SECURITY NOW!

Transcript of Episode #171

Listener Feedback Q&A #54

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-171.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-171-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 171 for November 20, 2008: Your Questions, Steve's Answers. This show is brought to you by listeners like you and your contributions. We couldn't do it without you. Thanks so much.

It's time for Security Now! the show that tells you how to stay safe on the Internet. And Mr. Gibson is here. He's kind of like, you know, the crossing guard. He's got his little crisscross orange shirt on, and his flag saying "Do not cross here. It is now safe to cross."

Steve Gibson: Proceed with caution.

Leo: Proceed with caution. That should be the name of the show, Proceed With Caution.

Steve: That would be a great name for the show, actually, yeah.

Leo: Yeah. Because there's nothing you can do. I mean, except proceed with caution.

Steve: Yeah. Basically that's what we're helping to explain to people is where the pitfalls are, where the potholes are, and how to behave themselves, so...

Leo: Proceed with caution. Well, we're going to do - we have - today is our Q&A day. I know people kind of are a little confused because we originally were doing this on odd episodes, I mean even episodes. And then we had an emergency to cover the DNS poisoning issue.

Steve: And we changed our parity.

Leo: We changed our parity.

Steve: Went from even parity to odd parity.

Leo: Typical engineer.

Steve: Something will happen. We'll have another emergency, and we'll switch back over.

Leo: Right.

Steve: So just to keep everybody off balance.

Leo: So even though it's Episode 171, it is Listener Feedback #54. We've got a dozen questions, including some really good PayPal tips and tricks. Should be good. And we're going to get to the latest security news and any addendum or errata from previous episodes. Let's get the latest security news.

Steve: Well, a bunch of stuff has happened. One follow-up on DNS. I'm still very much up to my elbows in DNS stuff. I hope, probably, I'm guessing three weeks from now - we've got the author of Sandboxie on next week, then a Q&A. Probably the week after, I think by then I'll be ready to unveil the work I've been doing for, like, the last five months on the whole DNS spoofability stuff, to allow people just to check to make sure that the DNS servers they're using cannot be easily spoofed. And in the process we discovered that we're able to crash a bunch of routers, which is a concern because, as we know, what starts off as a crash often can be evolved into a remote exploit. So I've also got a router crash test that has evolved out of this, so people can see whether their router is crashable and bug their router manufacturers, if it is, to update the firmware, to fix whatever might be an as-yet unidentified buffer overrun in a router. And of course that's a problem because if someone sent your router a packet that could take it over, that's a bad exploit.

Leo: Yeah, no kidding.

Steve: Anyway, all kinds of things have happened. An outfit called the Measurement Factory received a commission to evaluate the status, the current, like to take a snapshot of the current status of the DNS servers on the Internet. They grabbed the routing tables from some central routers and used that to determine how many IPs were actively being routed. And it was a little shy of two billion, which is interesting, since as we know there's a total of 4.3, actually it's 4.29, billion IPs potentially possible, although some are permanently removed, like anything beginning with 10. We know that the so-called "10-dot" IP range is reserved for private networks, so not publicly routable.

But this is something that Mark Thompson told me years ago that surprised me, is that here everyone's all worried about IP space depletion and running out of IPs. But still only half of the 'Net actually has publicly routable IPs. There are things like, well, for example, remember that Hamachi is using the 5-dot space because no one is using it. It's not assigned to anything. It's just, I mean, and that's a huge chunk of IP space that is still sitting there because typically they've been allocated to organizations that don't want to let them go because it's valuable, even though they're not using them yet.

So this outfit looked at the routing tables, found the 1.9 billion IPs that were routed, took a random, I think it was 10 percent sample of those, and sent out probes, DNS probes, just at random to find random DNS servers and check to see how they were performing. They ended up determining, based on their sample, their statistical sample, they estimate that there are 11 - I'm sorry, yes, 11,900,000 name servers publicly accessible on the Internet. So just shy of 12 million, 11.9 million name servers on the 'Net. Of those, 4.3 were open resolvers, meaning that you could send them a query, and they would do the work of looking up an IP address for you and send you a response. And of course those are what - that's the source of the vulnerability. So 4.3 of those are open. Of those, they found that 25 percent of the servers that were open resolvers, that would do open recursion, have not yet been patched.

Leo: 25 percent.

Steve: Yes, one out of four servers are still vulnerable, I mean, really vulnerable to this Kaminsky-style attack, meaning that when you ask them a query or you ask them a question, they launch the query out of a fixed port or an incrementing port, something which is easily predictable. And that's all you need in order to be able to fake the answers that that name server is looking for and poison its cache such that anybody else using it will then be redirected to a malicious site.

Leo: I'm going to say something surprising. That's a lot. I mean...

Steve: Yes, that's a lot.

Leo: I mean the other way around. I'm surprised 75 percent are patched.

Steve: Well, remember, a lot of news was made of this. Now, what was really interesting was that, as part of this study, a survey was sent out. And 45 percent of administrators responding to the survey said that they lack, quote, "lack the necessary resources to address the DNS vulnerability." Whatever that means.

Leo: We don't know.

Steve: I don't have time. I'm busy doing other things.

Leo: The guy who set it up doesn't work here anymore, and I have no idea how to fix it.

Steve: Exactly. It's a black box in the corner. It's got cobwebs on it. We're afraid if we touch it, we'll break it, and then we'll never get it going again. So we're just leaving it alone. 30 percent of the administrators responding said they do not know enough about DNS to make the required change. Which is encouraging - not. And then in their analysis they did fingerprinting. That's one of the technologies I've added to my test also. You'll be able to see what the make, model, and version of the DNS servers you're using are, among a whole bunch of other really cool things. So they found that 90 percent of the DNS servers are now running BIND 9, which is the latest and greatest. And they also, from a sample they'd made a year ago, they learned that there had been a significant decrease in Microsoft's DNS server, which frankly pleased everyone because it's not really very secure. And they did find...

Leo: That's the one you use; right?

Steve: No no no no. I'm using BIND on FreeBSD UNIX.

Leo: Oh, for some reason I thought you were using IIS on a Windows server.

Steve: Well, IIS for web serving, but not for DNS.

Leo: Not for DNS, okay.

Steve: Yeah, for many reasons I need real DNS.

Leo: What's the name of Microsoft's DNS program?

Steve: It's just DNS Service. And when you install, for example, the server version of Windows 2000 or 2003, it's one of the options you can just install. And, yeah. I actually, I had it installed here for a while, just sort of I'm curious about it, to see how it worked. And, you know, it's like I didn't see anything wrong with it. But I wanted to go with real BIND. And I'm running BIND 9 also.

And then the last thing that was of particular interest that they discovered was that there was a distressingly low adoption rate of DNSSEC, the DNS security extensions, which we've talked about, which really do solve these problems. Oh, that's another thing that I will tell you when you use my soon-to-be-unveiled test, is whether your ISP's DNS servers are supporting DNS security. So you have to sort of get a sense for where your ISP stands relative to others. And also I have a DNS benchmark that will allow you to determine the speed of yours versus, for example, OpenDNS, so you could see whether your use of the Internet could be faster by switching to some other alternative DNS server.

Leo: Oh, that's good, that's nice.

Steve: Cool stuff, yeah. Many, as I was running through the Q&A for this - I'm sorry, as I was running through all of the user submissions of stuff at GRC.com/feedback, I ran across a whole bunch of people that were wanting to bring to my attention that Visa over in Europe, in the EU, has released a new card that contains a built-in keypad and eInk display that produces a one-time code for part of their, Visa's rollout of a next-generation security technology.

Leo: Hallelujah. That's like the VeriSign card.

Steve: It's, yeah, in fact it looks like very much - I would imagine it comes from the same manufacturer.

Leo: But it's a Visa card? It's an actual charge card?

Steve: It's got a mag strip on it. And so you can hand it to, you know, the server in the restaurant who can scan it just like a regular credit card. But it also has this challenge/response technology.

Leo: Fantastic.

Steve: Yeah. So it's not something that's universally available yet. It does require substantial backend technology in order to support it, in order to do the challenge authentication handshake with the card. But it's really a very comforting sign because it demonstrates that we're moving forward. And the only reason you would do something like this is where you have a so-called "card not present" purchase, where you're doing an Internet purchase, and rather than having the physical card present, you're reading it out over the phone or over a web form. And so this allows that kind of technology. So I wanted to acknowledge all the people that wrote to tell me about it. And we'll certainly be keeping our eye on that.

There was a bunch of security news for the week. Apple has released a big update to Safari which fixed 11 vulnerabilities. Unfortunately there have been many complaints of crashes afterwards. So apparently there's some little bug of some sort, doesn't affect everybody, but there's been a high incidence of complaints of Safari crashing after this update was released. It's 3.2, and it fixed, as I mentioned, 11 security flaws. Most of them were for the Windows version of Safari, but a couple were for the Mac OS X version.

Leo: And the reports of crashing, are they on both platforms? Or are they on Windows mostly, or...

Steve: Oh, predominantly Windows. I don't know of any over on the Mac OS X platform.

Leo: Yeah, because I haven't seen any, haven't had any problems. But I don't use Safari on Windows.

Steve: Also I got an update of my use of Firefox to 3.0.4. I would imagine anybody using Firefox would have had the same thing saying, oh, we've got a new version of Firefox, you have to shut down and restart Firefox. But I wanted to make sure everyone knew. It was also a big update, fixed 11 security, I'm sorry, 12 security flaws. Several were remote execution vulnerabilities. So you'll want to make sure you get that fixed. Half of them, six of them, were rated critical. And those that weren't, that were not remote code execution, were technically local denial of service, meaning they would crash your machine, which is not good.

Chrome, Google's browser that's still in beta, expected to be in beta for a long time, has moved forward also. There was a file-stealing hole that was found in Chrome which has been fixed. The latest version apparently is a development release that may not yet be available to typical users. And they're at 0.4.154.18. That's the latest and greatest from Google. And they've added some features. There's now a bookmark manager in it, and they've reworked their popup blocker to be more effective. So that continues to move forward.

In some dialogue I saw online there were some independent researchers saying, you know, this kind of stuff reminds us that a browser probably needs about a year of gestation before it's the kind of thing you want to jump on. And until then you want to run it, like on an experimental box or in a secure virtual environment of some sort because there's just - it takes a while to nail all of the debris and bugs out of anything, you know, out of a big, aggressive chunk of code like this.

Leo: Well, and the browser is such an exposed surface to hackers. That's the most critical application; right?

Steve: Yes, it is now the target. Speaking of which, I picked up a little interesting tidbit of news that I knew our listeners would appreciate. NebuAd, the heinous install-their-equipment-in-ISPs'-facilities that we've talked about extensively, has been sued now by 12, or at least a dozen subscribers. NebuAd and six ISPs that were using it but have since dropped it like a hot potato have been sued. The suit asks for $5 million in damages and requests that it be moved to class action status. So and they're alleging that the web surfing habit tracking technology and the companies that used it without customers' knowledge violated anti-wiretapping statutes. And people who - legal experts who've looked at the suit believe that NebuAd is in bad trouble, that is, that it absolutely does violate anti-wiretapping statutes.

Leo: It's an illegal application.

Steve: Yes, it's fundamentally illegal, certainly within some areas, to do this. So the good news is the other guys, Phorm and Front Porch and anybody else who was thinking this was a good idea, they're certainly aware of this, too. And with any luck they will be staying far, far away from this kind of really invasive technology.

Leo: And they don't do anything so differently that it wouldn't be any more legal, I mean, it's the same idea. You're using somebody's computer to kind of spy on them, or to modify their content.

Steve: Yeah, exactly, yes. And in some cases you're modifying the content on the fly. But in any event, you are definitely reading the pages that they are requesting, and you're using the content of their pages and their search requests and all of that in order to target ads at them individually. I mean, it's just - it's a bad privacy problem.

Leo: Well, sorry it's illegal, guys. Guess you'll have to stop.

Steve: Oh, darn.

Leo: Oh, darn.

Steve: But I wanted to share a really kind of a fun and wacky note from a listener of ours. I hadn't chosen yet anything to share with our listeners for SpinRite this week, and I encountered one as I was reading through all of the input from our listeners. The subject sort of caught my attention because the subject was "Thank you. SpinRite got me fired, and I do mean thank you."

Leo: Okay.

Steve: So he asked to be anonymous. He sent me his name, but so he's calling himself "Scooby Drew." He says, "I did tech support for a company that I will not mention because I am in the process of getting a season job with them. We had a computer crash that had lots of customer information on it, including credit card numbers, addresses, and other personal information. They left the computer out in front of the building. Even if our IT guy said, quote, 'They can't do anything with it other than scrap parts,' I wanted to see if there was more to it than that. So, being curious about SpinRite, on my lunch break I picked it up and took it home. Now..."

Leo: It's scrap parts. It's scrap parts.

Steve: Exactly. "Now, I'll be honest." He says, "Now, I'll be honest. I did download a pirated copy of SpinRite from a dangerous pirate crack site to see if SpinRite really was the amazing program you and Leo go on about. After I let it run over the weekend, I was able to boot the computer up just fine. I went to work with the machine and started it up in front of the IT guys. They were shocked and amazed that I was able to, quote" - and he has this in quotes - "'work my necromancy on the hard drive,' unquote. Later that day they told the CEO about SpinRite and what it did."

Leo: What is this SpinRite? Is this a hacker program?

Steve: Well, he says, "In the CEO's opinion, he felt that I had stolen personal information, and I was fired."

Leo: Oh, man.

Steve: "Here is why I want to thank you for making SpinRite. Now the IT department has a copy of SpinRite, and so do I, both legally purchased. And because I was let go after that, it freed me from the dead-end tech support job and gave me the motivation to go back to school and get my bachelor's degree."

Leo: Good.

Steve: "So thank you, Steve, and thank you, Leo, for helping me finish my education. As a side note, please don't read my name. Now they want to hire me back. And the customers I had listen to Security Now. And, well, they know where and what I'm going for. Plus I don't think the company would like it if this story about them went public. I have an associate degree in software programming and now going for marketing and economics at Slippery Rock University."

Leo: Great. What a good story. That's...

Steve: Oh, it was a great story. Thank you, Scooby, for sharing your...

Leo: I know so many people who have been fired or gotten in trouble for demonstrating stupid security policies in companies, including our friend Randal Schwartz.

Steve: Yeah, I mean, I hear stories, I read stories, because I'm keeping my eye on what's going on, all the time with people who get into trouble because they say, when they bring it to someone's attention, literally it's shoot the messenger.

Leo: Yeah, yeah. You know, it's one thing if you're hacking their systems in-house to demonstrate a security flaw. That's probably ill-advised. I think Randal learned his lesson on that one. But they put the computer out on the front porch.

Steve: Yeah. It looks like, hey, let the people take it away because it's got nothing on it.

Leo: He did them a favor.

Steve: Even though it was full of critical information.

Leo: Unbelievable. But I guess maybe a lesson to all of us. Even though you know that it's a bad idea, sometimes demonstrating it to the company is a bad idea, too. Oh, man.

Steve: Yeah, wow.

Leo: That's an incredible story. Thank you, Steve. Ready for some Q&A?

Steve: I'm ready. We've got the first five, not surprisingly, are follow-ups from last week's WPA and TKIP hack/crack episode. So we've had, of course, listeners listening intently. In the case of that last episode, listening in some cases several times to pick up all the details. But some great questions came out of that, and a bunch of other stuff, too.

Leo: Well, you put some minds at ease when you explained exactly what the problem was. You sure made me feel better. Although I did, I have in the intervening week changed everything over to CCMP encryption, AES encryption, so that - just why not.

Steve: Yup, exactly. I have done the same thing. It's like, well, why not do it?

Leo: Why not do it? Ben - actually two similar questions from two different listeners. Ben Jaques in Des Moines, Iowa, had his thinking cap on during last week's TKIP Hack episode. He said: When I heard about the crack, I didn't panic. I knew that tech media often cries wolf on security issues. I also knew that in a few days you'd explain it all on Security Now! - that's a nice vote of confidence - in a way that only you can: precisely; correctly; in great detail; and, most importantly, in a way we can all understand. So here's my question. You said during the show if the access point detects more than one MIC failure, one MIC failure - what do they call them, Mickey...

Steve: Yes, well, Mickey is the - or, no, Michael is the...

Leo: Michael, that's right, yeah.

Steve: Michael is the protocol, or the algorithm that they use, yes.

Leo: More than one MIC failure in a 60-second window, it's going to set off an "alarm" - he puts that in quotes - in the network, shuts down for 60 seconds, rekeys everyone. When I heard this, "alarms" went off in my head. Doesn't this open the door for a denial-of-service attack? My idea is this: An attacker uses the chopchop method to come up with a valid checksum for a packet - you described that last week - then purposely causes multiple MIC failures within the 60-second window, causing the access point to shut down. When the network comes back up 60 seconds later, boom, do it again, and in another 60 seconds down, on and on. Later in the show you and Leo seemed to conclude that this new WPA problem is an interesting hack, but it can't be used to cause any real damage. Well, if this attack works, it would render the wireless network unusable. I mean, it's not as serious as, say, WEP, where an attacker can get in and see every packet on the network. But a DoS attack still is a serious threat, wouldn't you agree? Now, I'm not an expert. I haven't studied this problem as you have. But I have listened to every episode of Security Now! at least once, so I'd like to get your thoughts on this idea. I'm hoping it would not work and that you'll explain why not.

Bruce Harrison in Durban, South Africa, same problem. He says: Hi, Steve, and a wave to Leo. Listening to Episode 170 on the TKIP hack, I was wondering if one does generate two or more MIC failures in a row, does that stop the access point from working? Or does it just stop traffic from the client that generated the errors? If the former, it does seem that it would be trivial to continuously bring down a wireless network by generating two or more MIC failures in a minute. Warm regards from Africa. So what's the story, Steve?

Steve: Both guys are absolutely correct.

Leo: Okay.

Steve: It is a known and significant problem with the TKIP protocol. And this of course applies to WPA and WPA2 setups, as long as they have TKIP enabled even as an option, because in recognition of the fact that the use of TKIP is retrofitting a very insecure solution, the predecessor, WEP, in recognition of that, these guys who designed it saw, well, you know, this is really not as secure as we need it to be. It's as secure as we know how to make it. So we're going to detect hacking attempts and develop a countermeasure against that.

Well, as we saw, if the checksum on the packet is wrong, then the packet is rejected as a transmission error. But if the checksum is correct, and the eight-byte MIC check, basically a packet authentication, if that fails, then the access point says, whoa, the packet was received correctly, but it's got an incorrect MIC value inside. Meaning that we're in the process of being hacked. Well, for some reason they said, well, we don't want to be shutting down and rekeying because of this potential denial of service. We don't want to do it by mistake. So when we get one, we start a 60-second timer. And if we get another one before that 60-second timer expires, two within a one-minute window, then that's really unusual. We need to suspend and rekey. So literally the access point shuts down for a full minute, completely. The whole network goes dead. And then as it comes back up it rekeys everybody.

And so unfortunately what this means is it would be trivial to take the code that already exists - this code is in the Aircrack code, open source, freely available - and turn this into a persistent denial-of-service attack. So that anybody with access to a network that had the TKIP protocol running, even as an option, even if everybody using the network were using the CCMP that uses AES encryption, if they're using the good super-secure protocol, if TKIP was present, if it was enabled in the access point, the access point would sense this because broadcast packets will still use TKIP, believing that there might be some clients that need that. So even though no clients are using it, it's still available. You could sniff packets. You could maliciously spoof them in a way that would cause the MIC test to fail. Whereas the checksum on the packet would pass, the access point would think, oops, we're being hacked, and shut down. And you could literally hold the network down as long as you wanted, denying it to everyone.

Leo: You would need to have access to the network by being proximate to it; right? You can't do this over the Internet. You'd have to be right there.

Steve: Correct. So, exactly. So...

Leo: So you could, I mean, let's face it, there's probably a million ways to do a DDoS to a WiFi network, including just getting something on that frequency and jamming it.

Steve: Yes, exactly. And I think that's why it's not such a huge concern is, you're right, all you have to do is just spew out radio noise on the same channel that the access point is using. And you're going to, if it's loud enough, it's going to override the authentic transmission and cause all kinds of problems for people. So it's like, yes, well, people could do that. But at the same time the hacker is denying himself access, so that's arguably less of a problem. But it is...

Leo: Well, they have to be sitting in - they have to be within a hundred meters. So it wouldn't be too hard to figure out who's doing it. So, I mean, it's just, it's you're exposing yourself, right, because you have to sit there. It would be one thing if you could do it over the Internet. But since you have to be physically proximate...

Steve: Well, but you could imagine people, like in a coffee shop, if for some reason maybe there were too many people using the network, and so someone said, okay, I'm going to clean the network off by shutting it down until people give up and go away, and then I'll release the attack, and I'll have access to the network without all this too much competition.

Leo: Right.

Steve: But definitely possible.

Leo: It's doable in many, many, many ways.

Steve: Yes.

Leo: So that's why I think probably nobody's made too big a stink about this.

Steve: Yup. It's a known problem.

Leo: Well, as long as we're doing similar questions, I've got three similar questions from three listeners. You could just take one, you know? All right. Marc Carroll at the British Army Base - we want to give you all credit - Falkland Islands, VPNing to U.K., has a TKIP question. To help stop potential hackers even decrypting small packets, could one, if their router supports it, set their WPA group renewal key time to a lower value? Say 10 or 5 minutes or even lower than that? He says his router default is one hour. Not necessary for me as my router supports CCMP without TKIP, a Buffalo router WHR-HP-G54 - oh, he's using the Tomato firmware, that's why, 1.21. But perhaps that could be some use to people who don't have that luxury. Increasing your default group renewal key time.

Cam C in Melbourne, Australia, wonders about group key renewal, as well. You mentioned in Episode 170 that the method requires 12 minutes to perform the chopchop, and that routers rekey every hour. All of my wireless routers allow the changing of the group key renewal interval. Would reducing this timer to 10 minutes eliminate the limited attack?

Ted also came up with this idea. He's in Research Triangle Park, North Carolina. He says: I have to stay with TKIP. I have a PDA that only supports TKIP. This is not unusual, by the way. Same thing with WEP. People often are stuck with older hardware that forces the use of less safe protocols. He says: As it sounds, though, exploitation of the small hole would require 12 minutes. I run DD-WRT, another patch firmware like Tomato router, on my Linksys WRT54G router, which has a lot of options, as I'm sure you know. So I just changed my key renewal interval to 11 minutes, one minute shorter. Does that sound like an effective patch? Thanks again for the superb podcast. My coworker and I eagerly await the posting time each Thursday. What do you say, Steve?

Steve: Well, and I should say, you know, we read three. There were many people. I really appreciate, clearly people are listening so closely, and we've got smart listeners who are saying, wait a minute, okay, if we need to determine the last 12 bytes of the packet, and the way we determine each byte is by successfully changing - we take a byte off the end. We fix the checksum so that it works. But in the process we're going to get a MIC failure. And so when that happens we know that we got - we guessed the byte that we chopped off correctly. But we now have to wait for that 60-second timer we talked about in the first question. We have to wait for that 60-second timer to expire before we try to get the next byte. Which means it's absolutely the case that there is no way to perform this attack in fewer than 12 minutes.

So these guys are saying all of the routers that they discuss, and many other access points, do allow you to change the rekeying interval from an hour to whatever you want. Normally it's specified in seconds. So they often have 3,600 seconds, which is an hour. But, for example, 660 seconds is 11 minutes. And so it's interesting, too, because the attacker would, if he really wanted to perform this attack, remember that it generally takes 12 to 15 minutes to get all of this work done. Well, you don't know, okay, so 12 to 15 is a quarter of an hour. So you might have the access point rekey itself right in the middle of your attack anyway just because an hour, only four of those fit into an hour window, and so just by chance you might have the hourly rekey occur. So part of the attack, of an effective attack, would probably be to wait for a rekeying because then you know that it just happened, you know that presumably there won't be another one for an hour, unless somebody was wise to this and brought the rekeying interval down to something less than 12 minutes. Doing so prevents you from being able to get confirmation of all 12 bytes within 12 minutes, and you're out of luck. Because as soon as the rekeying occurs, everything changes. None of the bytes you guessed until then are useful then under rekeying. So, yes, absolutely it is the case, if someone has to continue using TKIP, and your router allows you to bring that interval down, that completely defeats the attack.

Leo: Well, there you go. These guys are sharp. Sharp cookies. Paying attention. Dennis Wigmore in Port Hope, Ontario, Canada thinks he may have found a faster approach for TKIP hacking. Man, I'm impressed. So I was listening last episode. You explained we'd have to wait a minute to continue every time we correctly guessed a valid byte of the key stream. What about hitting up the deauthentication attack? We force them to reconnect, send out another key. Couldn't we do this until we had enough information and go from there? That would dramatically speed it up, wouldn't it. As I know, this is only related to TKIP injections, but wouldn't this be a quicker method than just active listening?

Steve: Well, no. What Dennis missed is that we're not trying to get the key. So what we're trying to get is we're trying to get a sample stream of pseudorandom data emitted by the key for a given packet. And that's only going to be valid for that given packet. So the idea is to just change the data in the packet, but leave everything else the same. Leave the packet number the same because that determines specifically which chunk of pseudorandom data that is going to be used in that packet. So if we were to cause a deauthentication and reauthentication, that would be a rekeying, and so we're back to square one again. So even though you could certainly do that in much less than a minute, every time that happens you know nothing about the new key, so anything you learned previously would just be washed away. And that is specifically why you don't want to allow a rekeying, if you're a bad guy. You don't want the rekeying window to close on you when you're mid-attack because you just have to start again.

Leo: All right. Okay. Mike in Salt Lake City has an important note about WiFi QoS. We had mentioned that you need QoS turned on for this hack to work. He says: I just wanted to send my thanks for the good TKIP Hack show. I checked my router, and I do have TKIP enabled. I need to. Again, legacy hardware. I went to check QoS, I couldn't find it. It did, however, have an option called WMM. I did some research, found out that that's the same thing. So now that's disabled. I just wanted to let you know so you could notify listeners about the distinct acronym. I never heard of WMM.

Steve: Yes, it's WiFi Multi Media. And he's right. I should have mentioned this last week. That is what many routers call Quality of Service. It is a subset of the 802.11e subspec. What it does is - oh, and I should say WMM is like WPA and WPA2. It is a certification. And it is a trademark of the Wi-Fi Alliance. So newer routers, if you look at the feature lists they have, you'll see WMM, which stands for WiFi Multi Media. What this does is...

Leo: Why invent a new acronym?

Steve: I know, I know. I know.

Leo: Stupid. Just so they could trademark it.

Steve: Well, so they could trademark it, and so they...

Leo: So they can license it.

Steve: Exactly. And so that it's something that people can only put on their box if the Wi-Fi Alliance has had a chance to check their hardware. I mean, I really - fundamentally I really agree with the value of somebody certifying interoperability. Otherwise it'd just be a pain in the butt to buy something and not know if it was going to work.

Leo: Right, okay.

Steve: So that's good. So what they've done is - and I have a feeling we're going to do an episode on this because it's going to be important moving forward. They assign four different broad categories of traffic: voice, video, best effort, and background. So as you would imagine, voice is the highest priority. Any traffic that is tagged as voice traffic gets to use the router with - gets to use the airwaves with minimal delay. Video is second priority. Best effort is default. So anything not tagged is default. And then, being sort of a good citizen, you can tag traffic as background. And it's the lowest priority, meaning that when the airwaves are free and there's nothing going on, then the background track will have an opportunity to flow.

So, again, it's all about optimizing the use of a limited resource. You would not want software that said, oh, I want much better download speeds, so I'm going to tag myself as voice traffic so I always get more. In doing so you would corrupt any real voice traffic that was also trying to use that same WiFi access point because there would be some big bulk download going through with the same level. Now, it's incumbent upon the applications to perform the tagging. So it's not something that's done automatically. Which I was a little sorry to see in the specification because arguably you could say, okay, web traffic on port 80, and especially TCP, is not going to be voice. Voice is going to be UDP traffic, as we know, as opposed to over a TCP connection that does not carry voice very well.

So they could have done some things smart by default. But apparently they haven't because you do require application in order to do tagging. But with any luck, applications like Skype that are all about quality and Voice over IP, you would imagine that Skype probably is tagging their traffic as voice in the case that it might be going over a WiFi access point and want to maintain the highest quality possible.

Leo: Right, yeah. So WMM is QoS.

Steve: It is QoS. And if - oh, and here's the point. If you need to use TKIP because you've got legacy hardware and your access point says either QoS or WMM, disable WMM. In doing so you've turned off Quality of Service support, and this hack will not function.

Leo: Okay, good. So that does work.

Steve: So that's the cool thing, yes, it absolutely does work. Disable WMM, WiFi Multi Media, that is QoS. And without that the trick these guys use of allowing a replay attack, because that is something, basically, when I mentioned before we're changing just a part of the packet, we're changing the payload, but there's a lot we can't change. Part of what we can't change is the packet number. And so by not being able to change the packet number, the replay protection comes in and prevents us - we would normally reject that packet because it's already been used. But they use the Quality of Service hack in order to get around that. So if you turn off either QoS or WMM, WiFi Multi Media support, in your router, then you shut the hack down, too.

Now, if you happen to be using Voice over IP through a WiFi connection, you might find that there was some difference. Although my sense is, I mean, you hadn't even heard the acronym, that acronym before. My sense is this is still very new and not yet widely deployed. So it's probably nothing is using it anyway. It's one of those features that's there for the future, not getting heavy use yet.

Leo: Yeah, VoIP does use QoS, but usually they use their own router, and usually they want to be wired. In fact, in order to do it they want to be outside your router, connected directly to the Internet.

Steve: Right. They want, like, nobody in the way.

Leo: Right. Which means they're probably doing some sort of hack to QoS. Paul Kamet in Kaneohe, Hawaii has a great question about WiFi encryption. What a surprise. Steve, during the TKIP Hack show - I've listened to it several times now, by the way - it sounds like the IP address fields in the WiFi packet are encrypted along with the data. If that's so, how can this packet traverse the network? Even with the MAC addresses in the clear, it doesn't sound like enough information is available to send the packet across the country. Your thoughts?

Steve: I thought this was an interesting question because he's right. The IP, the source and destination IP addresses are encrypted. So it's only when they're decrypted that anyone knows where these are bound for. What's important to recognize, though, is that within a single network, within an Ethernet-style network, whether it's a wired network like we all have at home or a wireless network, the addressing from one adapter to the next is not done over IP. IP has nothing to do with it. It's the MAC address. It's that 48-bit MAC address which is composed of a 24-bit vendor field and a 24-bit device field, together making 48 bits. That's the entire addressing information within the network. So that's why that is not encrypted and cannot be encrypted, because that is the way the packet physically gets seen by the receiver. Once it's seen, then it's decrypted using the shared key in the case of preshared key wireless, either TKIP or the CCMP AES style, the good, still strong, and believed not to have any of these weaknesses technology.

Once it's decrypted, then the machine is able to look at the packet and see whether the destination IP is one or more of the IPs that that machine may have. Because, as we know, it's possible for computers to have more than one IP. And this is the way they can have more than one IP. They can only have one MAC address per adapter. But that adapter could be associated with more than one IP. So the machine says, oh, is this an IP that is mine? And, if so, then it proceeds to process the packet further.

Now, as for how can a packet that's encrypted, whose source and destination IP addresses are encrypted, how could that, as he asks, cross the country, well, this is a source of another set of problems with potential with WiFi, and that is the nature of the gateway, which is typically the access point, which is also normally a router. The nature of the gateway is that when the packets which are encrypted are received at that gateway router access point, just as they come in they're received because they've got the gateway's MAC address. So exactly the way it works with a client, the access point will decrypt the packet in the clear and then look to see where it's going. It will typically have an external IP address that is not part of the local network, but part of the external network somewhere out on the wide global Internet. And so the router does what a router does. It says, oh, this is for outside. So it shoots it out of its WAN port. Notice that it's been decrypted in the process of crossing that router.

Leo: So it's only encrypted internally, never externally.

Steve: Yes. Yes. And in fact, that's one of the interesting things that can be done, one of a well-known attack that was primarily used earlier on about WEP. But it's still a concern in some cases about TKIP. If you can arrange to have, like, even just a ping, if you can arrange to have some sort of packet with known data sent into the network, which you're able to receive, if you - for example, say how a ping packet is able to carry a payload. So you could have a ping packet with just all zero, just filled with zeroes. If you could get that into the network and receive it, then for that particular destination that had a unique key, you end up with the encryption stream that was used to XOR against that plaintext, which is all zeroes. So it's just the key stream.

Leo: Excellent.

Steve: So because you've essentially - you've gotten the access point to perform encryption for you on a known packet. And as we know, if you know what the packet is, you just XOR that with the encrypted data, and you get back the key stream, as we talked about last week. So anyway, it's unfortunate that we're dealing with this legacy technology that just uses "exclusive or" and pseudorandom streams for encryption because it just really never was very secure. And it's tremendously good that we've gone to AES. And certainly in the future we're going to see more and more equipment moving in that direction.

Leo: Matt in Howell, New Jersey wants to add just a dash of salt. He says: Hi, Steve, Leo. Been a listener since Episode 1, love the show. I'm a computer science student. And many times lectures were just refresher courses on something I'd already learned on Security Now!. I was in the lab last week, and a teacher was about to start a class. He said I could stay for his lecture as long as I didn't cause any disruptions. I stayed to finish my lab and listened in on his lecture. It was for a graduate course and was all about hashing. This is a smart kid, I can tell. Something he mentioned at the end of his lecture caught my interest. He described something called a "salt value," which as far as I could understand was a constant that is mixed into the value being hashed to increase randomness. Am I right? If not, can you explain this to me? What is salt? Thank you, Steve, for your supplementary education.

Steve: Okay. So we have well-known secure hash functions. MD5 is a hash. SHA is a hash. The idea being, and we've discussed this in episodes long ago, that a hash function is a so-called one-way function, meaning you put any stuff you want into the front of it. And when you're all done, you read a value out which is essentially a signature. It can be thought of as like a signature for everything that you put in.

Now, the reason a hash is cryptographically strong is it is impossible, or I should say really, really, really infeasible, to deliberately get a signature out from something that you put in. If you make a change at all to anything coming into the hash function, you get something completely unrecognizable out. That is, for any change you make, half the bits are going to be different. So there's no way to figure out, given what you got out, what it was you put in. But it is the case that every time you put the same thing into, for example, MD5 or SHA-1, when you put the same thing in, you get the same thing out.

Well, now, that's important because, if someone makes a fingerprint, for example, people who download software on the 'Net may have seen where source code will be available, and then the MD5 will be given for the source code. Well, that allows someone to receive that source code and to perform their own MD5 hash separately, that is, to recreate the signature that was originally created by the person who was offering the source code. Then you compare the MD5 listed with the MD5 you've got, and that verifies that you're using an exact copy of whatever it is that was originally run through the hash. So it's important that their MD5 is the same as your MD5, that SHA-1 is universally used in the same way in instances where you want fingerprinting.

Well, there is a downside to this, and that is, for example, with passwords. If passwords, for example, were only hashed, for example, with MD5, you could do something called a "precomputation attack." And it's popularly known as "rainbow tables" because the rainbow technology that's been used for anti-software hacking and anti-piracy was using an unsalted hash for a long time. And the idea is that you take like a whole bunch of dictionary words, and you hash them into the result. Then, if you have the ability to see what the hashed secret password is, you simply do a comparison between this large dictionary you've built of already hashed things because you can compare that much more quickly than you can perform all the hashes. So the idea is you only need to do the hashes once to create all of the results of the hashes. And those you can compare very quickly. When you get a match, you know what the original password was that created that. And there are on the 'Net, you can download huge, huge tables of precomputed hashes to use in hacking.

Well, the reason that works is the hash that was used to create the password is the same hash function as was used to build the precomputation table, if you added a little salt to the hash. If, for example, the system that was hashing the passwords, if it simply put in some pseudorandom stuff before the password, then essentially what you've done is you've created a keyed hash. You've created a hash that uses a standard function, but this salt is a key. And unless you know also the key, you're never going to get a hashed value that matches. So it's a way of taking a standard hashing function, but for your particular purpose. For example, just verifying passwords, you never need to have that transportable, unlike the instance I gave where you download software and you need to compare hashes. In this case a system that's a closed system merely wants to verify that the same input generates the same output. But it doesn't ever need to compare it with anything else. In that case you want to make a keyed hash by adding some salt. Just put something else in, in addition to the normal input, and you're going to get a hash value out that's completely different. So that's what salt is.

Leo: Very cool. And you've mentioned it, I think, before in the program.

Steve: Probably.

Leo: Yeah. Sounds familiar to me. Rodney Morton from Round Rock, Texas, on assignment in Germany, has the sniffles. He says: I read the following article about keyboards being remotely sniffable. Yeah, in fact I meant to ask you about this because I just saw this myself. I want to have your take on it. It's a BBC article. He says, and the article says, "Keyboard sniffers to steal data. Computer criminals could soon be eavesdropping on what you type by analyzing the electromagnetic signals produced by every key press." The attacks worked at considerable distance, like across the office, like 20 meters. Have you heard about this?

Steve: Oh, yeah. I'm looking at the article. It says, "By analyzing the signals produced by keystrokes, Swiss researchers have reproduced what a target typed. The security researchers have developed four different attacks that work on a wide variety of computer keyboards. The results led the researchers to declare keyboards were, quote, 'not safe to transmit sensitive information.' The attacks were dreamed up by doctoral students Martin" - whoo, boy, Martin [Vuagnoux] and his friend Sylvain [Pasini] - "from the Security and Cryptography Laboratory at the Swiss Ecole Polytechnique Federale de Lausanne, EPFL." Thank goodness for acronyms. "The EPFL students tested 11 different keyboard models that connected to a computer via either a USB or a PS/2 socket." So these were not wireless keyboards. "The attacks they developed also worked with keyboards embedded in laptops. Every keyboard tested was vulnerable to at least one of the four attacks the researchers used. One attack was shown to work over a distance of 20 meters.

"In their work, the researchers used a radio antenna to fully or partially recover keystrokes by spotting the electromagnetic radiation emitted when keys are pressed. In a web posting they added, 'No doubt that our attacks can be significantly improved since we used relatively inexpensive equipment.' In videos showing their early work, the researchers are seen connecting keyboards to a laptop running on battery power. They avoided using a desktop computer or an LCD display to minimize the chance of picking up signals from other sources. Details of the attacks are scant, but the work is expected to be reported in a peer-reviewed journal soon. The research builds on earlier work done by University of Cambridge computer scientist Markus Kuhn, who looked at ways to use electromagnetic emanations to eavesdrop and steal useful information."

So we've talked about this kind of thing in various different formats in the past. Traditionally this is what was known as TEMPEST, where it was possible to aim a directional antenna at a CRT, and the CRT generated electromagnetic radiation in a defined pattern that was exactly dependent upon what was showing on the screen because the electron beams were being turned on and off to energize and illuminate the phosphor on the back of the glass of the screen. And it turned out that you could - it was possible to decode it.

So it's really not surprising when you think about it. But this demonstrates, I mean proves, that typical computer keyboards which use a - they're being scanned electromagnetically. There's a scanning process going on in order to essentially pick up which keys are being depressed. So when you press a key, there's an event that occurs which can be a change in the scanning pattern that is always generating a weak transmission from the keyboard. And the act of typing generates enough output, enough radio frequency interference, that it's possible to determine what's being typed. And that's not using any sort of wireless, you know, we've talked about the really weak, single-byte XORing encryption that's being done by many insecure wireless keyboards. This is wired USB and PS/2 keyboards.

Leo: This is something similar to what they call "van Eck phreaking," where you can see, monitor the electromagnetic radiation of monitors through a wall and see what the monitor - on the monitor. Which I don't know if it's science fiction or real, but that's what they call it.

Mark Piper, writing from an undisclosed location, wants a hard drive password: I understand now most hard drives have the feature to request a password on startup. How is this feature typically activated? I can't seem to find it anywhere in my BIOS configuration. If my BIOS doesn't support it, is there anything I can do short of whole disk encryption to require preboot authentication? Thanks in advance. I think we talked about this before. This is something IDE has had for years.

Steve: Yes, in fact I don't think there's a drive on the market now that doesn't offer this as an option. Unfortunately, you absolutely do have to have support in the BIOS because it is before the drive can be used at all it needs to be given the password that the BIOS knows to contain. Either it's in the BIOS, or it's asking you for it, depending upon how your security setup is working in the computer. It might be that you authenticate, for example, by swiping your finger on a fingerprint reader. That authenticates you to a TPM, the Trusted Platform Module in the BIOS. That then unlocks the key that the BIOS gives to the hard drive in order to unlock the hard drive in turn. The problem is, until that's done, you can't read anything from the drive. So there's just - there's no way to do it.

I actually came up with an interesting idea a while ago that I never implemented in a product, either commercial or freeware, where you could boot from a floppy or a USB device. It would support that function that the BIOS lacked and unlock the hard drive and then transfer the boot over to the hard drive. But I've never known that that's been done by anyone. I never got around to do it, and I've got so much stuff on my plate now, there's no way that's going to happen. But the answer is, unfortunately, if you poke around your BIOS, and you cannot find anything about establishing a hard drive password, then the BIOS doesn't support it.

You might, if this is, for example, a laptop, you might check to see whether there's any newer firmware for the laptop because it is a feature which is appearing more and more currently in late-model laptops and in late-model BIOSes. So you might find that updating your BIOS would allow you to suddenly have that feature where you didn't before. But without going to extreme measures, if the BIOS doesn't support it, there's really no way to get that hard drive to - the password to the hard drive in order for the BIOS to boot it.

Leo: And this is different from the password protection, the more modern, like Hitachi, the encryption that's built into the hardware of the drive itself. This is something else. This is an IDE password, basically.

Steve: Well, yes, exactly. So it's not whole-drive encryption. It's essentially the drive is locked. So even though the data out on the drive is still in the clear, you just - you can't get to it at all because it's got that password. Although...

Leo: Is it pretty secure?

Steve: It's good for thwarting most people. But if you had a government subpoena, the government could go to the hard drive manufacturer and say here's the court order, remove the lock from this drive. And they could definitely do so in order to unlock the drive. And then all the data that is available on the drive would be in the clear. It's not as secure as full-drive encryption.

Leo: So don't confuse it with that.

Steve: Right.

Leo: John in Columbus, Ohio wonders how universal Plug and Play should be. He says: Hi, Steve. I'm wondering if I should still be disabling Universal Plug and Play on my wireless router. I wasn't sure if Microsoft solved the buffer overflow issue or not. Thanks, and keep up the great work. I disable it all the time, on every router.

Steve: I know you do. In fact, I was listening to you, Leo, when you were having problems with the...

Leo: I was listening to you.

Steve: You were having problems with a new router that you had installed. And I think it needed to be configured for VPN use or something. And you thought, well, maybe I'll try turning it back on to see if that will solve the problem.

John's question confuses a couple different things, but there was an important point that I wanted to sort of clarify. He's talking about a Microsoft buffer overflow which existed long ago in the Universal Plug and Play service that was running by default in Windows. In order to deal with that, and this is in the era before Service Pack 2 that has the firewall on by default, in order to deal with that, I created the little freeware UnPlug n' Pray. And all Unplug n' Pray does is make it trivial for anyone to disable that service. Because most people at the time had no use for Universal Plug and Play. That is, and so what happened was, in original versions of XP that did not have the firewall turned on by default, it turns out they were exposed to a buffer overrun vulnerability that was widely exploited until Microsoft brought up their XP firewall and issued a security patch in order to close that.

My feeling has always been let's not have services and open ports unless we need them. So I've always been annoyed that Microsoft was running all this stuff by default. And that's, of course, why early on Microsoft had such a bad reputation for security. Being behind a router, of course, further protects you.

Now, when John asks, should I be disabling Universal Plug and Play on my router, the answer is absolutely. There are known trojans and malware which are now Plug and Play aware. I mean, I predicted this years before it actually happened. It has happened. So the idea is you can get malware in your machine, for example, which you invite in through a browser vulnerability. The malware will send a packet out, a broadcast packet, onto your LAN, looking for your router, through Universal Plug and Play. The router says, oh, hi there. Yes, I'm here. And the malware says, ah, thank you. Please open the following ports. And the router will dutifully do so. There is no security, believe it or not, built into Universal Plug and Play. Nothing that prevents something on the inside of the network from accessing the Universal Plug and Play service that is actively exposed and unfortunately often turned on in routers.

Leo: And you won't get a warning or anything?

Steve: Nothing.

Leo: The firewall won't tell you?

Steve: No, there's no - no. I mean, again, this is Microsoft making sure it's all easy. We just want, you know, when you plug in your refrigerator, we want everything to be discoverable and for all these different services to expose themselves so that everything just kind of works. Unfortunately, that implies that everything, every single thing within your network perimeter is trusted. If anything isn't, if anything is compromised, then - and for example, Universal Plug and Play, essentially it is a programmatic access to your router's API. That is, anything you can do from the screens, and in some cases even more than you can do through the user interface, can be done silently, without your knowledge, behind your back. Like setting up a DMZ or opening ports, that then of course just allow the floodgates because it's like not having a router anymore. Certainly not one that you can trust.

Leo: You know where you really get people using this is Xbox 360. A lot of kids want to use Xbox Live, which lets them play games, start games with other people on the Internet. And the Xbox, I think, in a very irresponsible way, will check your network and say, if you don't have Universal Plug and Play on, it will say, oh, you have limited connectivity. And I get calls all the time on the radio show.

Steve: And what's so annoying, Leo, is you only need to map a few ports through the router to the Xbox in order for it to say, oh, you've got excellent connectivity. It's simple to do. But again, but you're right, they just say, oh, turn this on. And...

Leo: Well, and most people, you don't really want, I mean, most people would probably - when people call me, I say, well, you've got to do port forwarding. And they go, huh, what? They get nervous. And so I see why Microsoft turned this on. But it's unconscionable. I mean, they really are encouraging people to turn on Universal Plug and Play without explaining to them any of the risks. You know, they're really scaring them into it. And it just makes me crazy. I get calls about that all the time.

John H. Storey in the United Kingdom wrote with the subject "Port 1029 Open." You must get - you probably get a hundred of these a day, port XYZ open, because of ShieldsUP!; right?

Steve: Right.

Leo: Hi, there. How do you close this port externally? What do you recommend? I'm a beginner at computing. Best regards, John H. Storey.

Steve: Well, this was obviously a very short question. Something is telling him port 1029 is open. It's probably ShieldsUP! which is telling him 1029 is open. And that's a problem because it means that he doesn't have any other protection between his computer and the Internet. That is, there's no router. And it sounds like his firewall is down because even the XP built-in firewall would recognize that port 1029 is only meant to be open internally, within the system. And if you look, we had talked about, I guess it was a couple weeks ago, netstat and the mysterious self-closing of the netstat window because I imagine that the listener was entering it at the Run line under the Start menu, rather than starting up a DOS box separately.

Netstat will show you often a number of very low-numbered ports, like starting at 1025, 1026, 1027, down, very low number, just over 1024, which is the boundary. And those are typically ports that are opened by processes within the system that want to use the networking stack for talking among themselves. So that's meant to be local, and never meant to be exposed publicly.

However, if you've got no firewall on the computer, or if your XP firewall is turned off or has been expressly configured to allow one or more of those ports to come through, they'll be exposed. So when he asks how do you close it externally, this is not something you can stop. That is, you can't stop the process that is opening 1029 because it's probably service host, that ubiquitous service carrier inside Windows.

Leo: And you don't know what it really is.

Steve: Exactly. So what this does say is that your firewall must not be currently enabled. That's also surprising because XP is so annoying now about making sure you've got your firewall enabled. I know, I say "annoying" tongue in cheek because we really want it that way, especially for a beginning user of computers. So I would explore whether the firewall is enabled. And in the firewall configuration there are various overrides that can be applied. You really want to turn those things off. If you're a beginning, you probably don't need them turned on. Maybe something installed itself that wanted to make an exception for itself through the firewall, which is on that port. But it's worth pursuing why this thing is open to the outside world because that's not usual.

Leo: Let's move on now to our topic of the day, Steve Gibson. Or our Tip of the Day is a PayPal tip. We got two different ones from our listeners. I don't use - I use PayPal for donations, so I'm always interested in PayPal tips and tricks. Starting with Mike O'Hare in Georgetown, Massachusetts. He has a PayPal workaround. He says: Steve, listening to Security Now! podcast 169, you said PayPal only accepts a checking account number. What I did to get - I guess when you, well, I don't - you can use a credit card, but eventually you have to give them a checking account; right?

Steve: Well, the problem is that they always default to a checking account.

Leo: Oh, right, right.

Steve: So you're having to manually override that. But more importantly, when you use their one-time credit card number generator, there's no way to override that. It insists on pulling from a checking account.

Leo: Ah, okay. So he says: What I did to get by that was open a "savings account" in my bank, specifically for PayPal use. I can then transfer whatever money necessary from my checking account into the savings account to cover PayPal purchases. PayPal is happy, and I'm happy they don't have access to my checking account. And I imagine he keeps the balance at zero unless he needs the money in there. I wish you and Leo the best. I'm a happy SpinRite user. That's a good idea.

Steve: I just liked that idea. In fact, I may adopt that idea. I mean, I do like using the PayPal service. It's just annoying that they want to pull from my main checking account. And I hadn't - I don't know why I hadn't thought of just setting up an account for PayPal.

Leo: Just for them, yeah.

Steve: PayPal likes pulling from checking. The reason they even have my checking account information is that that's part of what you have to do to become fully identified or certified or recognized or something. I mean, they really do push that on you. And so it's like, okay, fine. Take this one, and I'm just going to move money as I need to into that. So keep a relatively low balance, PayPal pulls from it, and then you replenish it as necessary. I think that's a useful solution. So I wanted to...

Leo: I am not fond of PayPal in so many ways. And yet it is, you know, the most convenient way for us to get donations to the network. So we continue to use it. I'd love to find an alternative, though. I guess Google Payments, maybe Amazon Payments?

Steve: Yup. It's going to happen. Ultimately I think somebody will come along and - but PayPal has a huge edge because they were there so right off the bat.

Leo: Right. Moving on to Jason Deabill in Southampton, U.K. He provides an important update on PayPal security tokens. That's that little football.

Steve: Yup.

Leo: I love my little football.

Steve: And that's one of my security devices I have not gotten tired of. I've got mine hooked up to my PayPal account.

Leo: It's really, really handy. So he says: Hi, Steve. First up, many thanks for the excellent website and podcast. I can't express how useful they've been to me in recent years. I just thought I'd drop you a quick note to point out that PayPal security tokens are now available outside the U.S. Yeah, these were available only in the U.S. for a while.

Steve: Yeah, yes. In fact, we disappointed a lot of users because I didn't realize that was the case. And we immediately got mail saying, hey, I can't get it where I am. So apparently, and I wanted to bring this again to the attention of all of our listeners, that I don't know exactly where they're available, but they're still very inexpensive. He said he ordered his for about three pounds, which is about $5 U.S., so it's about the same price, super affordable.

Leo: It's a token price because it costs them - if you forgive the pun. It costs them more to make it than that. So that's just really to encourage you to buy it, but they want to give you some price to kind of make sure you're going to use it.

Steve: Yeah. So any of our listeners who have been disappointed that they could not get the PayPal token wherever they are, you may want to check again because at least we know it's in the U.K., and it may be many other places, as well. Which would be really terrific.

Leo: You know, this is today some good news. Both Visa is going to start doing those - it's the same idea, that token card. I'm really glad to see more and more of that.

Steve: Yeah, these things seem to take a long time to happen. But it's really something we desperately need because we want to facilitate secure and safe Internet commerce.

Leo: Absolutely. Steve, we've come to the end of our 12 questions. 12 questions, good and true. We thank all of you. We do this every other episode. You can always ask more questions of Steve by going to GRC.com slash...

Steve: Feedback.

Leo: Feedback. And a great place to put your feedback. Of course, he has security forums. You can go there to find out more, as well, and to ask questions on the security forums. In fact, GRC.com is a great resource all around. Besides ShieldsUP! there's lots of other free utilities. Of course there's the great SpinRite, which everybody ought to buy just because it's the hard drive maintenance utility. You've got to have that. And you also find 16KB versions of the show. There are transcripts. Every episode we've ever done. It's all at GRC.com, the Gibson Research Corporation. Steve, thank you so much.

Steve: Always a pleasure, Leo. And next week we're going to have Ronen on...

Leo: Oh, that'll be fun.

Steve: ...the author of Sandboxie, to tell us about the product in more detail, the challenges he's had to overcome, and answer some questions.

Leo: Very good. Thank you, Steverino.

Steve: Talk to you then, Leo.

Leo: Bye bye.