SECURITY NOW!

Transcript of Episode #158

Listener Feedback Q&A #48

Description: Steve and Leo discuss the week's major security events and discuss questions and comments from listeners of previous episodes. They tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world 'application notes' for any of the security technologies and issues we have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-158.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-158-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 158 for August 21, 2008: Listener Feedback #48. 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!, time to protect yourself and your systems and your Internet. And here he is, ladies and gentlemen, the star of our show, Mr. Steve Gibson. Hey, Steve.

Steve Gibson: Yay. Hi, Leo.

Leo: It's good to talk to you. Steve is, of course, the guy behind SpinRite and many other wonderful security programs, as well, like ShieldsUP!. And he was the guy who first discovered spyware and let the world know about it, actually wrote the first spyware program but quickly passed it on to the folks at Ad-Aware, let them do the job. And he joins us every week to talk about the latest in security. This is a Q&A week, isn't it.

Steve: It is a Q&A week. I've been having a ball for the last couple weeks coding a set of pseudo DNS name servers for GRC's forthcoming very, very high-quality DNS spoofability test.

Leo: Oh, cool. Oh, cool.

Steve: So it's going to be extremely nice. I'm real - I've just been having a ball. It's like there's nothing I like better than just keep my head down in the code and...

Leo: You love coding, don't you.

Steve: I do. It's just - it's just perfect.

Leo: I've done - I've been, you know, I'm learning a language called Lua which is a very nice little scripting language, very similar to Python or C. But it's used in - it's so lightweight, and it's very easy to embed into applications, so it's used in a lot of applications as their scripting language. So it's kind of a handy thing to know. And it's in World of Warcraft, for instance. It's in - I started using it because of a program I use on the Mac, VoodooPad, that uses it. But now I'm kind of into it. And yeah, it is fun, it's fun to write little dippity-do programs. And a scripting language is very good for that because you have an interpreter, and it's very easy to test your stuff out. And it has some nice little features that make it fun to write. So I understand how you feel.

Steve: Yeah, it's just - I just love it.

Leo: I regret, in a lot of ways, there are a few things that I would have liked to have done in my life, and one of them is be a coder.

Steve: But your path was different, Leo.

Leo: My path was different. And I probably wouldn't have been very good at it anyway. So there. You know what I'm really...

Steve: I actually disagree. I think you've got a very logical, very, you know...

Leo: You're kind.

Steve: No, I'm serious, I think you would have been good. And you're good at this, too. So this is what we need you for. I'll write the code.

Leo: Yeah. You write the code, and I'll...

Steve: You create the TWiT zone.

Leo: All right, now, Steve Gibson, I have 100 - I'm sorry, this is 158. I have 12 great questions for you. But before we get to those, I do want to find out if anything has happened in the world of security since we talked last.

Steve: Well, the good news is no, not much.

Leo: Hey, I like that.

Steve: Last week we had the, you know, one of the grand mal Windows patch, second-Tuesday-of-the-month updates in a long time. I mean, like a long list of all critical security problems that Microsoft was patching and did patch. And I noticed, too, there seemed to be some stray ones that kind of came along later. I think twice after I did the full update myself on Tuesday afternoon, I was alerted twice more between then and now that, oh, here's a little something else. It's like, okay, I guess this one wasn't cued up when I did my manual restart and check, or who knows what. Maybe they were just fixing some things. But anyway, it was, you know, last week was a big one. This week things have been relatively quiet on the security front. The one interesting sort of news item that I saw and I thought was sort of interesting was that Congress is beginning to understand about the problem of ISPs doing deep packet inspection and...

Leo: Oh, hallelujah.

Steve: ...and being involved in spying on their own customers. And in rather startling congressional testimony they - actually it was the questionnaires they sent out to - but under the congressional stamp, so you want to tell the truth on these. Almost all of 30 major connectivity providers, ISPs, confessed to Congress that at one time or another, and maybe even now, they were doing surreptitious monitoring of their customers' traffic without their customers' knowledge or permission or even mentioning it in the license agreement.

Leo: Some of them admitted to using NebuAd, even.

Steve: Yeah, I think at least two of them said, oh, yeah, we tried NebuAd, and we're not doing it now, but we're looking at this in the future. And they said, but, you know, we would like to make it an opt-in thing. Like...

Leo: Well, that would be okay; right?

Steve: Oh, absolutely. I have no problem with that at all as long as people have that opportunity. The good news is the EFF folks, the Electronic Freedom Foundation...

Leo: Electronic Frontier Foundation.

Steve: Frontier Foundation are, like, on top of this and financing educating Congress and doing what they can to keep this thing from spiraling out of control. And what I think Representative Markey is...

Leo: Yeah, Ed Markey, yeah.

Steve: Ed Markey, who is, like, big on privacy, he's making noise about working to pass a law next year that would require opt-in.

Leo: Good.

Steve: And if we could have that, that would be spectacular.

Leo: Well, I think that would be good for everybody, including the industry, because the industry can, you know, most people are going to opt in if you give them a reason to opt in. So the industry just is upfront about it instead of sneaking around and doing it; right?

Steve: Yeah, exactly. And it was funny, speaking on that, I saw another little thing where there were some strong cautionary advisories sent out to people traveling out of the U.S., and I think the timing was - it wasn't coincidental that it was relative to the Olympics in China. And that is in basically just warning people to take your electronics and the data on them very seriously relative to security, that in other countries that don't have even the fundamental constitutional privacy rights that we do in the U.S., your hotel room can be searched. You can be asked to hand over any electronic device you've got by a customs official who can dump the contents of it. And in some cases you can be prevented from bringing into a country encrypted data. They just say no.

Leo: Wow.

Steve: Yeah, wow.

Leo: So if they say - if you have something encrypted, they're just going to take it off. I mean, they're just going to say turn around, you can't...

Steve: Yeah, if we can't read it, you can't bring it in.

Leo: Wow.

Steve: So, you know.

Leo: That's one way to handle it, though, rather than saying you have to unencrypt it for us.

Steve: Yeah, exactly.

Leo: Say sorry, you want to leave your laptop with us, you can come into the country.

Steve: And then also, following on from last week's discussion, remember we were talking as you were getting ready to head off to Podcast Expo, you'd had some problems with RAID 0, and we were talking about the importance of not running SpinRite in front of the RAID controller, but rather running it on each of the drives individually.

Leo: Right. And I haven't had a chance to do that yet. But, yeah, I've got to do that, yeah.

Steve: I got email from someone, actually I ran across this as I was catching up on my mailbag in preparation for today's Q&A, some guy named Doug Sauer, he said, "Steve and Leo, thank you for your podcasts. The DNS discussions are so important at this time. Unless we are paranoid freaks, many IT professionals tend not to believe the evil of the hackers out there today." He says, "Okay. One of my home systems has a three-drive RAID 5. This system is four years old now, still running fast enough because I bought an Intel board with dual 2.2GHz Xeons back then. The RAID reported that I had lost a drive two weeks ago."

Now, of course since he's in RAID 5 he's running three drives. So that means he's got the space of two drives, so that any one drive could fail, and he would lose no data. So essentially that's more efficient, for example, than a pure mirroring where you get exactly 50 percent of your total drive space. Here he's got one drive essentially acting as a parity drive so that any one of the three can die, and the RAID keeps on going. But in this case it said oops, a drive just died. And he said, "So I pulled the bad drive bay out and ran the bad drive on a freshly purchased copy of SpinRite. Presto. Disk fully recovered. I also separately ran each of the other two drives, and SpinRite found and repaired errors on each of them, as well. I will now be imaging to an external drive and coming up with an upgrade plan.

"I was not going to send a letter to you as I imagine you get so many accolades. However, your last podcast with Leo you advised him to do a drive at a time on his RAID 0. For once in my life I was just one small step ahead of Leo. This temporary state of euphoria prompted me to write to both of you. I appreciate both of you and your service to us as listeners. I forwarded off all the podcasts relating to DNS to all my team members. They're a big help as the DNS issue continues to unfold. Thanks again, Doug Sauer, Indianapolis, Indiana."

Leo: Very cool.

Steve: So thanks, Doug, for your note. Really appreciate it.

Leo: Glad to hear that. All right. Let's get to the questions. I've got them. You ready?

Steve: I'm ready.

Leo: Ready to answer some toughies? Well, I shouldn't say they're tough. I don't know.

Steve: Well, they're interesting and illuminating.

Leo: Illuminating.

Steve: Just what we always look for.

Leo: You picked them, so I figure you must have something to say about them.

Steve: Yeah, I'm pretty ready.

Leo: Paul Sargent starts us off from the U.K. He wonders whether debouncing is really double trouble: Hi, Steve. I was listening to your Security Now! podcast Episode 167 on the continuing DNS vulnerabilities. At one point you stated that the idea of debouncing spoofed DNS by requiring two identical query results had been aired but discarded because this would double the number of packets required for each lookup. Is that really true? The key point is to only allow changes to the DNS when they've been validated with a second query and reply. Each DNS server holds a cache of results from previous lookups. So it will only generate a new request in one of two cases: One, it hasn't seen the address before, boom, then you're going to ask again; or, two, the TTL, the time to live, has run out on the cached result. In this case a single request would be generated. If the result is the same as the cache, no second request necessary. If it has changed, then it would be verified with another request.

So what he's saying is you'd only have to make that second request if something seemed out of the ordinary or was a new address. Assuming that DNS records rarely change, the number of additional verification query packets related to TTL expiration would be very low. The question then becomes, how often are DNS lookups a complete cache miss, that is, no record exists, expired or not, because that's the only case when the 2x increase would occur. Is that often enough that you get a cache miss? Is it still a problem? So is he understanding the idea of debounce properly?

Steve: Well, kind of. First of all, I liked his clever notion that, if a DNS name server had an entry in its cache which it was going to verify by asking again, then if it got the same answer, and we know that, I mean, the whole concept of DNS is that it's a distributed database, so chunks of the current mapping between domain names and IP addresses are able to float around on servers that only periodically check in to see if there's been any change. So that's a really nice tradeoff. It means that there's no, like, sort of central server that bears the brunt of everyone asking questions. But the tradeoff is that you don't get instantaneous updates in the event that an IP address changes. So, for example, I know you and I, Leo, have from time to time needed to change the IP address of a domain.

Leo: Right.

Steve: And there's this notion of it needing to propagate through the Internet. What that really means is that all spread around the world are, in locations where people have accessed a domain recently, there is a slowly expiring copy. And as long as it's not yet expired, then when questions come to that server, they'll be answered from the cache rather than going back and having to re-resolve it immediately. So it's a clever, beautifully clever solution. So he's noted that, in the event that a name server has some data in its cache, when it sees that it's expired it could only make a single query in order to verify that what's there is still current. And if the result it gets back is different, then that raises its suspicion that, oh, wait a minute, maybe I need to check this a few more times to increase confidence that I didn't get a spoofed response. So, I mean...

Leo: Well, that makes sense.

Steve: Oh, absolutely. I mean, he's right about that. The one thing that this misunderstands, which is why I really liked the question, is that what it was that Dan Kaminsky realized was that you could force servers to make queries rather than waiting for their caches to expire. That was the brilliant gotcha that occurred to him. It used to be, I mean, if you - like three months ago, before this whole DNS spoofing nightmare arose, you could put DNS spoofing or spoofing DNS or something into Google, and you'd get pages and pages and pages. I mean, this notion of spoofing DNS is not new. But everyone believed that you had to wait for the server's existing cached record to expire, and then you could make - as soon as it expired, it's not going to replace it all by itself. It's going to wait for a query. And then, upon receiving a query, it checks the record to see if it's expired, thereby launching its own query out to resolve that name.

So the idea would be you would, you know, you bad hacker person would sit there, wait for the record to expire - because when you ask the server, it tells you how much time there is left remaining on that record. And that's a cool thing, too. If you think about it, it has to because then you might be caching that record, and you don't want to start at eight hours again. You want to understand how stale the record is so that your own cache will expire at the same time that the cache of the source of that record was. So anyone asking a DNS server - you can tell I've been living in DNS for a couple weeks, and I've really got this stuff now on the tip of my tongue. Anyone talking to a DNS server, querying it, knows how much longer it'll be before one of its records will expire. So they're able, as soon as that happens, that's when they launch their query to it, knowing that it's having to launch a query out, and then they rush their spoofed response back in. That's the traditional way. And that limited you to one shot at replacing a record, only every TTL interval. Which is typically one day. A standard Internet time to live for DNS is a day. So once a day you had a tiny window of opportunity. Clearly this wasn't a huge problem.

What Dan realized is you could ask it for a bunch of nonexistent machines in a domain, forcing it to constantly ask upstream for that domain's name server if this machine exists. And you could sneak in a response to that request that carried replacement name server records. And of course that's the nature of the attack. So Paul was right that, if we didn't have what Kaminsky realized happening, then you could do verification essentially in a cost-effective manner, only duplicating some queries if a domain was asked for that was not in the cache, and then only again if a verification that the IP had not changed from what was in the cache came back with a different answer. So I think Paul was clever, but that doesn't solve the problem that we've just had.

Leo: Could you rewrite the way DNS works so that you couldn't ask for these multiple updates? I mean, isn't that a bug, too?

Steve: Oh, yeah. I mean, there's all kinds, well, I'm sure right now, Leo, there are firewall vendors, you know, corporate firewall vendors madly adding code to their firewalls, and they'll have new bullet points on their new brochures in a month saying "Special next-generation DNS spoof-proofing for your corporate DNS server." So there's all kinds of things you could do that would, I mean, make it really obvious that, you know, here comes a flood of responses in response to one query? Okay, that's wrong. I mean, it stands out like a sore thumb. But right now nothing is aware of that. So I'm sure there are - doubtless firewall vendors are madly rushing to get theirs to market first because essentially one query goes out and 10,000 come in. It's like, okay, maybe I'm going to ask that question again.

Leo: Something wrong there, yeah.

Steve: Something's wrong.

Leo: Yeah. Good, all right. On to Question 2. An anonymous listener asked a great question. Is IP whitelisting a secure method of limiting access to a website? Hi, Steve and Leo. IP whitelisting. I'd like to run a private company wiki, but located offsite using PBwiki. I want to make access for our users easy, no user or password required, so I can use PBwiki's whitelist feature. So you essentially say - you make a list of IP addresses that are allowed to log on, they just log on automatically. All traffic for a business account at PBwiki runs over SSL. So if I were to whitelist only our company's IP addresses, would this be a secure way of limiting access and keeping everyone else out? Thanks for a great show. I've been a listener since Episode 1.

Steve: The answer is yes.

Leo: Even with spoofing? Oh, because he said SSL.

Steve: Exactly. Well, and remember now that wiki access is going to be over a web page, and web page is TCP based.

Leo: Right.

Steve: So even without SSL you are unable to spoof an IP. Nobody at a different IP is able to obtain a connection because your connection attempt will be checked against a valid IP range. And even if you sent a SYN packet in from a valid IP range, that is spoofing your source IP, so that that incoming SYN packet to the wiki site carried the IP address of the corporation, well, when it acknowledged that, it would go to that IP.

Leo: Oh, that's right, of course.

Steve: It would go to the corporation. So...

Leo: So spoofing the incoming packet's not going to work.

Steve: Correct. The only vulnerability, and it's not a big one, is some sort of a man-in-the-middle attack, that is, even SSL won't prevent you because we're assuming that any browser that's using SSL is going to be connecting. So if somebody, like, in an ISP or anywhere in the line of your traffic were able to, outside of your corporate enclosure, were able to get into your traffic, that is, so that they were able to receive IP packets bound for the corporation, that would allow them to create a connection that would be validated by the IP whitelisting. But that's - we're essentially saying, well, sure, you'd have to be an ISP.

Well, ISPs have access to all of our traffic anyway. I mean, Level 3 has access to all of my traffic. I mean, I'm protected myself for the things that I do between home and my network at Level 3, you know, SSH and so forth, and a secure VPN. But technically anybody in the line does have access. So IP spoofing certainly protects any random person anywhere on the Internet from being able to make a TCP connection to a web server like for using a wiki. But it doesn't protect you in the case of an ISP. And, by the way, if you were to use a username and password, well, that could be sniffed, except an SSL connection would encrypt a username and password. So that's one place where SSL would make sense. If he was using SSL and was concerned that somebody that had access to his wire might be reading the company wiki, then that's where using a username and password, or just a passphrase or something known only to corporate individuals, that would not be sniffable by somebody doing a man-in-the-middle attack because you can't break into an SSL connection.

Leo: Yeah, yeah. Very good, and very simple. So it's safe. IP whitelisting would be safe.

Steve: Yeah, it's a good solution. In fact it's one of the things I use. There are some ports that are open at Level 3, TCP connections, which it was impractical for me to block or to run SSL over. So I have some filters at Level 3 that give my IP range here at home special rights, using the IPs to get special access to Level 3. And it's not spoofable.

Leo: Very cool. Would you recommend not having passwords? Or wouldn't you just have passwords too, just to be belt-and-suspenders?

Steve: Well, I guess, again, it's like anything in security is here's the tradeoff. He says he wants to make it easy for his company people to access the wiki, where he'll just give them a URL, and they'll just, bang, there it is, no logon required. That's really good because he can't give me the URL and have me access his wiki. I can't. I'm not at his IP. I have no access to his traffic. So it's a tradeoff. All he has to understand is that an ISP or somebody carrying his traffic could put a browser there and intercept his traffic and get access to his corporate wiki. That's probably not a big problem based on what's there. He just wants to keep it private. It doesn't sound like it's state secrets. But if, knowing that, he decides, okay, it's worth using a passphrase and SSL so that absolutely nobody outside of our corporation is able to access the wiki, and in which case you might then remove the filtering because that way an employee could access the wiki from home.

Leo: It looks like it's just local.

Steve: Exactly.

Leo: Oh, home, yeah, home.

Steve: Yeah, I was going to say yeah because, I mean, that might - somebody might want to be able to access the corporate wiki from home. The IP filter absolutely blocks that; whereas, of course username and password would permit it.

Leo: Right. Stroker in Florida poses some interesting questions about DNS: Hi, Steve. Regarding DNS, why does a domain's name servers need to be able to change their IP numbers at the top level domain name server? When we register a new domain, we specify the domain name server's IPs at the registrar; right? So why can't we just make the registrar the only place to change the domain's DNS IP numbers? Then we only need to make the registrar-to-TLD DNS name server connections secure, like via IP numbers only. As to the SSL worry, why should we - do you want to answer that, or should I move on to this second part, do it all...

Steve: They're all kind of related.

Leo: Okay. As to the SSL worry, why can't certificate authorities use IP address only, no DNS? Certificate authority IPs could be included in the CA list used by the browser. That's the Certificate Authority. I suppose the CAs depend on DNS for traffic leveling, that is, bandwidth distribution using multiple IP numbers. There must be another way to do that. I think financial institutions and other mission-critical sites should be required to allow access by IP number. I have tried such access, but most are using traffic leveling so that I could not access them directly by IP number. That's interesting. Couldn't they just distribute a range of IPs among their customers when giving them out?

Steve: That's practical.

Leo: When you visit our bank, go to 192.168.1.4. That's your number. Couldn't they do some kind of redirect at the server or use some other means to distribute the load besides IP distribution? This is some random thoughts. Thanks for your time and efforts. Maybe you should explain what he's proposing here.

Steve: Oh, I'm going to. There were sort of a number of misconceptions embodied in the questions, and Stroker was not the only person to have them. I ran across this a bunch of times. So I sort of wanted to clarify some things. There were a number of people who maybe were confused by me last time or the time before, talking about the nature of DNS spoofing and SSL certificate verification. That is, there seemed to be this sense, and this is what Stroker was referring to in the second part of his question, saying, well, why don't certificate authorities use their IP addresses?

The belief, I think, that I saw among several people was that there's a connection being made to validate and verify a chain of authority, a certificate chain, at the time that an SSL connection is made. That is, so you're not only connecting to a foreign site, but you're also at that time connecting to VeriSign or GoDaddy or whoever has issued the certificate. And so I wanted to make it clear that that's not the case. Your computer and specifically your browser comes with it built in, all of these certificate authorities, the so-called "root authorities" which are signing the certificates you get from websites when you attempt to connect to them. So when you initiate a connection over an SSL connection to a secure website, your browser gets the certificate from the other end and then verifies its digital signature, verifies that it has been signed by one of the certificates in your own local cache of certificate authorities. So there is no connection being made constantly to some other remote certificate authority. So that doesn't need to get changed.

The other thing, in the first part of his question he's asking, when we register a new domain, we specify the domain's name server IPs to the registrar. And he says, so why can't we make the registrar the only place to change the domain's DNS IP numbers? What he's saying is, why can't - because the real problem, the power of the attack that Kaminsky realized was that when responding to a bogus machine request for a domain like xyz123.google.com, if you could respond to that request and get it in, then that response could contain replacement name server records that would replace the records in that name server for Google.com. That was the whole power and the horror of what Kaminsky discovered. And so Stroker's saying, wait a minute, why not disallow that?

Well, okay. The problem is that that would require far more accessing up the domain name space. Essentially what he's saying is, why not disallow caching name server records? And essentially all of our, you know, the bulk of our caching would disappear. We absolutely depend upon caching, as I was mentioning at the beginning of this Q&A, in order for the system to operate. So name server records have TTLs, time to lives that expire, as do address records, and MX records for mail and, you know, basically that's the whole structure of the system. So it's true that disallowing name server caching would prevent this particular problem. But the whole architecture of DNS would collapse because it's near the edge as it is with all the caching we've got. And if we disallowed caching, then it just wouldn't work.

Leo: Congratulations. You cured the disease but killed the patient.

Steve: Exactly.

Leo: Well done. Bravo. There are a lot of cures like that, even in medicine. Moving on to Question 4, Olle Lindgren from Sweden wants to blend his threats. Who doesn't? You were speaking about Kaminsky's DNS findings - and we have been, quite a bit - and how it didn't break SSL. That, of course, is correct. But imagine the blended threat if an SSL certificate was issued on a vulnerable Debian system. In that case the private key corresponding to this certificate would be quite easy to guess. Because of the - I presume that's because of the Debian randomizing issue; right?

Steve: Right, where they thought they were improving it, and they just badly broke it.

Leo: The private key could be used to authenticate the server, but also to cryptographically sign content, that is, software updates. Together with DNS cache poisoning, it becomes quite a nasty threat. At least three large OpenID providers and one major bank have been using weak Debian certs. Ooh, that's bad.

Steve: So let's describe how this works because this is a great question. And we've never explicitly talked about blended threats. So this is a great opportunity. I mean, I have a perfect example. So, okay, so here's the problem. Debian was discovered some time ago, and we talked about this at the time, to have bad certificate randomization code such that, if you generated a certificate request on a Debian system, the random number in the request had very low entropy, very easy to figure out. So that request would then go to a company like GoDaddy or VeriSign, Network Solutions, anyone who issues secure certificates. They would sign your request and send it back to you. So now you have a website running that is able to accept SSL connections from users' browsers. And as we have said before, SSL is a prophylactic around spoofing because if you have a secure connection to a remote website, like for example we've used the case of PayPal, a secure connection to PayPal or even Gmail, if you use HTTPS to get to Gmail, you're able to check the certificate on the secure page and see that it was actually issued to Google.com or PayPal.com.

So here's the problem. If the site you wanted to spoof had their certificate generated by a Debian system, you could independently crack their certificate because their certificate is not secure. And in cracking their certificate you get their private key. And that's the thing it absolutely must protect. And getting their private key, because it doesn't have much entropy, it's easy to get their private key. Then you could make your own certificate, essentially; and, okay, so now you have the ability to pretend essentially to accept an SSL connection to this bad website. The problem is you don't have any way to get people to come to you. Thus, DNS spoofing. So if you spoof DNS, then anybody trying to get to the valid site instead gets your IP. They go to your server, which has a valid certificate for that site, and bingo, you've got an SSL connection. And there, I mean, nothing we know, unless you happen to have memorized the IP address and was, like, checking the IP address your browser was connecting to...

Leo: Oh, I do that all the time.

Steve: Nobody does that. So that's what a blended threat means. It's the idea of taking sort of like two small problems that individually aren't such a huge problem, but when you put them together you get something that's much more than the sum of the two. And so this would be, if you did something like this, you'd have a complete bulletproof spoofing of whatever site had a weak certificate to begin with. And you'd mix that with DNS spoofing in order to bring people to you who were saying, oh, look, I really am at PayPal because I right-clicked on my page, and I checked my security chain, and everybody signed it, and the certificate's valid, so I'm going to type in all my personal information. Oops.

Leo: Oops. That's what I do. Yikes, scary thought. Very scary thought.

Steve: So Olle was right. That's definitely a blended threat.

Leo: Trevor Harrison, Vancouver, British Columbia wonders, how do IP addresses work? One thing, Steve and Leo, you didn't mention about DNS is how does the IP address know where to go - oh, this is a great question - to find the right computer. I know it goes DNS to IP to NIC. But how does everyone in the world know the IP address of a NIC? Also, related to this, even if you have the correct domain and the correct IP address, couldn't the NIC somehow be spoofed and redirect the IP address to the wrong computer on the Internet? How does it work, Mr. Steve?

Steve: That was a great question. We sort of covered this when we were talking about routing in - probably in our first year of Security Now!. But that's what routing is. The idea is that IP addresses are assigned in blocks. You know, at Level 3 I've got an IP address for GRC.com which is, you know, starts off 4.97.142.whatever it is. Most of the people in that building or in that region of Level 3, their IP addresses also begin with 4.79.142. So the idea is that routers scattered around the Internet, they may know, for example, anything that starts with 4 goes over in this direction, and anything starts with 5 goes in that direction, anything starts with 6 goes in that direction.

And so the idea is they send the traffic sort of in the best direction they can based on what they're connected to because all routers are connected to multiple other routers. And they have a routing table whose sole job is, when a packet comes in, the routing table is checked, and that just tells it, okay, send that packet out of that connection, that interface, toward another router because the routing table says somewhere off in that direction is the IP, the actual endpoint. So when that packet arrives at that next router, it's got its own routing table which is different from the previous router's routing table because this one says, oh, these are the places I'm connected to. When packets come in, where should I send them?

And so it's just that. Essentially the packets get closer and closer to their destination. And as they do, more bits from the front of the IP address, starts off just being 4, then it's 4.79, that gets it much closer. And then 4., maybe it's between 0 and 128 goes over here, and between 129 and 255, that goes over here. So it sort of successively breaks down the IP address using more and more bits from the start of the IP address toward the back until you finally get to the router just in front of you that says, ah, I actually am connected to that IP, to that particular machine, so that's where it sends the final packet to.

Leo: It's basically how a zip code works. Right?

Steve: Yes, it's very much - zip codes are also hierarchical. And in fact that's one of the things, again, one of the brilliant things about the way the Internet was built. And Trevor last asked couldn't the NIC somehow be spoofed and redirect the IP address to the wrong computer on the Internet. Well, it's an interesting question. It's not exactly the NIC that would be spoofed, it would be the router. And of course that's another whole class of successful attacks is you do what's called BGP spoofing. BGP is the protocol that routers use for exchanging this routing information among themselves. And there have been attacks where, for example, a BGP protocol uses TCP. So it's a theoretically non-spoofable connection except that early versions of the TCP protocol did not do a very good job about randomizing their initial sequence numbers, the sort of the 32-bit number that sort of starts the numbering for all the bytes that follow. So there were weak sequence number randomization. And in fact, in the beginning, they weren't random at all. They just kind of kept going linearly upwards because you didn't want them to wrap around or that could cause some confusion. So it was sort of better not to have them completely random.

So there were attacks that actually had people, like they would contact the router themselves. That would initiate a connection that would allow them to read the router's current TCP initial sequence number, very much like the attack, when you think about it, on DNS, where you would make a query and get the current transaction ID and then send a bunch of responses with successive IDs, back in the day when those were linear. So similarly you could guess what the communication would be with another BGP server and essentially splice into their TCP connection and load false routing tables, causing packets to go where you want them to. So that's another class of attack on the Internet. Not something we've talked about before, but it's been there, it's been done, and security has been improved in order to make it harder to do.

Leo: Put that on your list of shows we would like to do is a show about routing and how it works, and a show about BGP and how it works.

Steve: Yeah.

Leo: I think both of those would be great subjects. We've never talked about routing, have we? Maybe we have.

Steve: Yeah...

Leo: That's pretty fundamental. We must have done a show on that.

Steve: I think we did a "How the Internet Works."

Leo: Yeah, okay, sure. So I refer you back to that, many moons ago, Trevor. Tim, who's hiding in Houston with his iPhone, says how secure is my phone? Steve, I've downloaded many podcasts, but whenever I get a new episode of yours I always listen to it first. As the title suggests, my question involves iPhone security. My question is simply, or maybe not so simply, how secure is the iPhone? It has Bluetooth, which I leave off unless I'm using it. I use WiFi when I'm at home and work, but I usually leave it on. It often sees other open networks. Is there any security risk to leaving the WiFi on all the time? Can someone hack the phone while it's in my pocket? For that matter, how secure is it when I'm on the 3G or edge network? I'd ask about joining one of the open networks, but I think I already know your answer to that. I guess he means open WiFi networks.

Steve: Right. Well, the answer is radio is bad.

Leo: Well, that's all a phone is.

Steve: Next question.

Leo: And anything you say here is true of any smart phone, with WiFi and stuff like that.

Steve: Exactly. I'm not singling out the iPhone at all. Radio is bad. I mean, it's necessary. I'm not saying we don't all use it. I do. I've got WiFi here. I've got it wrapped up in a WPA encryption with a passphrase from hell that I got from GRC.com/passwords. So I'm as secure as I can be. But several times during the history of this podcast we've talked about vulnerabilities in, for example, Intel's WiFi drivers. There was one not even that long ago where it was found that down in the actual packet processing, way down at the bottom, the first place the packets go when it comes hot in off the aerial, off the radio antenna, had an overflow. So before encryption and decryption, before authentication, before anything else, it was possible to simply broadcast a malformed, deliberately malicious bit of radio noise and take over a machine.

So that isn't saying anything about specific phones or PCs or anything. But we know how difficult it is to make truly, truly bug-free software. And if a bug of this nature occurs down at the very first stages of data processing on WiFi, then you've got a serious problem. And it means that, if this were found, somebody could easily be broadcasting malicious packets in a public place, taking over any, for example, if a problem were found in the iPhone, I don't know that any exists, but if there were one that were discovered, they could broadcast a malicious packet which would take over your phone.

Leo: You do have to explicitly join a WiFi network on the iPhone. Does that protect you? No.

Steve: No, no. In the case of this Intel, all you had to have was your radio on. And it was receiving stuff, and the kernel-level driver was processing it. And there was an exploit for that that was - that came out. So all you had to do was just receive the packet, even with no connection to the network because all the network connection stuff is way back further upstream, behind encryption, behind authentication, behind the TCP stack and all that. This was right down, the first place that the bits went after they turned from radio bits into electron bits. Well, I guess radio bits are electron bits. But still, into digital bits. You know, bang. There was a buffer overflow opportunity. So radio is bad. I would say turn off WiFi. Besides, it consumes power. As I understand it, the iPhone has no power to spare.

Leo: Yeah. It's one good way to save. If you're not - and I turn off "Ask to join WiFi networks," but I think you're absolutely right. Turn off WiFi, turn off 3G, you know, anything you're not using you should turn off anyway.

Steve: Yeah, and turn off Bluetooth. I mean, he is keeping Bluetooth off, and I'm glad for that because we know that there have been problems with that, as well.

Leo: But again, in general, well, I mean, there are problems with Bluetooth even if there isn't an exploit, if you just leave it open.

Steve: Yes, if you leave it discoverable.

Leo: Right. But in general these are exploits that require errors in the programming. It's not like, except for that Bluetooth discoverable issue, it's not like this is kind of part of the natural, normal nature of things.

Steve: That's correct.

Leo: Yeah. But there are always holes. That's the problem.

Steve: There always seem to be, don't there, Leo.

Leo: Yeah, they just don't - they don't go away. Let's see. Let's move on to Question 7, Bill Richardson - former governor of New Mexico? No no no no, he's in Fort Worth, Texas - was surprised by the results of his sleuthing. He says: Hi, guys. I've been a faithful listener for almost two years, enjoy the show and look forward to it every week. Thanks for an informative, insightful, deep dive into security. On with the story. Long ago I changed my static DNS entries in my Linksys home wireless to the IP addresses of OpenDNS - something I do also. So during the recent discussions of DNS vulnerability I thought I had little to worry about.

However, the other day I fat-fingered a web request and got a Charter Communications "not found" result page. Wait a minute. Confused by this, I began to research. My router settings had not changed, and my client machine's IP config showed DNS set to the router IP and the two IP addresses of OpenDNS. Yet when I went to the DoxPara test page I found my DNS server was vulnerable to DNS cache poisoning, with all the requests being sent out over a single port. Lastly, I searched ARIN, which is the - what does that stand for?

Steve: Used to know [American Registry for Internet Numbers].

Leo: Anyway, Whois, and found that of course the IP address identified by DoxPara is owned by Charter Communications. Well, that's - huh? That's interesting. I guess that's who's hosting his site. My questions are many. How could Charter intercept these? Oh. Maybe they don't own his site.

Steve: Exactly.

Leo: How could I stop them from intercepting my DNS requests and use a security-aware DNS service? What are my options? So let me recap to make sure I understand. He's using OpenDNS, so theoretically all his DNS should go through it. But for some...

Steve: Well, let's say he's configured to use OpenDNS.

Leo: Ah. And he's configured properly, it seems.

Steve: Yes.

Leo: For some reason Charter is intercepting these. What's going on?

Steve: Yes. Yes.

Leo: Yes. Yes, he said, yes.

Steve: This is a problem. This means that, okay, and think about it, DNS is over UDP, that is not connection oriented and that has no TCP connection. And his ISP could do anything it wants with his packets. So they are receiving DNS queries coming from his network, out of his router, with a destination IP on the packet of OpenDNS. They know that it's a DNS query because it's aimed at port 53, which is a universal DNS query port. All DNS servers are listening for incoming UDP and TCP traffic on their port 53. Charter, for whatever reason, perhaps for the purpose of intercepting and putting up their own advertising page, they're ignoring the destination IP of his DNS packets and changing them, rewriting them to their own. So that...

Leo: Oh, man.

Steve: ...no matter what he does, no matter what DNS configuration he uses at home, Charter is ignoring the destination IP, rewriting his packets on the fly so that they go to their DNS server. And if he, as he says, "fat-fingered," which I think means typo...

Leo: Mistyped it, yeah.

Steve: Mistyped it. Then instead of his browser being told there's nothing at that IP, he's given an IP, his browser is given an IP of a server that Charter is running that brings up their own interception page for whatever purposes they choose.

Leo: Usually advertising. VeriSign did this for a while, and it was very...

Steve: And, boy, they didn't do it for long because it really pissed people off.

Leo: A lot of ISPs do that. And they say, well, we do it as a convenience, we have a much better page. And it's true, if you don't do that, then you get that stupid Internet Explorer page saying I can't get anywhere.

Steve: Okay, well, two things. First of all, you said many ISPs do this?

Leo: Yeah.

Steve: The only way to do it is rewriting DNS.

Leo: Ah, interesting. Of course, yeah, of course.

Steve: So that means all the ISPs that are able to present their own page are doing so by preventing you from going to the DNS server you have asked for. And secondly you'll notice that he ran the test at DoxPara, and it said all your requests are coming from one port. So not only is his ISP screwing up his deliberately configured-for-security DNS, but they're aiming it at an insecure server.

Leo: They're doing a bad job.

Steve: Exactly. They've got a spoofable server, and they're forcing all of their customers to use their misconfigured spoofable server.

Leo: Is there any way to find out if your ISP is redirecting?

Steve: Not quite yet, but GRC...

Leo: A traceroute?

Steve: GRC will be telling you soon.

Leo: Oh, really. And how are you going to do that? I guess you could compare what you expect to see - hmm.

Steve: Yes. I'll be making it very clear about the IP and the reverse lookup of the IP where we see queries coming into our test. So you would instantly see that this was a Charter Communications DNS server, not OpenDNS.

Leo: Wow. That's sad. So even a traceroute you wouldn't necessarily, well, you can't trace the DNS requests, can you.

Steve: Well, a traceroute is, well, exactly. A traceroute, let's see, what would it do? There's many trace - there are many ways to do a traceroute. The idea...

Leo: ...packets, but you don't see the DNS requests go.

Steve: Well, the way a traceroute works is that you deliberately send out packets with very short TTLs, times to live on the packets themselves. And you incrementally increase the TTL. So the packet hits the first server. It decrements the TTL, which we have sent out as 1, decrements it to 0. Which prohibits it from forwarding it to its destination. Instead, it sends it back to you with its IP as the source of that returned packet. You receive it, and inside that is the beginning of the packet you actually sent. So you know what you sent, and you know where it bounced. Then you send a packet out with a TTL of 2. So it goes to the first server, then decrements the TTL to 1, forwards it to the second router - I'm sorry. The first router, that decrements the TTL from 2 to 1, which forwards it to the second router, which decrements it from 1 to 0. That prohibits it from forwarding the packet. So it bounced it back to you. And that's how you're able to build a trace of the direction the packets are taking and the sequence of routers that they visit. So what would happen would be this thing would bounce its way over to Charter and end because the packet would have a TTL, probably pretty low. Since Charter is your own ISP, the packets you generated would hit the DNS server quickly as opposed to heading off across the Internet over many more hops to get out to, for example, the OpenDNS server. So a traceroute, if you knew to do it, would be suspicious, but you'd have to suspect that first.

Leo: There's a UNIX command called "dig" that does DNS diagnostics.

Steve: Oh, it's worth noting also, Leo, that you would specifically have to do - you'd have to have the power to do a traceroute of DNS traffic. That is, most traceroutes will just use, for example, ICMP packets, which would not be redirected. So ICMP would go on its merry way, heading off for OpenDNS. Only a UDP packet bound for port 53 that you were using as the traceroute payload would have its destination IP rewritten and then get rewritten and aimed at Charter's DNS server. So it'd be a pretty sophisticated thing to do. But certainly doable.

Leo: Interesting.

Steve: Easier to go to GRC once I get that service up.

Leo: I can't wait till you get that up. That's going to be fantastic. That's a very good use for that. That's not - that wasn't your primary intention, though, was it?

Steve: Well, no no no, I've got...

Leo: It's a side effect.

Steve: ...a much better statistical system built than anyone has before. And it's coming alive. And I'm excited. And this is an example of why this problem has not gone away. I decided to invest in the time of creating a permanent facility, much as I have for ShieldsUP!, for DNS because these kinds of bogus things can happen where people can get into your DNS business and end up rerouting you to a spoofable server, even though you've deliberately configured one not to be.

Leo: Wow. When I run that better DNS search that you gave us last time, I think it was snipurl.com/...

Steve: The one that goes to OARC's site.

Leo: Yeah, dnstest, I think. Snipurl.com/dnstest. But I would get results back from both Comcast and OpenDNS.

Steve: And you're a Comcast user?

Leo: Yeah.

Steve: What must be happening, there are a number of things have happened. I've got a preliminary test up that the people in our newsgroups have been using. And, for example, some of them have received responses from as many as 15 different name servers.

Leo: Yeah, I got two different - three different name servers responding.

Steve: Yeah. We're going to spend an episode talking about this once I have this up so people will be able to see what's happening. There are all kinds of things that are going on. For example, some of our early testers were reporting, wait a minute, I'm seeing IPs that are not the ones that I configured.

Leo: Right.

Steve: It's like, well, yeah, because what's happening is those, the IPs showing up in the test, are the IPs of the server that actually issued the query out onto the public Internet. But it can also often be that you are - that the IPs you configure, for example, your ISP's IPs, they could go to a pair of servers that are forwarding the requests to a pool of servers in some sort of a round robin fashion.

Leo: That is what's happening because I go to San Fran Comcast and then go to Salt Lake City Comcast.

Steve: Okay.

Leo: So they have some pool somewhere. But I also go to OpenDNS, so I'm very confused.

Steve: We will resolve your confusion, my son.

Leo: I can't wait, I can't wait. That's going to be a very valuable tool. Let's get a listener from Heidelberg. This will be fun. His name is David. He "hosts" an interesting idea, a little pun from Steve Gibson. I've been listening to the podcasts for several months, and I love them. After hearing Peter Chase's idea of avoiding spoofable DNS by using IP addresses in browser bookmarks instead of domain names - which is something we've talked about a little earlier - I think it would be better to use the hosts file. However, if Internet site IP addresses changed, that would pose a problem. So why couldn't the hosts file be updated automatically? Suddenly sounds like he's setting up a DNS server. It could help block ads, map domain names to IP addresses, throw malware off the track. An application/service or other automated method that updated the hosts file could be very useful - perhaps dangerous - and better than something that will go through my Firefox favorites or bookmarks, say, or my IE favorites, and just replace them with all the IP addresses. It's not a silver bullet. I find it interesting, and I would like to hear your comments. Keep up the great work. I mean, this is how DNS was born, basically; right?

Steve: Well, yeah, it is. I mean, it is the case, I mean, I could see an interesting piece of software that is sort of aware of spoofability problems, that periodically it opens up your hosts file, reads the hosts file, and goes out to verify and possibly update any entries you have in your hosts file to keep it synchronized with reality. Basically it'd be like setting up a little sort of custom DNS interception system where you put www.doubleclick.net in your hosts file. And the question is, okay, the problem is, if DoubleClick.net changes their IP, then your hosts file is obsolete. So imagine a little tool which would itself be a DNS resolver, driven by the current contents of your hosts file, whose job is to keep it current. I think that's kind of cool.

Leo: Yeah. But isn't that, I mean, isn't that kind of what DNS is doing is it's keeping a master list and a matching list? In fact, that's what you do with OpenDNS. You can have it block porn sites if you want.

Steve: Right. So all you're really doing is you'd be - the benefit of this would be, if you don't trust an external source of DNS, but somebody wrote something that was, like, really anti-spoof, I mean, it is absolutely possible now to write an anti-spoofing, an anti-spoofable DNS server, simply by issuing a query and seeing how many responses you get, as we said earlier. I mean, so it's - or to, like, start, if you're just you, and you've got a small hosts file, there's no reason not to start at the root servers, who are based on IP addresses. They have domain names. But, you know, you can't look them up unless you already know them.

So you could imagine a little DNS resolver that marches down from the root servers to the com servers to the domain servers to the machine level, and itself follows the DNS step by step, which makes it unspoofable. It's not caching. It's getting authoritative information at every step in the DNS chain. And that would allow you to have an absolutely unspoofable hosts file of sites that you cared about that your system would refer to first and nobody could screw with them. And they'd be kept current.

Leo: Very clever.

Steve: If I had more time, I'd write one. But I'm a little busy right now.

Leo: You're busy doing other things. Let's see, Question 9 is from Derek O'Harrow in Reading, England. He suggests another blended threat. We've got to do something on blended threats. This is very interesting. Hi, Steve and Leo. In Security Now! Episode 155 you talked about SSL and the fact that it cannot be faked. So sites like PayPal can be relied upon because they force you to use SSL and a certificate. However, in a previous Security Now! you talked about the root certificate authorities and how many of them there are. Wouldn't it be possible for Mr. Russian Hacker to spoof a domain and then, using one of those other more dubious root certificate authorities, create SSL server certificates for PayPal.com but with a different signing authority? In this way, couldn't a hacker pretend to be even an SSL-secured site? And wouldn't the web browser client happily accept this without question, eh, eh? Thanks for the fantastic podcasts and for ensuring we never miss a week, including earthquakes. Well, we've talked kind of about this, haven't we.

Steve: Yeah. This is actually another one of the questions that sort of misunderstands the way SSL works. We have, as I said, I think it was a week or two ago, I'm going to do just a whole podcast on SSL, how the protocol works, to make it very clear. Because we've talked about the certificate side of SSL, but never really about the way the protocol works. So what Derek was asking was couldn't you spoof the domain of the certificate authority, which is really the same question that was asked earlier in this podcast, and wouldn't that then allow an otherwise invalid certificate to be seen as valid?

Okay, well, you don't have to spoof the domain of the certificate authority because your browser already has a huge number of certificate authorities. In fact, that's been my anguish is how many there are, and we just trust every single one of them because even one of them issuing a certificate maliciously or mistakenly compromises, well, compromises our ability to trust the real owner of that certificate. So if there were a dubious root certificate authority whose certificate was already in our browser, as so many of them are - not dubious root certificate authorities, hopefully, but legitimate ones - and that dubious authority issued www.paypal.com certificate to Mr. Russian Hacker, then yes, now he's got a maliciously issued or mistakenly issued or somehow he got himself a valid certificate for PayPal.com.

Normally that's not a problem. But again, in threat blending we take not a big problem, I mean, not good, but not a big problem, and the DNS spoofing, which certainly can be a problem. We put them together, and we end up with a much bigger problem because that would then allow Mr. Russian Hacker to set up secure connections to www.paypal.com. The browser would connect happily. Presumably, if it was one of those extended validation certificates, all the green lights would turn on, too.

Leo: See, I was going to ask you about that. So the EV certificate does not fix this problem.

Steve: No, it does not. It's just harder to get one, hopefully making it more difficult for Mr. Russian Hacker to get one.

Leo: Right. Yeah, because he has to provide all sorts of information, personal information and whatever, fingerprints and stuff like that. He's not going to do that, I would think.

Steve: Well, I'll tell you. I mean, I had an experience, very good experience, actually, with GoDaddy two weeks ago. I needed a wildcard certificate for the first time ever for my first...

Leo: What's that?

Steve: That's like *.GRC.com.

Leo: Ah, okay.

Steve: That is to accept secure connections. Because my first approach at doing this DNS spoofability test that I'm building, it used a whole bunch of browser queries over SSL because I wanted people to be able to use SSL connections with GRC specifically so that they could avoid spoofability problems. You know, we talked about, okay, well, how does it make sense to go to a DNS test if the DNS server you're using might be broken and spoofed, and then the test would be spoofed. So the idea was, okay, we're going to let people come in by IP or by SSL. And in fact I was going to force people to switch over to SSL and not allow them to have a non-SSL connection to GRC. But if I then had lots of assets that the web page was pulling, those had to be SSL. And the way this worked was it pulled them all from different machines at GRC.com in order to force DNS lookups.

So anyway, I needed a wildcard certificate, *.GRC.com. Well, the good news was I had it in five minutes. GoDaddy is very nice, was very quick. But the extent of their validation was sending email to the email address registered on my domain. And this was all automated. So they sent it to my Whois email address, which came to me, and I clicked a link that I received in the email, that I was expecting. It went back to them, verified it, I got an email certificate.

Now, not long ago, in the last couple weeks, we have heard of Whois database entries getting changed. Remember, that was just in the last month or so there was a Whois entry change. Which means all that Mr. Russian Hacker would need to do with an automated certificate issuance like GoDaddy has is get his email address into the domain record of the person he wants to spoof. And then he says, hey, I'm PayPal.com. They verify through automation, send email to that record, he gets a certificate, bang.

Leo: Wow. But that's the non-EV certificate, and that's how it's been all along; right? I mean, that's why we need an EV certificate.

Steve: Because it certainly is - we hope it is not going to be a five-minute automated email loop verification. Otherwise it's back to the drawing board.

Leo: Right. Matthew is an intense listener in Fresno, California who's been thinking about DNS spoofing: Steve and Leo, I was listening to Episode 156 while I was driving to work, and I heard Steve mention that the DNS vulnerability cannot spoof your email logon. Got me thinking. Our listeners are always thinking. One, if I was a hacker, and I was spoofing, say, https://mail.gmail.com, couldn't I also spoof VeriSign.com and have my fake Gmail certificate authenticate to a fake VeriSign, but authenticate correctly if I created a fake Gmail cert from the fake VeriSign? Two, if I were a hacker, and I don't care about the contents of your email, couldn't I just redirect the logon pages of the webmail sites and capture the username and password? As a hacker, I don't care if I can get your email, I just want your credentials. Browsing the most popular webmail services I've found that going to http://mail.google.com and http://mail.yahoo.com redirects you to their SSL site for authentication. But going to http://hotmail.com does not. So if I could duplicate the look of the Hotmail page and redirect the DNS traffic, I can capture more than enough information. Thanks for a great podcast. I listen intensely to it every week. So he's thinking. Is he thinking right?

Steve: Well, again, this shows that there was a little confusion in the way certificate chains are authenticated. And that is that many of our listeners, that's why I wanted to really pound this point home, many of our listeners were believing that there was, during SSL authentication, that there was some sort of communication back to the certificate authority. And so I want to just make sure everyone gets it that that is not the case. In the last couple months, for example, Microsoft sent out to Windows an SSL root certificate update package. And so that's the only way these things get into our systems is that they come in with the browser or as part of a secured, signed, authenticated update. So those root certificates then statically authenticate any certificate from a remote site with no further communication to Microsoft or VeriSign or GoDaddy or anybody else. So that part is not the case.

However, for example, his discovery that Hotmail.com does not use secure authentication, that's important because it means that they would be a perfect target for a DNS spoofing attack where somebody would set up a fake Hotmail account and acquire the credentials of everybody trying to log into their Hotmail account while that spoof was in place because he would not then need a Hotmail.com SSL certificate, which is more difficult to get. Unfortunately, as we learned from the prior question, not as difficult as we would like it to be. But they would just leave it over a standard HTTP connection and happily collect username and passwords. And in fact could turn around and open a connection to Hotmail and be a man-in-the-middle attack, essentially, capturing that, grabbing the Hotmail page that the user expects to come up, and forwarding that back. And we've talked about browser-based man-in-the-middle attacks before using, as long as you don't need security, that is, SSL, DNS spoofing is a perfect entre to a man-in-the-middle attack.

Leo: Wow. So you're saying it is doable, but it's a little more difficult than one would imagine.

Steve: Well, it's another example of why Gmail and Yahoo!, that do redirect nonsecure authentication to a secure authentication mode...

Leo: Right, that's why it's the way to do it, yeah.

Steve: They're doing the right thing, yes. And assuming that Matthew is right, and Hotmail doesn't, that's really bad on Hotmail's part.

Leo: Kind of odd.

Steve: Because of just all kinds of problems.

Leo: Isn't it possible that it's one of those situations where your...

Steve: If you had a cookie, for example, he might have a cookie on his machine. And if it recognizes in your incoming connection a cookie which has not yet expired, it might treat you with less security. Of course the problem there is you're not over SSL, so your cookies can be sniffed. Because cookie snarfing is another problem.

Leo: See, now, I'm looking at this. I just logged in. So first I went to HTTP Hotmail blah blah blah, logged in. Now, presumably that login, even though it doesn't say HTTPS, is a secure login.

Steve: No.

Leo: No?

Steve: I mean, it would have to say HTTPS.

Leo: But remember we have those pages where the submit is secure but the page itself is not?

Steve: The page you get to would have to be secure.

Leo: Okay. The page you get to would have to be secure. But now I'm in my mailbox, and it is HTTP. It's not S.

Steve: Sounds bad, Leo. I hope someone didn't just grab your username and ID on the fly.

Leo: Well, I can view the source, right, and see if it's a secure submission.

Steve: Yes, if you go back to the submission page and then, like, just do a search for HTTPS, and you ought to find it in - you ought to find the - or it could be using JavaScript, who knows what...

Leo: It is, you know, and it's all obscured because it's one big long JavaScript.

Steve: Then the way to do it would be to use a browser monitor tool, a wire...

Leo: See, but no user's going to do that. I mean, you're just going to assume it's Microsoft, it must be secure.

Steve: Sounds bad.

Leo: All right, let me try again. So I'm on a regular HTTP page. You're saying when I press the - give them my password and credentials, and I give them the sign-in button, I should then be dumped into an HTTPS page.

Steve: Oh, and if you hover your mouse over the button, does the URL of that button show up down below in the browser bar?

Leo: No.

Steve: Okay.

Leo: And I am now - oops, I gave it the wrong address. And now, okay, I gave the wrong password, and it did go to an HTTPS page. So presumably...

Steve: It does sound like they've got some mixed in there, yes.

Leo: I mean, c'mon, this is Microsoft. They couldn't be that...

Steve: Okay.

Leo: Forget I said anything. Number 11. Isaac Church in Loomis, California worries that his characters are being ignored: Hi, Steve. I'm a Wells Fargo customer, and as I happened to notice, they accept passwords that technically are not correct. As long as the first part of the password is correct, the characters that follow are ignored.

Steve: Can you believe this?

Leo: But it's well - okay. This might not be a serious security threat. But as a programmer it bothers me since wouldn't it make it easier for an attacker to correctly guess my password? Wouldn't it? I'm also curious if other banks have this issue. Thanks for the great show and software. So, what, they have, like, a seven-letter password, and if you put in 12 it just ignores the last five?

Steve: That's what it sounds like. It's like, oh, my goodness. Now, okay, this means a bunch of things. This means that they're not doing a hash, or if they are, they're only hashing on the first X characters. It probably means they're storing the password, and that they're only paying attention to - okay. I don't know if he's added characters after he defined his password or if he used a longer password and then he changed the end of it and they didn't care because they're only recording or checking the first N number of characters. Now, that's really disturbing.

Leo: That is extremely disturbing.

Steve: And so, and of course the true threat of security, first of all, anybody who's doing this you wonder, okay, who wrote this, and do I trust anything else about the site if this is how they start off? But then of course the issue, the true issue of security is how big is N? If they're only using the first N characters, where is that threshold? Is that eight? Is that nine? Is that, as you were suggesting, seven? Which we know really quickly becomes too short for security. So, you know, this is very disturbing. I almost hesitated making this public. But I thought, well, our listeners need to know, especially if they're Wells Fargo customers. Maybe some of them can start messing around with their passwords and seeing what N is and definitely complain to somebody. That's bad.

Leo: That's just bizarre.

Steve: Yeah.

Leo: That is just bizarre. We're going to get to our 12th question, in which Russell says, "I am not a pirate," in just a second. It's nice, you know, that when we talk about security all the time and how badly implemented it is, to know that there are people out there who are like you, that just take it really seriously and do it right. And so I just - I'm just glad that they're part of the show.

Steve: Yeah, I met those guys at the...

Leo: They're cool, aren't they.

Steve: ...RSA conference. They were seriously UNIX people.

Leo: You want somebody who is really, you know, you don't want somebody who's, like, the guy who wrote the Wells Fargo login. You want somebody who's, like, really paying attention here.

Steve: Yeah, we got most of it right, and we don't want customers complaining that they can't log in, so a guess is good enough.

Leo: You've just got to wonder.

Steve: He got started on the right foot. He just kind of fell off the track halfway through his password. So we'll let him have that one.

Leo: He's the code monkey. He's going to let the manager write the login page.

Steve: Come on in and manage all your money.

Leo: Question 12, Russell McOrmond in Ontario, he's in Ottawa, in Canada, the capital, the nation's capital. He says: I am not a pirate. In the last Q&A episode someone mentioned that SpinRite was the only program they had bought in a long time. Leo commented that this person must be a pirate. Well, SpinRite is also the only program I have bought in more than 15 years, and I am not a pirate . Instead I'm a user and commercial support person for Free/Libre and Open Source Software. FLOSS, baby. I've purchased CDs and manuals over the years. But given that licensing is royalty-free, I have not "bought" any of this software. All of it is perfectly legal. As it happens, I'm the policy coordinator for CLUE, Canada's Association for Open Source. It's at Linux.ca. Thank you, Russell. One of the hardest messages to get across to politicians is that charging royalties for software is not a necessity, but only one narrow business model among many. I just thought I'd add this. Thanks for the great security show. Russell, you are right on. You are right on, and I apologize to open source advocates everywhere who have not purchased anything but Security Now! in 15 years.

Steve: Well, SpinRite. And I really - I thought that was neat. I really appreciate that, you know...

Leo: That he bought that.

Steve: Bought SpinRite.

Leo: There's no open source analog, that's why.

Steve: Yup.

Leo: The one and the only. And Steve, will you do us this favor, put it in your will that the day you pass on, that SpinRite goes open source.

Steve: Okay, but it's all Assembly language.

Leo: That's okay. I think there's somebody out there who could figure it out.

Steve: I think that's true.

Leo: In fact, I think there's somebody out there who can't wait to get his hands on your source code, who's dying to look at your source code.

Steve: Actually, SpinRite is so stable and generally my major versions have many years...

Leo: Few and far between, yeah, yeah.

Steve: And GRC's running so smoothly now that I could even see, after I'm in my grave, Greg and Sue could really...

Leo: Just keep it going.

Steve: ...easily continue on, and they ought to because, you know...

Leo: You know, what's going to happen at some point, and we're seeing it already, SSD hard drives, their hard drives are going to change in such a way that SpinRite doesn't make sense anymore.

Steve: True. True.

Leo: I mean, if everybody goes SSD, then we won't need SpinRite.

Steve: Yeah, the drive technology is maturing at such a rate that it's going to be tough for solid state to catch up. But we know ultimately it will. And, I mean, look at the prices of hard drives, Leo. It's just, I mean, it must be that the manufacturers are saying, uh-oh, we have such an investment in spinning magnetic platters that we want to make sure people keep buying these, so we're going to sell you a terabyte for $125. It's like, oh. Well, in that case, that's a good deal.

Leo: I just bought the new VelociRaptors, two of them, for our ultimate gaming machine. They're so cool. They're so tiny. I just, it's amazing. And they're 300GB, this big, you know? I mean, it's mind-boggling.

Steve: And you're going to run SpinRite on each of them...

Leo: Before...

Steve: ...at Level 4 before you put them in.

Leo: And we're going to ship a copy of SpinRite with it so that they will never have trouble. And I just bought, for 98 bucks, 500GB drives, these eSATA drives. 98 bucks for 500 gigs.

Steve: Wow, wow.

Leo: Unbelievable.

Steve: Fast, nice. And what manufacturer?

Leo: I don't know.

Steve: But there they were.

Leo: It's all a commodity. I don't know.

Steve: Good price. Good price.

Leo: I don't know what's inside. You know, the enclosure says "Cavalry." But I don't know what kind of drive. Nowadays it's all kind of commodity.

Steve: Yeah, just plug it in, and it goes. It'll be Western Digital in the...

Leo: Yeah, almost certainly Western Digital, yeah.

Steve: They're still the rock-bottom price.

Leo: Yeah, and good drives, though; right? Not bad drives. I know you like Hitachi. But WD is fine. Well, we're out of time, I think, and out of questions, so I think maybe time to wrap this up. Next week, Steve, what do we want to talk about?

Steve: Next week we're going to discuss the other really interesting presentation from Black Hat which was this supposed complete defeat of all the much-heralded security benefits of Vista.

Leo: Oh, boy. Vista, is it safe.

Steve: Address Space Layout Randomization, ASLR; and DEP, Data Execution Prevention. The question is, how easy is it just to bypass them? And apparently the Vista security bypass is a reality.

Leo: It's so good to have somebody like Steve in our back pocket who can look through these presentations with a technical eye and judge them on their merit. Because otherwise, you know, I mean, The New York Times is not going to know. They're just going to report that this guy says it. They're not going to know enough to figure it out. And frankly, I can't look at it and figure it out. So thank you, Steve.

Steve: Yeah, it's a 54-page report. And I'll be doing my homework between now and then. But I will figure it out, and I'll give our listeners the complete rundown.

Leo: Thank you so much. Steve Gibson is at GRC.com. That's where you'll find 16KB versions of this for the bandwidth impaired, the low-quality but still audible versions.

Steve: The low quality.

Leo: You can find a crappy version that's there. No, no, but it's there for people, it's a small, small, small file. And also transcripts, which are very high quality thanks to Elaine. And she edits out all my ums and uh, hey...

Steve: But she leaves in my [trumpeting].

Leo: How does she spell that?

Steve: I think she says [Steve plays trumpet]. Doo-to-doo.

Leo: She was a court reporter. I guess when you're a court reporter you have to do those things. [For the record, I studied court reporting, never got my license. Elaine]

Steve: Oh, and she does, like, excruciatingly detailed biomedical reporting, I mean, stuff where every single syllable matters. [Again for the record, that's transcribe, not report. Elaine]

Leo: And the doctor goes doo-to-doo, you'd better get it in there.

Steve: And, you know, she gets a copy of the Q&A because she insists on spelling everyone's name right, you know, all of our listeners.

Leo: [Trumpeting].

Steve: I wonder what she does when you do one of your accents. Leo says...

Leo: In Russian.

Steve: ...[in bad Italian accent]...

Leo: Bad Italian accent.

Steve: Or actually I have to say my tech support guy, Greg, who listens to Security Now!, he says, you know, "Leo is really good with those accents. He ought to just keep it up throughout the entire question."

Leo: No.

Steve: And there were several here where I noticed that you started off...

Leo: No, I do a little bit, just a flavor.

Steve: Sort of fade out pretty quickly, so.

Leo: Flavor. A little flavor.

Steve: Greg's request has been heard, so...

Leo: All right.

Steve: ...I did my job.

Leo: I get equal, maybe more requests saying knock it off, Laporte, it's not funny.

Steve: You've struck a nice balance, then.

Leo: Don't forget, if you go to GRC.com, to get your copy of SpinRite, the ultimate, the only, the one-and-only, the only program worth paying for, hard drive maintenance and recovery utility, and all of Steve's great free stuff like ShieldsUP! and all that great stuff. And soon to come, some neat new stuff.

Steve: New cool stuff.

Leo: Steve, we will see you next week on Security Now!.

Steve: Talk to you then, my friend.