustable foundation can by definition be trusted. So there's work towards DNSSEC. And we're going to give it a complete episode because it's very sophisticated. But here's the fundamental problem. It is a public key technology, the idea being that it signs, DNSSEC signs, for authentication purposes, queries. Or you sign responses to queries so that the receiver of the response is able to check the digital signature of what it's received from the authoritative name server to verify its correctness. Interestingly, they do not encrypt the data. So it's sniffable. But the presumption is, all this is is an IP address, so everyone - and that's public information. So let's not spend any time encrypting what is already publicly known. So it is, however, authenticated and digitally signed.

Well, digital signing requires a public key process which has substantial processor overhead. So in addition to making the packets a little bit larger, both the sender and receiver, the sender would have to generate the digital signature; the receiver needs to verify it. And so again we're talking about substantial new overhead added to an existing structure. And for that reason it just - they can't get off the ground. John Markoff, writing for The New York Times recently, noted that several governments, Sweden and Puerto Rico, have adopted DNSSEC. And the United States, our government, is likely to deploy it, at least for the .gov domain.

Leo: Yeah, in an official capacity, where you have lots of money and a real strong need for security.

Steve: Exactly. And essentially, I mean, if this were something you were doing from the beginning, this is the way to go. So you can imagine a smaller infrastructure, like you may have in Sweden and Puerto Rico, or just determination. They say, we decided this is important. We're going to implement this. And so it means replacing the existing iron with much stronger iron, and maybe in fact adding hardware accelerators for SSL, I mean, sorry, for PKI, Public Key Infrastructure acceleration, so that you're able to offload some of this in the hardware. I think it is clear that we're going to see an increasing market for hardware acceleration of public key operations moving into the future because this is the way to secure things where your traffic is eavesdroppable, where it is possible for someone to see what it is you're sending. As we've discussed before, public key technology solves that problem. So there is one really, really clever solution that we'll talk about next.

Leo: The clever hack solution.

Steve: Oh, this is a hack. This is a hacker's hack.

Leo: So what is this - I'm excited. I'm dying to hear this hack, this workaround.

Steve: This is the galactic hack. I don't know if I can think of something that is more a hack, more sort of like, oh my goodness, but clever. And so, I mean, this is the definition of a hack. And it works without any additional overhead, without any additional packets, without any additional anything within the existing infrastructure of DNS. So, and remember that, as I said toward the beginning of the show, what we want is more bits. The idea being, we want the query that the resolving name server sends out to contain more entropy, more bits of randomness which the responder will be able to easily send back, such as the matching query ID and the matching port number, but also something else. What more could it possibly send back within the existing definition of DNS? Okay, get a load of this. This is hack of the year.

Leo: All right.

Steve: It's referred to as the 0x20, or the 0X20 hack.

Leo: Is that the hex? Is that, like, hex 20?

Steve: Yes, that's hex 20, 0x20.

Leo: Just space; right?

Steve: Actually, that's very good, Leo, hex 20 - wait a minute.

Leo: Yeah, it is.

Steve: Yeah, hex 20. But it refers to the 20 bit because that is the difference in an ASCII representation of text between uppercase and lowercase.

Leo: Right. You add that bit, and it's uppercase.

Steve: For example, 41 and 61 are the hex for upper and lowercase A.

Leo: Right, right.

Steve: The difference being that 20 bit. So get a load of this. The DNS spec, the original RFC that everybody wrote to and coded to and follows, says that case is not significant, but it will be preserved. Meaning that - and you'll notice this. You could do GRC.com in all uppercase, or GRC.com in all lowercase, or GRC.com in any random combination of upper and lowercase, and you'd get to GRC.com. And I notice that, like, TWiT.tv is capital T, capital W, lowercase I, capital T; right?

Leo: Right.

Steve: And that works. The point is, case doesn't matter. But in a DNS query and response, case is preserved, meaning that when I issue a query, the response I get comes back with the same case of the alphabetic characters as I issued. Yet the authoritative name server that is checking to see if it's got a match within its records, it ignores the case, doesn't care if the alphabetic characters are upper or lower, to find a match. That gives us more bits.

Leo: Oh. That is clever.

Steve: Oh my goodness, isn't that just incredibly cool? What it means is that the querying server can randomly upper and lowercase all the alphabetic characters in its query. So, you know, we've got www.domainname.com. So we've got, okay, www, there's three; com is three more alphabetic characters. Plus however many alphabetic characters are in the name. And more if you are several - if you have several domain, dotted domain names. All those characters can have a random case. They will go to the resolver that will - I'm sorry. They will go to the authoritative name server that will ignore your wacky casing that you've done.

Leo: Right, because it's case insensitive, yeah.

Steve: But it will return your - it will return in its reply, it returns the same case that you sent it, which means you can - you the querier, who is trying to be spoof-resistant, can verify...

Leo: Oh, if it's the same one you sent.

Steve: Yes.

Leo: But wait a minute. Numbers, dotted quads are all the same case. How would it reply in a different case?

Steve: Well, okay. The idea is in the query goes the domain name...

Leo: Oh, in addition to. Okay.

Steve: In addition to. So the domain name - basically the response echoes the query and adds the answers. So you get back what you queried as part of the answer.

Leo: Plus the dotted quad. But now if somebody's in the middle, if they're doing a man in the middle, they're going to see your randomly capitalized query. So can't they just copy it back?

Steve: Oh, Leo, DNS is completely hosed by man in the middle. Man in the middle...

Leo: Oh, this doesn't protect against that. This - okay.

Steve: No, no. In fact, man in the middle is a single query attack because if you're able to see the query and block the reply, you simply respond because you know exactly...

Leo: Whatever you want, right.

Steve: Exactly. So and then...

Leo: Okay, all right, all right. So this only is good against cache poisoning.

Steve: Well, this is good - well, and which is the problem that we're trying to solve. This is good against a third party trying to guess queries in order to spoof replies. What this means is the query is now much more difficult to guess by the number of alphabetic, you know, two to the power of the number of alphabetic characters in the name being queried. So, for example, there could easily be 10 alphabetic characters - www and com gives us six. Seven, eight, nine, 10, so even a four-letter domain would give us ten additional bits. Well, 10, that's another 1,024 possibilities. So we've just added 10 bits of difficulty, looking up www and a four-letter domain, and many domains are much longer than that. So it's just, I mean, that is a hack. I mean, it's like, it's ugly, but it works.

Leo: So that's just something you obviously have to patch the DNS servers to do.

Steve: You don't have to patch the authoritative name servers. That is to say, they're already programmed to echo the incoming packet's case and ignore the incoming packet's case.

Leo: Oh, so it only would have to be the querying servers you'd have to change.

Steve: Right. So, I mean, just as we - we just changed the querying servers this month so that they would do reliable source port randomization. So we would just need to change them again if the decision were made or if anyone wanted to make the decision, or you could imagine this would be, you know, all this stuff is open source, in the case of BIND, for example. So you could have a BIND compile time flag saying I want this server

to employ the 0x20 hack, and then your build of your BIND would issue queries with random alphabetic case on all the alphabetic characters, thus making itself far more difficult to spoof. And the beauty is it would automatically work with all the other servers on the 'Net that don't need to be changed. So this is something that individual, for example, BIND users, companies, ISPs, end-users, could easily do when this is an available compile-time or run-time option of BIND that just makes that one server much more resistant to spoofing.

Leo: Easy thing to add. But it would be a patch. You'd have to change the send, and you'd also have to remember that and the query and look at the result and make sure that they matched. But that's an easy compare. That's pretty fast.

Steve: Right. So we're talking, exactly, we're talking additional code. Already obviously there is code which verifies that the incoming port and the incoming transaction ID match up with what's expected. So you just increase the expectation to verify that the incoming reply had case of its alphabetic characters that matched what you sent out, which is what you were expecting. Right now, no servers, as far as I know, care. So all you need to do is to add that they care about matching returning response case, and then randomize your outgoing case, and you've got a much harder to spoof server. Just you. You don't need anything of anybody else. Your server is now substantially more difficult to spoof.

Leo: I really like that. So...

Steve: It's so cool.

Leo: Is there any move to do this, to implement this?

Steve: It's too early. This is just hot on the wires right now. People are beginning to talk about this. It's like, hey, you know, why not? It's more bits.

Leo: I'm presuming that there's some sort of a process where people would want to vet it and think about it and really see, oh, does this really work, would it really solve the problem. And of course always with fixes like this there's always some physicist named Evgeniy Polyakov who's going to come along and say, oh, yes, but we just do this.

Steve: Well, okay. So counter arguments. What about, like, all numeric domain names? Okay, well, fine. I don't know of any that I've typed in recently. But sure. This is the amount of protection is a function of how many alphabetic characters are in the lookup because it's only those whose case you can change.

Leo: I have a domain name, for instance, 8888AskLeo. That's the phone number for the radio show. But there're still six alphabetic characters in there plus the dotcom, that's nine. And plus if you do www, that's 12.

Steve: Yes. And it's like, okay, it's free. These are free bits. These are bits that nobody was thinking about. It's like, wait a minute, the case bits. They're preserved, and they don't matter. So let's make them matter in terms of matching on the verifying the response because they already don't matter over at the query end where the name server, the authoritative name server ignores the case, as we already know. You can put in capital GRC.com or lowercase GRC.com. The query typically goes out with that case preserved, and it comes back the same way. It just hasn't mattered until now. We make it matter, and then our own use of DNS is much more difficult to spoof. That's just a tremendous hack.

Leo: It is, I mean you say it's extra bits. I mean, it's not exactly bits. I mean, I guess it is if you were going to try to brute force it, it's quite a few bits.

Steve: Yeah, well, your replies would then have to...

Leo: To be case sensitive.

Steve: ...be randomly upper and lower casing all of the ASCII also and hope that you get a collision. And, well, so now you've got to get a match of port number and transaction ID; and every single character, every ASCII character in the query doubles the difficulty. So if that's 12 ASCII characters, that's doubled 12 times is 4,096 times more difficulty.

Leo: That's good.

Steve: So, yeah, it's really cool.

Leo: It's helpful. Wow. That's neat. Somebody's really kind of thinking out of the box there to come up with that. I like that a lot.

Steve: And it's, yeah, I mean, and the thing I like about it, too, is again, when this is implemented in BIND, I know that in our own user groups we've got a lot of people running BIND. And they're not happy, for example, maybe their NAT router is messing up source port randomization so they still feel too vulnerable. They could compile a version of BIND with this on, with the 0x20 hack turned on, and just them get more spoofability protection, and nobody else needs to change anything else on the Internet.

Leo: Wow. That's really neat.

Steve: It is neat.

Leo: That's really neat, yeah. Steve, you've done it again, my friend. You've come up with a very interesting show all about DNS and potential flaws. It's as you said before, it's a seesaw battle back and forth always. Nobody gets the upper hand

forever.

Steve: That's true. But people are now worrying, after hearing the Russian physicist hacked his own local server on a gig-E in 10 hours, it's like, okay, DNS is too important for us not to do something. Now, if nothing else, this whole drama over the last month could provide necessary steam for DNSSEC. That is, ultimately incorporating public key technology into DNS will really strengthen it. I mean, make it as strong as it arguably needs to be. But the problem is it's been so expensive in terms of the overhead that DNSSEC brings, that it just hasn't happened yet. So the least that'll happen is people are thinking, ooh, you mean that it's still not fixed? It's better, but it's still not fixed? And it's like, yes, it's not ultimately fixable. All we can do is raise the bar. So using case in the query names raises the bar substantially. For example, it raises it more than Microsoft's own port randomization change did. Because reportedly they only preallocate 2,500 ports. They're only giving us another 10 bits. We can get that much using case sensitivity easily.

Leo: Wow.

Steve: So it's a good thing.

Leo: Yeah, it is a good thing. And next week we're going to do a question-and-answer session. But the week after, let's talk about DNSSEC, why don't we? Is that what we want to do next time?

Steve: No, remember, week after next we want to talk about the interesting presentation at Black Hat where these guys have bypassed Vista's much heralded...

Leo: Oh, that's right, we've got to do that. Then we'll do DNSSEC later. But I think that's worth talking to. Yeah, we don't want to get too DNS heavy. And I think this Vista thing is more - see, the thing about this DNS is it's not an end-user problem. There's nothing you can do about it except to encourage your ISP to do it.

Steve: Oh, oh, oh, but Leo, I mean, the reason I wanted to spend another episode on this is the end-users are really freaking out.

Leo: Rightly so.

Steve: I mean, it's not something they can fix, but it's something they want to understand and, for example, have their general level of spoofability awareness raised. It's like, wait a minute, is this really eBay.com? I am going to actually check my security certificate rather than...

Leo: Nobody's doing that.

Steve: ...just assume it's okay.

Leo: I'm not even doing that. And these exploits are in the wild now.

Steve: Yes, yes. This is now being actively exploited. There is, out in the blogs, there are all kinds of reports of these tools being employed against weak servers.

Leo: Wow. Oh, the bad guys, they are bad. But Steve Gibson is here There's no need to fear, Steve is here. His website is GRC.com. Great security stuff there. He's working on a new security program that's going to be very cool. I look forward to that. But you can also find ShieldsUP!, Shoot The Messenger, DCOMbobulator, and even Wizmo, a great little gadget for your desktop, XP or Vista, all at GRC.com. SpinRite of course is the obvious choice there. It's a must-have, the must-have hard drive maintenance and recovery utility. I'm going to have a SpinRite fest right after the show today.

Steve: Yes, it sounds like you are, Leo. That'll be great.

Leo: A lot of spinning to do. And you'll also find 16KB versions of the show and transcriptions, too, at GRC.com.

Steve: I will remind our listeners that GRC.com/feedback is where to go to send me questions, which I scan through in preparing our every-other-week Q&A episode. So GRC.com/feedback.

Leo: Our first Q&A episode of our fourth year, Ollie.

Steve: Of the fourth year.

Leo: That's unbeliev- mind-boggling. You know, it's funny, too, because we are dealing with much more sophisticated stuff as time goes by. Things like this DNS stuff is pretty sophisticated. But...

Steve: That's cool.

Leo: ...that's what we do. Thank you. That's what Steve does. I just sit here and go, oh, yeah, I think I understand. Thanks, Steve. I really appreciate it.

Steve: Thanks, Leo. Talk to you next week.