SECURITY NOW!

Transcript of Episode #508

# Exploiting Keyless Entry

**Description:** After catching up with a busy week of security news, Leo and I take a close look at the surprisingly weak and insecure technology used for today's modern automotive keyless entry and engine start systems. We show how easily it may be bypassed - perhaps for as little as $17 on eBay.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-508.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-508-lq.mp3

SHOW TEASE: Some shocking news: Steve Gibson says that this keyless entry fob that I use on my car every single day could also make it possible for thieves to get into my car or even drive it away. The details on keyless entry, plus all the security news including Steve's thoughts on that guy who claims to have hacked an airplane midflight, it's all coming up next on Security Now!.

**Leo Laporte:** This is Security Now! with Steve Gibson, Episode 508, recorded Tuesday, May 19, 2015: Exploiting Keyless Entry.

It's time for Security Now!, the show where we talk about your security, your privacy. It really actually is one of the most fun shows we do now because this is the era of security breaches. Steve Gibson's the guy who for almost the last 10 years has been covering this subject, so nobody knows better than Steve. Hi, Steve.

**Steve Gibson:** Hey, Leo. We finish our - we're in our 10th year. We finish it up in August. And it's funny because almost immediately after - apparently - after Verizon purchased AOL, they cut off the CDN caching of our prior episodes.

**Leo:** What?

**Steve:** You switched, I guess, yeah, you switched to Cachefly. But all of my links were still bouncing through Podtrac for enumeration.

**Leo:** Oh, you just were out of date, that's all; right?

**Steve:** I guess. Well, but...

**Leo:** Because we have all those shows on Cachefly; don't we? I hope we do.

**Steve:** Well, no. I verified that because what happened was I start getting reports last week, because remember, like, last week was heavy on referring people to prior podcasts. Because like the whole How a Computer Works series that we were talking about, basically, in last week's Q&A, people were asking questions that we had answered years prior. But it turns out, when people went to those links, they were broken. And so I started getting reports saying, hey, you know, after Episode 417, everything's okay. But earlier than that, none of the high-quality links work.

Well, that was the transition from AOL to Cachefly. And my links are uniform, and I've got code in the server that looks at the link to decide where to bounce it after it runs through Podtrac. So anyway, the point is that I thought it was interesting that, of course, one of the pieces of news of the last couple weeks is Verizon's purchase of AOL. But apparently somebody from Verizon, like, went through the datacenter and saw this caching of podcasts and said, "What the heck is this?" and pulled the plug.

**Leo:** Well, a couple things. First of all, Verizon has not yet acquired AOL, so that's not the case. I mean, they bought them. But, you know, it takes a while. This has to be approved and everything.

**Steve:** Yeah, well, anyway...

**Leo:** So they're not allowed to do anything. And so I wonder when this actually happened. Maybe it's been going on for a while. And so did you check to see if the shows were on Cachefly? Because they should be.

**Steve:** Yes, yes. And so what I did was, I verified that Cachefly had them uniformly and then just removed the special casing code so that now everything bounces through Cachefly, and it works great. But the point, the thing that made me think of this is that, in order to make sure it went all the way back, I played the beginning of Episode 1.

**Leo:** Oh, yeah.

**Steve:** And we will do that on the 10th Anniversary, just - it was a short, I'm not sure how much of it, but it was just sort of charming. It's like we've come so far in 10 years. So I just got a kick out of - I think it was 18 minutes long, for one thing. But I got a big kick out of, like, listening to different music, and you were in The Cottage, and the air conditioning wasn't working very well, and...

**Leo:** Oh, horrible, horrible, horrible.

**Steve:** And you'd only done three TWiT episodes at that point and so forth. So it was kind of a kick. I thought, oh, this would be just perfect, to play the beginning of it to celebrate our 10th, the end of our 10th year of the podcast.

**Leo:** Let me just check real quickly because I think, oh, yeah, ours still point to AOL, as well. So I think we're going to have to fix that, too. I didn't realize that we still pointed - this is Episode 2.

**Steve:** And in fact I think someone did tell me that it was the TWiT links that were broken, and I got the tweet. And so I checked mine and fixed them. But come to think of it, I do think someone said that...

**Leo:** Oh, yeah. So we moved them over, but we apparently never - I wonder when that happened. That's interesting. Yeah, we used, I mean, I think they probably didn't know who to call. Like what is this stuff? They had no idea.

**Steve:** I know, like, what is this stuff we're caching and spending our bandwidth on?

**Leo:** In the earliest days of the network, I had forgotten, AOL was our bandwidth partner for the audio. And then when we started doing video, we did Cachefly. And then we eventually moved everything over. So I will fix that.

**Steve:** And I was glad to know that you - I was glad to see that somebody had backfilled all of the early audio on Cachefly.

**Leo:** Oh, yeah. We have them all. Oh, yeah. I mean, that's not a big deal to do that.

**Steve:** Yeah, but, you know, it took someone to, like, upload them or post them or do something to them, like, to get them back. Because it was, for me, the changeover was Episode 416. One of the people who tweeted said...

**Leo:** Wow, that was a long time.

**Steve:** I know, it really was.

**Leo:** Huh. So it's like eight years we did it on...

**Steve:** And then there were problems with AOL, like sometimes we would have a problem with the podcast after it had been put up on AOL, but we were never able to replace them. So, like, some podcasts, like 208A, but my link says 208. So then I also had a series of special cases that are in the registry. So the link referrer would check the registry to see if this was a special case; and, if so, it would rename it on the fly in order to do it. The good news is all of that I was able to get rid of, just in cleaning it all up, because you guys just uniformly moved what you had to Cachefly. So that was a win.

**Leo:** Yeah.

**Steve:** I think there were a couple that AOL didn't have, too. I used to get complaints about them from time to time.

**Leo:** That might be, yeah. Well, okay. I'm going to - actually, we're launching the new website in a couple of weeks, so...

**Steve:** Perfect timing.

**Leo:** That should all be fixed. If we don't fix it before then, it's because we know it's going to be fixed in a couple of weeks.

**Steve:** Right.

**Leo:** So apologies.

**Steve:** So this week we've got a bunch of news. And as you said, you know, talking about security, I mean, unfortunately, some of these topics have already been topics of the TWiT network because they're important, and they're what's happening, and they're about the Internet these days.

**Leo:** Well, we had you on The New Screen Savers because we'd just missed Venom.

**Steve:** Right. Right. So we're going to talk about Starbucks discovering the downside of convenience over security; yet another remote router takeover. This one is just - you've got to just shake your heads over this one. The largely overblown Venom vulnerability we'll just cover so that people aren't worried because I got a lot of tweets from people saying, oh, my god, Venom. Why it's never a good idea to hack into planes that you're flying in.

**Leo:** Yeah.

**Steve:** The growing ad wars. There's some news coming from Europe. My own troubles with Google. I've gotten a little more information about those links that were being blocked to TrueCrypt. A bunch of fun miscellany stuff. And I've got a really - I found a really good technical security, like, attack from three researchers, three Swiss researchers at ETH University in Zurich, who reverse-engineered and completely explained how Keyless Entry and Start - actually they're called "Passive" Keyless Entry and Start, "passive" meaning that they're user passive. They're active electronically, but the user needs to do nothing. So PKES is the acronym in the Industry, Passive Keyless Entry and Start. And it's shockingly bad.

**Leo:** Uh-oh. Oh, boy.

**Steve:** So we're going to have a great podcast today. And everyone will come away understanding what the vulnerability is in modern auto keying, what they can do in the short term, and what needs to be done in the long term. And the short term involves a little Faraday cage that you can get for $6.

**Leo:** I have keyless entry on my car, but I still have to push a button to start it. And most of the cars I've seen do that. So there's cars where you just get in, and they start?

**Steve:** No, you've got the problem, my friend.

**Leo:** I have it.

**Steve:** Someone could drive away with your car. I will explain how.

**Leo:** That would not be good.

**Steve:** And apparently, for $17 on eBay, you can get the equipment you need.

**Leo:** That really wouldn't be good.

**Steve:** Yeah, I know.

**Leo:** Yikes. How about my Segways? They can't drive away with those, can they?

**Steve:** That? You'll be able to chase them on the Segway.

**Leo:** At 12 miles an hour.

**Steve:** That's right.

**Leo:** So let's get going here.

**Steve:** Yeah. So a little bit of a problem that Starbucks has run into, not the first problem that they've had. But this one is recent and has drawn some attention. For those who are not Starbucks frequenters, as I certainly am renowned to be, Starbucks has a system which, for the iPhone or for sort of a loyalty card - I have a gold card somewhere that I got way back, immediately when they began offering it. And the system has always functioned such that you pull funds that have already been transferred to your Starbucks account, using the card. But as it gets low, below a certain level, you can in your account set it to auto refill. And then as iOS devices, and I'm sure they're on Android, too, have become popular, people are now able to - it puts up sort of a rectangular bar code that you just scan with the little scanner at the Starbucks register; and, bloop, you've paid for your drink. And similarly, you register a credit card, or apparently a PayPal account or a bank account, with Starbucks. Then that provides it a source of funds from which it periodically pulls in order to refill.

Well, the problem is that whole process for user convenience is autonomous. There is no verification, no double-checking, your card is low, just confirm to refill. They thought, oh, let's just, like, I'm sure someone said, wait a minute, if we tell people that their card has gotten low, they'll think, already, and won't drink as much coffee. So they want to make it transparent. They want to just sort of make it magical that, yes, that $7.45 coffee that you just purchased sort of didn't cost anything because the funds just magically float out. So I'm sure there was a marketing motive behind not verifying.

The problem is bad guys have figured that out. And although Starbucks corporate asserts, and I believe, that there's been no general hack of their system, individual subscribers are apparently being hacked. And, for example, in one case a woman who was hacked, and I'll explain what happened in a second, but she confessed to using the same password on her Starbucks account as she used for her email. Well, email passwords are among the least secure things ever. Anyone sniffing in-the-clear plaintext traffic on an open network, like you have at Starbucks, where you don't have encryption, or in any hotel setting and so forth, you know, a broad open WiFi, you see email credentials all the time because email is by default a nonencrypted protocol. And so more recently people are using SSL or STARTTLS in order to switch into secure mode. But still you see lots of email credentials by default on an open network.

So what's happening is, for whatever reason, weak password, maybe they checked their email while they were at Starbucks or someone just did a brute-force guessing on their account, if a bad guy gets in, then they're able to transfer all the money on the user's card to a gift card of their own. And then Starbucks' system autonomously notices that the user's card has fallen below the refill point, and so pulls funds from PayPal or their bank account or another credit card in order to refill the card. And you can specify, I want 25, 50, $100, whatever, per refill. So the bad guys waits for the new funds to appear and then transfers them again.

And in some cases people have lost multiple hundreds of dollars within a few seconds because apparently the system is very efficient at refilling your card. So anyway, just another lesson here in security that unfortunately it looks like, for economic motives, that is, not wanting to bother the person to say that their card is low again and proactively get permission to refill, they've made it transparent, and that transparency is allowing, from the reports, thousands of dollars of people's money to be bled out through their Starbucks accounts. And turning it off, turning off auto refill doesn't help because, once a bad guy gets in, they can turn on auto refill.

**Leo:** Oh, yeah?

**Steve:** They may not have - yeah, exactly. And so you need to use a unique strong password. And the advice is, until Starbucks addresses this somehow, and they haven't said one way or the other whether they will or not, I mean, there's a strong incentive for them not to, to make this transparent. But you should, basically, if this is a concern for you, you need to erase any payment methods from the account. Take away any means for refilling the card. It makes it much more troublesome, unfortunately, but it prevents this from happening. But maybe - as far as we know, it's just a password hack. So if you used a really good, unique, strong password, that should be enough. And for what it's worth, my app on my phone, I don't know if it's ever asked me for my password after I gave it to it once. So you can use a crazy one, you know, that you store away in a password manager, because it's not asking you for it all the time. I don't think I've ever been asked after giving it to it once. So that would be my best advice. Or just, if you're really worried, remove any means for it to pull money from some other source, which means you have to do it. But maybe it's good to kind of keep you under control.

**Leo:** Don't use the card. Use Apple Pay. Don't they support touch-to-pay solutions there?

**Steve:** Good question. I don't know.

**Leo:** I'm pretty sure they do.

**Steve:** Oh, then I've got to try it. I haven't yet tried it.

**Leo:** Yeah. Then you wouldn't - then you don't need a third-party card at all.

**Steve:** Although I don't - there are benefits to using the Starbucks system, of course. You get loyalty...

**Leo:** You get free birthday coffee.

**Steve:** You get birthday coffee, and I think every 12 somethings you get a free star, and then you get a - so forth. So there is that. There is a manufacturer, Ubiquiti, which is unfortunately named because their, I'm trying to think, their DSL and ADSL modems are somewhat ubiquitous. And the bad news is that - and it's hard for me to understand in this day and age how this can still happen. But unfortunately it is. They have both HTTP and SSH secure shell management interfaces open to the WAN, open to the Internet-facing side of this router, with default logon credentials. All of them are the same.

So it didn't take long for miscreants to find these modems that had, you know, I mean, it's an HTTP server and a secure shell server. And who knows, they probably found one and looked at the microcode, or maybe figured out one way or the other what the admin logon was. And then they wrote bots which would infect these and, once infected, set up scanners to look for others. So it's considered to be a self-sustaining botnet.

Incapsula was the security firm that reported this. Their scan turned up 40,269 different IP addresses from 1,600-plus IPs in 109 countries, all associated with this botnet and this unfortunately configured modem. For whatever reason, just ISP concentration and ISP choice of a router, 64% of those IPs are in Thailand, 21% in Brazil, 4%, only 4% in the U.S., and 3% in India. So in some cases, in the routers that have been examined, multiple types of malware have been found. I've always thought that, like, the first malware that would infect something like this would close the door behind it so that it didn't have to worry about something else coming along and booting it out, probably literally. But in this case it overwrites chunks of the file system to take up permanent residence.

But there have been multiple instances of malware found. And attribution, as we know, is always difficult. But control of the command-and-control servers, I think about 60 of them have been found located both in China and in the U.S. Some appear to be associated with the group Anonymous. But more recently, Lizard Squad has been talking about how they've got some new botnet under their control. And one of the things they're doing is HTTP query floods, which of course are new style, difficult to defend against. They're the kind of things that the Great Canon was shooting at GitHub that caused it such trouble.

So anyway, you know, here again we've got just poorly configured consumer hardware that is deployed globally and is insecure. And none of this even talks about the fact that 40,000-plus users have their interface between their network and the Internet compromised. So right now apparently this is only being used to form a botnet. But every one of those 40,000-plus users has a vulnerability that will allow anybody to secure shell into their router and have access to their internal network and all the traffic passing back and forth. So that's just not a place that any of those users want to be, either. So, wow. Just amazingly annoying and sort of surprising in this day and age. But I think we've got a lot of surprises in store for us.

Now, the award for another great name goes to "Venom." And there were many overly hyped headlines saying that Venom was even worse than Heartbleed or Shellshock, you know, two other notoriously well named, but much more significant, vulnerabilities. This was found by a guy at CrowdStrike who found the vulnerability and disclosed it to the people who maintain a number of virtual machine technologies, all based on the quick emulator, QEMU.

The flaw is not a huge concern. The thing that made Heartbleed and Shellshock both very worrisome was they were trans-Internet remote exploits. In the case of Heartbleed, we'll remember that it was possible to generate a query that would return as much as 64K of arbitrary memory that the server didn't intend to give you. And the question was, ooh, are there any goodies in that? And we found out, yes, sometimes the server's private key is in that, and that's really not good. In the case of Shellshock, what we learned was that many Internet-facing services use the shell in the background, and that it was possible for bad guys to generate queries which modify the environment variables, which bash would then execute as commands, when bash was subsequently invoked by something using the shell. And it was tricky, but a really bad vulnerability, again, something you could do at Internet distance.

Venom is not anything like that. Venom is a local problem that was found with some virtual machine hypervisor managers, those that are descended from QEMU. QEMU is about 11 years old. 2004 was when it was first released. And it turns out, for those 11 years, there have been some problems, persistent over those 11 years, with its emulation of the floppy disk controller. If you have real hardware on, like, a box sitting next to you, it's got peripherals. It's got a network adapter. It's got a USB physical adapter. It's got maybe serial parallel ports, if it's an older machine. And maybe it's got a floppy disk controller. But for whatever reason, many of these systems still offer a virtual floppy disk controller.

So what happens when you're creating a virtual environment, that is, you're creating the simulation of that hardware, you essentially write software that pretends to be those different pieces of hardware, so that the various virtual machines think that they're accessing hardware. And when the operating system in the VM, in the virtual machine, does physical I/O commands to hardware, the hypervisor intercepts those I/O commands and virtualizes them, meaning that software intercedes and pretends to be the hardware. And that software that's interceding is in the hypervisor. So if there's a problem with that emulation software that, for example, can cause a buffer overrun condition, the buffer that is overrun is in the hypervisor.

And what this allows in theory is malicious software running on one VM to compromise the inter-VM isolation by getting into the hypervisor. So that's what this is. The KVM and the Xen and the VirtualBox VM systems are all based on QEMU and have had this problem. The good news of it being so crazily overhyped is all of them know, and all of them are fixing it or probably already have.

So it's worth, however, updating your environment, if there's a chance that something bad could get in. And that's the reason why I felt this was somewhat overblown, is that most environments are running, most virtual machine environments are running operating systems that the user knows and trusts. They're just not, like, wide open. However, it is the case that there are Internet hosting providers that host VMs. And in that condition, or in that situation, the VM will have root privileges.

And that's another thing I forgot to mention. It must be that you have root privileges on the VM where some malicious code could run to do this. Not vulnerable are VMware, Microsoft's Hyper-V, and the Bochs virtual machine system because they all wrote their own floppy disk controller emulators. And no one has ever seen this thing exploited. This is purely theoretical. That's another reason why there's like, okay, let's not all get too crazy. This needs to get fixed, but those that are based on QEMU will get fixed. And I'm sure this will never happen to anybody. But we do depend upon virtual machines increasingly. And having them robust, and environments that no software in them can break out of, is important.

So from a theoretical standpoint, this was an interesting security problem. But, wow, nowhere near anything like Heartbleed and Shellshock, which required the entire industry to immediately respond and address them. This is like, yeah, okay, you know, it would be good to fix it. But far as we know it's never even been exploited.

**Leo:** How would you - would exploit it with a floppy? You don't need to use a floppy.

**Steve:** No, no. It's just - and that's what my first thought was, well, just disable the floppy disk controller, turn that off. Turns out that doesn't work because, even if you configure the VM not to have a floppy disk controller, if you still write to the floppy disk controller I/O ports, they will be intercepted by the vulnerable code, even if there's no floppy there. And so, yeah, so the idea is...

**Leo:** I guess you could do it with a browser; right?

**Steve:** Well, you would need - you need to be able to do physical I/O.

**Leo:** Oh, you do need physical access. All right.

**Steve:** Yeah, yeah. And so the browser's own sandbox container does not give it the ability to do, like, peek and poke physical I/O addresses. So you would need some sort of environment where, you know, basically you have to have malware in your VM that is developed for this purpose. And so we can sort of think of it, it's very much like the analogy of an operating system where we have a kernel which is in Ring 0 and protected, and we have our various apps which are running in Ring 3. And there's a limit to what they can do. So, and the idea is that the app wants to get into the kernel because it can then do anything it wants with kernel privileges. And the operating system security prevents applications from doing anything that gets in. So vulnerabilities are those little holes, little cracks that are found in that wall that allows something to poke itself through into the kernel.

This is similar. This is a crack that would potentially allow software in a VM to poke itself into the virtual machine manager, which then, I mean - and the virtual machine manager is running at root. It's got full privileges to do everything it wants to with the hardware. So if something could get its code running there, it would be bad. So again, definitely we don't want our VMs to have any ways of breaching them. And if this thing had become known, like, for example, in the underground community, maybe there would be a way through some means of exploiting it.

But it's difficult. The only thing I could see is where those hosting services that host virtual machines where your VM is running alongside many others in the same hardware. For example, we've seen situations with crypto where a side channel attack works because crypto running in one VM is sharing the processor and the processor's cache with all the VMs. So one sneaky VM could, if it were really well designed, could theoretically notice changes in the performance of the processor that it's sharing with the other VMs and reverse-engineer the cryptographic keys that a different VM is processing through the same hardware. That's an example of very subtle cross-virtual machine leakage, but it's been shown to be possible. So the only real vulnerability I could see is if, in a shared VM hosting environment, if this had become known and was exploited, maybe some mischief could have been created.

But the good news is this was responsibly reported. It's funny, too, because the QEMU guys couldn't verify it. They came back to the CrowdStrike guys and said, no, we tried that, that doesn't work. So the guys, you know, okay, here, you know, let me take you through this. And he walked them very - apparently it's two commands out of all the floppy disk controller commands. I mean, and I know them, or did once, because SpinRite used to...

**Leo:** Oh, yeah, of course.

**Steve:** ...write to the floppy disk hardware in order to do floppy disk data recovery. So there are a bunch of commands, seek and read and write and so forth. So some of those commands, given in a certain way, two of them, apparently allow, make an allowance for a buffer overrun. So again, interesting from sort of a theoretical, we-don't-want-leaky-VMs standpoint, but it wasn't any big problem.

Now, okay. So I don't know what to make of this story. There's a guy named Chris Roberts, who is regarded as a white hat hacker, with the company One World Labs. I think he's in Denver, if I remember from one of the stories I read, because there's a lot that's been written about this guy. A few days, about four days before this most recent RSA conference that we've been talking about because a lot of news came out of it, he was on a flight, which I think ended in New York. And while on the flight, he tweets, "Find myself on a 737/800. Let's see Box-IFE-ICE-SATCOM? Shall we start playing with EICAS" - that's E-I-C-A-S - "messages?" And EICAS is Engine Indicating and Crew Alerting System. And then he tweets: "'PASS OXYGEN ON' Anyone?" And a smiley face. Well, that sounds like gobbledy-gook to anyone who isn't deep into the acronym soup which is anything deep like this. But the FBI didn't find it funny, and they were waiting for him when the plane landed.

**Leo:** As they should be.

**Steve:** Yeah, I mean, it's like saying, gee, my bomb didn't go off, I mean...

**Leo:** Yeah. And they don't like that.

**Steve:** I mean, you know, or my underwear got hot.

**Leo:** They didn't arrest him, though. So...

**Steve:** They didn't. They held him for four hours. They questioned him. They did strip him of all electronics. So as he said, he walked away bare of electronics.

**Leo:** Rightly so.

**Steve:** So most people believed that the tweet was a joke, though few were laughing. Subsequently, and I don't - I wasn't clear whether it was during this four hours or afterwards, because then there was an affidavit that was filed by a special agent of the FBI who cited many of the things that Chris Roberts is alleged to have told them. And I'm sure they had a tape recorder running. So, for example, and this affidavit is what surfaced a couple days ago. I think it was Friday. It generated a big hubbub because of what this guy was alleging, I mean, what he himself was telling the FBI special agent that he had done. He said he "...connected to other systems on the airplane network after he exploited/gained access to or 'hacked' the inflight entertainment" - that's IFE acronym; and, by the way, that was one of the acronyms, Box-IFE - "inflight entertainment system."

He stated - oh, it's hard for me to even read this. He stated that he then overwrote code on the airplane's thrust management computer while aboard a flight. He stated that he successfully commanded the system he had accessed to issue the climb command (CLB). He stated that he thereby caused one of the airplane engines to climb, resulting in a lateral or sideways movement of the plane, during one of these flights. He also stated that he used Vortex software after compromising/exploiting or hacking the airplane's networks. He used the software to monitor traffic from the cockpit system.

And in other reporting that I read about this, he said that there's like a box of electronics at every passenger's seat, and that he worked the lid back and forth and wiggled it off and then plugged a CAT6 cable with a modified connector because they're not using the RJ-style connector, plugged it in and then, with a laptop with some special interface and software, proceeded to hack into, get into the inflight entertainment system, which was what was there running through his chair, maybe to control channels and volume and things, and then cracked through the firewall and got into the deeper avionics electronics.

Now, okay. Either way you look at this - and no offense, Chris, but this guy is a moron. First of all, either he did it, which is loony tunes, or he didn't, but he's saying he did.

**Leo:** Which is loony tunes.

**Steve:** Which is loony tunes.

**Leo:** Yeah.

**Steve:** So, you know. And I also - there is some footage of him at - oh, and what happened was four days later he tried to go to the RSA conference on United, which was the same...

**Leo:** On a plane.

**Steve:** Yes, and was blocked. He got through the TSA security...

**Leo:** Really?

**Steve:** ...got to the gate, and then some United officials came up and said, we're sorry, we have the right to refuse service to anyone, and you are refused.

**Leo:** Who claims to hack our jets. We're going to turn you down, yeah.

**Steve:** Yeah, so he took Southwest instead and...

**Leo:** I'm surprised he's not on a no fly list.

**Steve:** I'm just amazed by this. So a warrant has been issued for them to get access to his stuff. They've got, like - he had, like, eight hard drives and some Macs and a few other things that he was traveling with on that flight.

**Leo:** He sounds, frankly, a little delusional.

**Steve:** Well, yeah.

**Leo:** You could quickly verify this because wouldn't the folks on the flight deck, wouldn't the pilot notice if something happened like that?

**Steve:** Yeah, it's like, god, we seem to be drifting to the left.

**Leo:** Why are we slewing left? What's going on here? So I think you could quickly verify this. I would guess almost certainly the guy's delusional, probably wants attention.

**Steve:** Well, as I was going to say, in the footage - there is some footage online, and I've seen the transcript of it. And he's got, I mean, he's old enough that you would think he would have outgrown that sort of style that some hackers have of, like, impressing their mothers, you know? I mean, where they say things that someone who doesn't know technology would be impressed by, like, well, I got partial, partially penetrated the firewall. It's like, wait. Partially penetrated the firewall? Okay, what does that mean? You know, and then I slid to the side and got in touch with the other firewall. And, I mean, anyway, I'm giving a bad example.

But if you read it, it just - it doesn't read credible. It doesn't read like somebody who understood what it was he did. Even if he did do something, it wasn't like he's able to articulate it in a way that sounds like he knows what he was doing. So I don't know what to make of it. And as you say, Leo, you would think that he would have left a trail of evidence, if something of that magnitude was done. But also, who was going to - first of all, I have a hard time believing he overwrote code on the airplane's thrust management computer.

**Leo:** Seems like that you would need certain permissions to do. To do it from the flight entertainment system seems odd.

**Steve:** However, I will say I believe this notion because many security experts know that you can hack into the plane from the entertainment system.

**Leo:** Well, that's terrible. That's like the cars with the cheap CAN bus that we've been talking about.

**Steve:** They did not, yes, they did not do this right. They engineered it with hubris, believing that they could put a firewall of some sort between the two, but for weight or economy or convenience or shared topology, who knows what. But it does appear to be very credible that from the passenger cabin you can access flight avionics. And I feel there's no excuse for that. That's just...

**Leo:** No, that's a serious problem.

**Steve:** And in fact Winn Schwartau, I can't quite remember the name, he was like one of the top 25 named top security researchers. I just saw before the podcast, I was reading up a little bit more on this, he is suggesting that all inflight entertainment systems be disabled until a thorough investigation can be made because, I mean, real researchers know that planes have problems. And so this guy may not be the real deal. It's just very difficult from what we know to judge. But it absolutely appears to be the case that these systems are not secure. And then in some sort of a PR stunt, I guess, United has started offering mileage, what are they called...

**Leo:** Miles, yeah, yeah.

**Steve:** Yeah, 25,000, 50,000, or 100,000, maybe as much as a million, I think I read, depending upon the severity of what is found, for hackers providing information about defects in their inflight systems. And I was thinking...

**Leo:** That just encourages people to get on the plane and hack it.

**Steve:** I'm saying no.

**Leo:** How else are you going to test it?

**Steve:** I'm going to drive to San Francisco next time.

**Leo:** Oh, my god.

**Steve:** It's like, now you...

**Leo:** That can't possibly be true. Is that true?

**Steve:** Yes. Oh, it's absolutely true. It's been covered, I'm sure if you google the news right now...

**Leo:** Are they offering their code to somebody to analyze? Or they're saying, oh, just hop on a plane and let us know?

**Steve:** I think that's the latter.

**Leo:** They can't possibly. That's insane.

**Steve:** I know. But google "United mileage hacking," and I'll bet you'll pull up stories.

**Leo:** It's like a bug bounty. Which makes sense on Facebook.

**Steve:** Yes, for a plane.

**Leo:** But on a plane?

**Steve:** I know. I know. I don't understand this at all.

**Leo:** Both Boeing and Airbus say that the - have said many times that the inflight systems are not reachable from the entertainment systems. But they don't - that doesn't mean - they may say, oh, we've...

**Steve:** That could mean firewall.

**Leo:** We have a firewall, right.

**Steve:** That might mean your mother can't get it.

**Leo:** Right, right.

**Steve:** But it doesn't mean that, you know, somebody who really knows what they're doing can't.

**Leo:** No, no, wait a minute. I think that United's clarified this now, according to Sophos. They're saying "hack our site, not our planes." Hack our site. Thank god.

**Steve:** Okay, good.

**Leo:** They're saying "Hack our site for free miles, but don't mess with the onboard systems, please." Oh, lord.

**Steve:** Oh, yeah, wow.

**Leo:** "Bugs that are not eligible for submission: onboard WiFi entertainment systems or avionics." Yeah, if you do that, I'm sorry, you're not going to get free miles. Come on.

**Steve:** Yeah, you'll be lucky to land.

**Leo:** Well, that's what's really crazy about this. It's almost a suicidal thing to do.

**Steve:** I just, you know, just...

**Leo:** Not to mention endangering everybody else.

**Steve:** There's nothing you want more than, like, nobody who is in the passenger compartment with you to be messing around with their flight entertainment system.

**Leo:** And did no one notice him pry the lid off the box and plug an Ethernet cable into the entertainment system? Maybe they...

**Steve:** What would be nice would be to know if this is credible. And I don't, you know, I've never poked around my seats to check it out. Although apparently you have to have a video display in front of you, and I do little short flights, and so...

**Leo:** Well, I've seen those boxes. The ones I've seen are not in every row, but they are spread throughout the cabin. They're humps.

**Steve:** Nodes, nodes.

**Leo:** And they're nodes. And I've seen that before. I've even seen flight attendants open them up and go in them when they're having trouble. But if a passenger did that, don't you think that would raise an alarm?

**Steve:** Eyebrows, yeah. Wow.

**Leo:** I feel like he's just - we know these guys. There's guys who want this kind of attention.

**Steve:** Well, but the fact that he was speaking at RSA then surprised me, too, because that's - you'd think they would do some vetting of people.

**Leo:** Well, maybe he's a legitimate expert, but also likes attention.

**Steve:** Yeah, yeah. Okay. So we need to get GetShine.com, G-E-T-S-H-I-N-E dotcom, because this is a story that VentureBeat picked up from Financial Times. Unfortunately, the original Financial Times article is behind a paywall that I was unable to penetrate.

**Leo:** I know, I hate that.

**Steve:** Yeah. So all I can do is quote what VentureBeat said. But they said: "A story reported by the Financial Times states that 'several' carriers have installed ad-blocking software" - a these are mobile carriers in Europe - "developed by an Israeli company" - who actually has offices in Sunnyvale, California - "called Shine in their data centers, and plans are afoot to switch the technology on by the end of the year. The software would stop most web-based ads from loading..."

**Leo:** [Mild agitation]

**Steve:** I know, "...though 'in-feed' ads like those on Twitter or Facebook would not be affected. Citing a source at one European carrier, the report" - the Financial Times report - "suggests that the network will introduce an opt-in, ad-free service initially, but is also considering extending it to its entire network automatically. It's not clear whether this would be a paid or free offering, but ultimately it's designed to target the major online ad companies such as Google."

And so on that site that you've brought up and you're looking at, the banner there is "We champion the consumer's right to control mobile ads." And then they say, "Advertising technology has gone unchecked, polluting our web and app experiences with privacy-infringing and obtrusive advertising. We work with mobile carriers who are redefining their services to meet the true needs of consumers by offering the power of ad control to millions of subscribers around the world."

And to me this is interesting because this is sort of the next notch in a coming skirmish that we're going to be having. The mobile carriers are complaining that Google is generating tremendous revenue for bandwidth that they're providing. You know, when have we heard this story before? This sounds very much like Cogent and Netflix or any of the, you know, the whole Net Neutrality and peering problem.

And if you saw, well, I know that you saw it because I heard you gasping, Leo, their example of sliding ads or preventing ads from coming up on mobile phones. They've got the technology. And it isn't, however, clear whether this will be able to work over HTTPS or not. And consumers tend to have less control over their mobile phone than consumers do currently over our desktop and laptop computers. But it'll be interesting to see how this pans out.

**Leo:** It's interesting to note that one of the examples they give is a free game, 2048, that's ad supported.

**Steve:** Yes.

**Leo:** I've got to think, if you're the developer of apps, free apps that are ad supported, this is bad news. This means you're just going to do paid only. And the unintended consequences of this may reach far and wide, not just to Europe. Europe never liked Google; right? They just hate Google. They just hate them.

**Steve:** They really do.

**Leo:** Yeah.

**Steve:** Yeah. So it's like, well, okay. Again, we live in interesting times, and it'll be fun to see how this all pans out. This is a little quickie that I got a kick out of, which was that a security researcher, EgyptianGeeks.com, discovered a pretty simple exploit, I think I heard you talk about it on TWiT on Sunday, on ESET's website. Those are the guys that do the NOD32, which is a well regarded AV suite. They sell, for $29.95, an annual license for NOD32. The researcher looked at the way their ecommerce license authorization system was working, basically did a "view source" on the page, poked around in the code, realized that it was trivial to bypass, created a bypass so that you didn't have to pay anything. You just, one click, and you got a NOD32 license mailed to you, good for a year. And then, being responsible, after playing with this for a while and getting bored, he reported it to ESET. And it turns out they have a bounty program. They offered him a $30 annual license to NOD32 in thanks. And he said, that's okay, I've already got one.

**Leo:** I got a few - hundred thousand.

**Steve:** I got it.

**Leo:** Yeah. They were a sponsor for a long time.

**Steve:** I remember.

**Leo:** And it's a great antivirus. I love it. Obviously whoever designed their site, probably not them, left a little flaw in there.

**Steve:** Yeah, I mean, you know, companies are big, lots of people, and left hand not knowing what the right's doing. And probably the people, there are probably really good techies down in the depths of reverse engineering. I mean, to do AV you've got to know your stuff.

**Leo:** Yeah.

**Steve:** To do a website, eh, let's go see what this site did and copy and paste and, you know, sort of...

**Leo:** Oh, I'm sure they went to a third party to do the site. They didn't even do it themselves, I'm sure.

**Steve:** Yeah, yeah. So we talked about how some of our listeners, or visitors of GRC, had been reporting that TrueCrypt was generating errors when people were trying to download from my archive of TrueCrypt 7.1a, both in Firefox and in Chrome, because both of those use Google's site reputation technology. The code has never changed in all these years from what it originally was. The hashes are the same. My code is the same as is available in other places, other archives on the Internet. So it's just a false positive.

Then I got another report saying that Google was saying that the site was suspicious. And it's like, whoa, whoa, whoa, what? So I poked around, and there's something, Google Web Services or Web Developer Services. And so I went there. I hadn't been there for years. But Google knew me because I have a, you know, Google knows me. And I associated GRC.com and www.GRC.com with those web services by accepting a file from Google, which I stuck on the root of GRC's server to prove that I had control of the server. Then it said, oh, okay, you must have access. Now we'll tell you what we found.

What they found isn't even publicly linked anywhere. I don't know how Google found it, but that's Google's job. It was called "fp.exe." It is more than two years old. It was dated March of 2013. It's been there since then. It's a Windows console app which I wrote when I was developing - "fp" stands for "fingerprint" - when I was doing the SSL fingerprinting work and prior to creating GRC's fingerprint demonstration page, to explain about SSL fingerprints.

So it's been sitting there for two years. Somehow Google saw it. It's 8K. But it's compressed with UPX, which is also the compressor of choice for many malware authors. I've always compressed my code with UPX because Microsoft's, it's called the PE, the Portable Executable format, is so wasteful of space. It's just got runs and runs of zeroes, and it just, I mean, my point is that it is an incredibly inefficient format for executable programs, thus highly compressible. So I want my stuff to be small. I don't want it to be ridiculously large, so I use UPX to compress. And we've talked on the show from time to time, some random AV program will trigger and decide that something that has been around forever, like Leaktest, that has seven million downloads, that, oh, it's a problem. It's like, well, okay, it hasn't changed in 12 years. I don't think that's a problem.

But anyway, I removed fp.exe, since I certainly don't need it around. It was just there because I didn't delete it before, and it had been sitting there for two years. I informed Google, asked for a reanalysis, and they said allow 24 hours. And that was, like, 72 hours ago, and they're still saying something's wrong. What's weird is they're saying, when I look at this detection tool, the page reads, "Google has detected harmful code on some of your site's pages. We recommend you remove it as soon as possible. Until then, Google's search results might display a warning to protect users when they click a link on your site."

And then, as details, it says "Undetermined malware. These pages directed users to a site that serves malware or unwanted software. Unfortunately, the malicious code within the page could not be isolated." And then they, like, and initially they showed this one file under the miscfiles directory. And then I deleted it, and I told them I deleted it and that it was never malicious, blah blah blah, and so far there's been no change. But anyway, that's the story there. So I've never had any problem with Google before. I don't know how this happened. I'm just waiting for them to clear it up.

**Leo:** I think, I mean, that's the problem with automated scanners; right? And there's no human there. It's just a machine.

**Steve:** Well, and they own VirusTotal; right? And zero out of 56 of the AVs that are all aggregated there think I have any problems. So who knows. But you're right. It's some sort of a heuristic that something set off. And I imagine it will get itself cleared up. Maybe somebody at Google.

**Leo:** Yeah, somebody listening at Google will fix it.

**Steve:** Yeah.

**Leo:** Or contact you and explain what's going on. That would be useful, too.

**Steve:** That would be - I'd love that. That would be great.

**Leo:** Do you use robots.txt on your site? I mean, do you restrict...

**Steve:** Oh, sure, sure. And I think miscfiles is in robots.txt.

**Leo:** It should be excluded; right?

**Steve:** Yeah. I'll check.

**Leo:** I wonder if they scan, even if you say don't include it in your search index, if they still scan those.

**Steve:** That, I was thinking the same thing, too. They may still go in there. Although, I mean, the idea is you want them excluded because they can get into trouble there. That is, the presumption, one of the presumptions is that, if something went into an area of your website, for example, it might never come out. I mean, you know, like you have dynamically generated pages. It would be some sort of a problem. So they're supposed to honor it.

**Leo:** Yeah, or you just don't want it in the search index. You know, I'm using this as a storage directory for my server. It's not publicly accessible, and I don't want you looking at it, either.

**Steve:** Right. And that is the link that they - or the program they were complaining about. But there may have been a link, there may be a link somewhere else, like maybe in some...

**Leo:** Something must link to it, and they were spidering.

**Steve:** In the newsgroup posting from back then, I gave everybody the link because a bunch of people were playing with this and testing it for me. It's the only reason I would have put it on the public server is that I was saying to a group of people in the newsgroup, hey, guys, check out this little console app, see if it works for you.

**Leo:** It wasn't a publicly viewable directory otherwise, though.

**Steve:** No. No.

**Leo:** The other thing is of course bad guys use file compression to hide virus strings.

**Steve:** Yes.

**Leo:** So once you've compressed it, then a malware tool can't see that it's a virus inside there.

**Steve:** Yes.

**Leo:** So the very fact that you used a compressor is probably all it was. That's the red flag.

**Steve:** I think, yes. And it wasn't signed because it was just an R&D - I was just, you know, spinning this thing out, and it was an R&D program. Any code I publish now I digitally sign. And so maybe, if I had signed it, then Google would have said, oh, you know, this is probably legitimate.

**Leo:** Yeah, because no hacker would sign his code.

**Steve:** On the other hand, the signature adds 4K, and it was only 8K. So it would have increased the size of my program by 50% just to sign the darn thing. Come on.

**Leo:** Wow.

**Steve:** So I just, without any spoilers, and we need to be more careful about that than you were on TWiT because I really don't want to do that, but I wanted to tell you I instantly agreed with you totally about the unbelievable plot mistake at the end of "Ex-Machina." The moment that she went where she went.

**Leo:** Like, where are you going?

**Steve:** The first thing I thought of was...

**Leo:** What are you doing?

**Steve:** ...you can't leave.

**Leo:** What are you - where are you - well, maybe, you know, I was thinking about it. It's a world where inductive chargers are widespread. That's all I can say.

**Steve:** Maybe she'll sit on one of those pads somewhere.

**Leo:** She says, oh, Leo's got a charger for his phone. It may take a little longer.

**Steve:** Stick it under her arm. Oh.

**Leo:** But what did you think of the movie? Did you like it?

**Steve:** I came away disappointed.

**Leo:** Yeah.

**Steve:** And for a couple reasons. And these are not spoilers. This is the kind of movie where I will like it much more the second time.

**Leo:** I think you're right.

**Steve:** Because my expectations are now calibrated down.

**Leo:** Right. We expected a lot from it because it got such raves.

**Steve:** Yes. And I also think I live so much in a sci-fi environment that I'm not the audience for this.

**Leo:** Right.

**Steve:** This is so obvious to me. And I've read the plot in so many different contexts that it's like, eh, okay.

**Leo:** Just another one.

**Steve:** Great special effects.

**Leo:** Yeah.

**Steve:** I will say, and I've compared notes with a couple other techie friends of mine because I haven't remembered to talk to you about it until now, no way did I buy that the boss was capable of designing those.

**Leo:** Or building them.

**Steve:** He did not sell me on that at all.

**Leo:** Yeah.

**Steve:** I thought his acting was absolutely awful. The young programmer, I thought...

**Leo:** He was just a pro.

**Steve:** ...did a great job.

**Leo:** He was a programmer.

**Steve:** Yeah, I just - it's like, you could not have created this technology, you know, in between being drunk.

**Leo:** Yeah. Yeah.

**Steve:** So, yeah.

**Leo:** And it's not just software. The guy was building things; right?

**Steve:** Yes.

**Leo:** I mean, is he a brilliant - he's, like, brilliant in every - by himself? Come on.

**Steve:** Yeah, yeah. I just - that was a problem. Beautiful hotel, though. That's actually a hotel in Norway.

**Leo:** That's a hotel?

**Steve:** Yeah.

**Leo:** In the woods.

**Steve:** Yeah. They did it on a super low budget. Charlie Rose interviewed the writer/director a few days ago.

**Leo:** Oh, I'll have to see it.

**Steve:** And it was a pretty good interview. I would have reposted it on my new YouTube channel, but after finishing I thought, well, he really didn't say that much. And I was surprised, though, that he was as literate about the whole issue of the questionable safety of creating something self-aware. And you would hope he was. But the point was they did it on a super low budget. I imagine they're making a lot of money. I haven't - I meant to go check. Is it doing well in the box office?

**Leo:** I don't know. But it's certainly worth seeing, if you listen to the show.

**Steve:** Yeah.

**Leo:** By the way, there you go, there's a link, a clickable link right to FP, so...

**Steve:** But that's not it.

**Leo:** That's not the one?

**Steve:** No. That's the dev subdirectory. This is a miscfiles, M-I-S-C-F-I-L-E-S. Yeah. And that would be...

**Leo:** Oh, all right. All right. But this is an open directory. So if you've got a lot of open directories, that's, yeah.

**Steve:** No, in fact the dev subdirectory tree is the only one I have, and it's specifically so that people can go...

**Leo:** Oh, okay. All right. Yeah, right.

**Steve:** For example, there are, if you click up and then go into SQRL, you'll find a SQRL client that works.

**Leo:** Ah.

**Steve:** For example. That's where that stuff is. So there's the one I posted...

**Leo:** So that's intentional. This is all intentional.

**Steve:** Yeah. That is deliberate, yeah.

**Leo:** And certainly not illegal or in any way indicative of malware.

**Steve:** No.

**Leo:** No.

**Steve:** No, that's all stuff I've hand created.

**Leo:** Yeah.

**Steve:** Okay. Next is this site is very cool, the one below, the logic gates without scripting? It has no bearing except that I bet our listeners would get a kick out of it: silon.slaks.net. And it's just a - there's no script there. So if you click in the upper left on the menu at the top, Leo...

**Leo:** Oh, neat.

**Steve:** This is CSS only, operating logic gates. So you can click on the inputs, and I'm not seeing it change. Oh, there you go. So AND, OR, XOR.

**Leo:** Oh, look at that. That's cool.

**Steve:** Yeah. And no code, just using CSS. So very much like how GRC's menu system does the dropdown multilevel menus. And there's a half-adder, and he has a full-adder. So it's just sort of a cool little gizmo to sort of play with: silon.slaks.net.

**Leo:** You get a smart 10 year old in here, and next thing you know he's going to have artificial intelligence designed. Here we go.

**Steve:** I hope not.

**Leo:** Wow, that's cool. That's really nicely done.

**Steve:** Yeah. Very nice. And no code, just using the power of cascading style sheets.

**Leo:** Oh, look, some - ooh. Ooh. Oh. I like that.

**Steve:** Yeah, he did a good job.

**Leo:** Yeah, neat.

**Steve:** So I also tweeted, and I wanted to put it in the show notes and let our broader audience know, I found an interesting 12-minute audio discussion, it's just an MP3, between some security researchers. I want to say that one was Graham Cluley, but I'm not - I think so, but I didn't write it down, and I'm not sure. Anyway, I created a bit.ly shortcut, bit.ly/UKbackdoor. And this is a discussion that was from earlier this year, following David Cameron's intentions as he stated, you know, his position of there should be nothing that the government cannot listen to. And this is just 12 minutes. But really, I commend everyone to it, really worth listening to, I think, because very smart people, and even some smart but, you know, I think misguided people on the government side just not getting the fact that giving the government access to all communications is just going to be a really bad idea: bit.ly/UKbackdoor, with "U" and "K" capitalized.

**Leo:** Cool. Is it a podcast, do you think? Or...

**Steve:** No, because it was a snippet from - it was a news story.

**Leo:** Ah, okay.

**Steve:** And so it was - and there is a moderator. So there's a moderator who's moderating this discussion between a number of experts on both sides of the question. But definitely worthwhile.

**Leo:** Good.

**Steve:** Yet I don't think quite worth playing into the podcast's audio. I mean, not worth playing into our audio. So I'll just let people get it if they're interested.

I did want to mention that I achieved a milestone yesterday with SQRL. The work I've been doing for about six weeks I wrapped up. I updated the code on the server, published a new client. I finished sort of the reconceptualization that I had referred to earlier. Of course SQRL is all about managing your identity. And essentially there were two ways you could do this. You could - that is, in terms of a user interface. You could press buttons on the website that had your identity, things like if you want to change your identity, delete the identity, like remove it completely from the server, update it. We have a means of allowing you to disable the use of SQRL, which you then require higher privileges to reenable. I'll explain all this when we get SQRL finished.

But the point is all of that could have been mediated on the website side. And that's sort of the first way I had implemented it, where there were buttons that took you to different pages, that asked you things. And the idea was that you were constantly being prompted with a QR code to reaffirm that this is really, for important things, that this is really you wanting to do these things. So you were reasserting your identity.

And as I got closer to it and looked at it, I realized this was the wrong place to put that; that, with a tiny change, essentially adding one verb and a different type of error message to the protocol, the client could do all of that so that the site just presented passively a QR code that the client could use with the identity management over on the client side. And the advantage is, whereas every site on the planet does things differently, you know, we're always having to figure out, okay, how do I manage my account, where do I click to get to account management. And there's no homogeneity to that. It is just everyone makes it up.

So I decided moving that into the client would offload the burden from all of the repetitive effort on websites and only needs to be implemented in the client once. Moreover, the user gets a consistent experience for managing their identity. They can disable it. They can remove it. They can update it. Everything that they want to do is contained within the client. And basically the server is just complicit in that. So all of that's done, and the gang in the SQRL newsgroup have been pounding on it for more than 24 hours. So far it is surviving.

I have one last feature to add, which is a provision that a lot of people have wanted, which is for the server, the website to be able to confirm something critical. Like say that you were on a banking site, and you wanted to transfer a lot of money somewhere. That's scary because web pages are just not secure. So the SQRL system allows the site to say, we're going to do something, and we want extra confirmation. And so using the super secure SQRL channel rather than the website, the site is able to put up a prompt, like on your smartphone, to say, hey, you're telling me that you want us to send $10,000 to this place. Are you sure? And the point is, by being out of the website, out of the whole web browser channel, we hugely improve the security of anything that a site wants to do that it wants to absolutely verify has not been intercepted in any way. And that sounds like a lot, but it's actually a very trivial bit of UI code.

So that's the last feature. I'll get that in, then clean up a bunch of user interface stuff like default pushbuttons on dialogues and things, and controls that have the focus when you move. There are some things that seem wrong. I fix those, and then we go over the text that's in the app to make sure that we're happy with the way I've worded everything, and it's done. So we're getting close. Yay.

**Leo:** That's amazing. Congratulations.

**Steve:** And then we'll explain it all to everybody.

**Leo:** Yes.

**Steve:** And I do have a nice story from an Israeli listener of ours, Leo. Nir Yaniv, who's in Israel, said "Overdue SpinRite Success Story." He said, "Hi, Steve. I've been listening to Security Now! for a few years, but I knew of some of your work before that, mostly thanks to ShieldsUP!. Upfront, thanks for everything you and Leo do. I learn so much every episode, and I use and recommend your products and services to everyone I know or even randomly encounter."

He said, "SpinRite success story: I purchased SpinRite mostly out of appreciation for the podcast. I have never used it to fix or even maintain one of my personal computers, but I have used it a few times in my previous work as systems administrator. I successfully used SpinRite to recover a few ESXi servers which refused to boot or failed to load their VMs. But the real success was this: We had a network traffic generator and analyzer for various tests. This was one of the more expensive systems in the lab network I managed. One day it crashed," he says, "I think it was a power failure, and refused to boot. I don't recall the specific error, but the Windows operating system underneath simply wouldn't boot. We had a support contract for the device, but the only solution offered was sending a replacement drive from abroad, which would have taken a couple of days.

"So I took out my copy of SpinRite, booted the machine from it, and let it run overnight. In the morning I rebooted the system. Lo and behold," he wrote, "it booted and resumed normal work. Obviously, I had my boss set aside budget for four copies of SpinRite after this. We ordered the replacement drive, and it stayed in a closet as a backup for the next crash, which hadn't occurred by the time I finished that employment." So, Nir, thank you for the report.

And for anyone who's wondering, the four copies is the way we handle corporate licenses. The idea is you buy one copy, and you can use SpinRite with our full sanction on any drives you as an individual own, and you know that we don't ever complain if you help friends. But for a corporation, where there's lots of machines, that seems a little unfair to us. So we just ask you to have four copies of SpinRite. And the reason is then it's simple to handle upgrades.

**Leo:** Oh, I see what you're saying. That's the multi-seat license is four.

**Steve:** Exactly. And then, for example, when I do SpinRite 7, there will be some upgrade cost. And so a corporation would just upgrade their four copies, and then they would have upgraded their license. That just sort of simplifies things for everyone. So it's a little screwy. No one else has ever done it that way. But it just sort of made sense to me, so that's what we do. So.

**Leo:** Yes.

**Steve:** Okay. This is so fun. They are known as Passive Keyless Entry and Start in the industry, PKES.

**Leo:** Okay.

**Steve:** And this is sort of the evolution of user convenience, unfortunately at the expense of security. I believe we're in a valley of security that we will come out of because this has become a serious problem, and it has a solution. But it's going to require a next generation of PKES, of Passive Keyless Entry and Start systems.

**Leo:** So here's my PKES. Inside this is a battery. I know because I just changed it.

**Steve:** Yup.

**Leo:** I don't have, you know, it has, like many keys, the lock/unlock, open the trunk, panic. But it also - all I have to do to get in the car is have this in my pocket.

**Steve:** Yes.

**Leo:** And I just touch the handle, and it goes click, and it opens.

**Steve:** Leo, you are - I couldn't have had a better foil.

**Leo:** And, by the way, now, it doesn't automatically start. I still have to press a button.

**Steve:** On the car; right?

**Leo:** On the car.

**Steve:** On the car, okay. Yes. Anyone who wants to can take your car and drive away with it.

**Leo:** No.

**Steve:** I'll explain how.

**Leo:** And then there's this other car, the same thing. Oh, man.

**Steve:** Yeah. That's why I'm glad you guys have Segways.

**Leo:** Most modern cars do this, by the way.

**Steve:** Yes. So it's funny because I've seen the progression. You know, I have an old car with low tech. I've mentioned before that I'm happy with it. And it's an '04, 2004. And when I take it in for its service from time to time, they give me a rental. They have, like, a rental facility as part of the service. And so I get, as if they're trying to sell me the latest and greatest, I get, like...

**Leo:** Yeah, wouldn't you like a new one?

**Steve:** ...a low mileage new version.

**Leo:** Yeah, of course you do.

**Steve:** And so for a while - and so I've watched it move along. For a while there was - I had to take this key that no longer had teeth on it of any sort and, like, stick it in a hole in the dash. And so that was for a couple years they were doing that. Then most recently when I had the car serviced, it was just, you know, you just sort of have it around. It's in your pocket. It's just like yours, Leo. It's in your pocket. You leave it in the center console, wherever. And the systems...

**Leo:** You know what's funny, my old Lexus had really two-factor because not only did you have to insert a key and turn it, but if the key was not chipped, it had a chip in it, it wouldn't matter.

**Steve:** And that's what I still have.

**Leo:** And that's two-factor. That's great.

**Steve:** Yes, I still have that.

**Leo:** Well, it's not exactly two-factor because it's something you have in both cases.

**Steve:** It's got a little wiggly key slot. And then but it has a key. And in there is an RFID chip which is being probed in order to keep the key from being duplicated.

**Leo:** And nobody ever stole my Lexus.

**Steve:** No. So just so that people understand this is real, in Toronto, since January, police have seen a spike in thefts from Toyota and Lexus SUVs parked in their owners' driveways with no sign of damage.

**Leo:** Oh.

**Steve:** In Los Angeles, a reporter noticed the same problem in his own neighborhood. He watched two teenagers on bicycles open his locked Prius as it sat on the street in front of his house.

**Leo:** That was Nick Bilton, by the way. That was the reporter.

**Steve:** Yup.

**Leo:** Many times a good friend of TWiT's.

**Steve:** In Tonawanda, New York, people who swore they had locked their car doors returned to their vehicles to find them rummaged through, spare change gone, but otherwise undamaged. In Springfield, Missouri, a rash of vehicle break-ins had residents wondering how thieves could go through so many cars leaving no damage. And in Long Beach, California, detectives shared a video on YouTube showing break-ins of two SUVs parked in their owner's driveway.

And remember, we teased this last week with Q&A Question #10. Our listener Gary Beals said, "Many of my neighbors are reporting on NextDoor.com that they locked their car doors and came out the next morning to ransacked glove boxes. I of course thought of you and knew you could explain how my neighbors' car security systems are getting bypassed. So I was pleased to hear that you are way ahead of me and are planning an episode on the topic. In your discussion, could you please discuss mitigation strategies? A recent CBS This Morning news segment suggested storing car fobs in a shielded box (your fridge). Can you also indicate whether the same hack works on garage door openers? Should we store our garage door openers in the fridge when they're not in use, too?"

Okay. So I found some great detailed research about exactly how these systems work. And it's - okay. So it's an interesting side effect of some of the, well, much of the convenience, but different aspects of the convenience. So as I mentioned at the top of the show, three Swiss security researchers at the Department of Computer Science at ETH University in Zurich, they analyzed 10 different car models from eight different manufacturers. They were able to open and drive away all of them. So complete breach. The attacks they developed allowed cars to be opened and started - and, now, here's a key - when the car's key was up to 50 meters, 164 feet away, non-line of sight. So you'll notice that a common theme in these stories is the guy watched his Prius being opened out on the street in front of him, or people's cars in their driveways were ransacked. The point is the key is - it still exists, but it is way further away from the car than should still allow the key's effect to be had, essentially.

So all of the systems used the same radio architecture. They all used advanced challenge-and-response crypto, meaning that all was done right. So the car periodically sends out a ping. In some cases the ping is just a wakeup for the key. If the key acts, acknowledges the ping, then the car will send the cryptographic challenge, which the key will respond to, to prove that it's the correct key. So that's a four-way handshake. However, some of the designs just used a simpler two-way, where a cryptographic challenge was constantly being sent, and only when it was received was the key acknowledged.

Now, a big part of the problem with this is range. And longtime listeners of the podcast will know that I always have what we've come to call a "Gibsonian response" to the whole idea of radio. Radio is just creepy. It's scary from a security standpoint. We've talked about, for example, the assumptions made with RFID tags, that they only have a certain distance. But then we see that, if you use a high-gain antenna, oh, what do you know, you can ping somebody's RFID tag in their passport a hundred feet away, even though RFID is supposed to be a few centimeters.

Again, the problem is radio itself is a boundless radiative technology. And so if you have enough gain in amplifiers and in antennas, then nothing bounds the distance at which it can operate. And that's the key Achilles heel of the systems we have currently, that is, they assume that, if the key is close enough to have this transaction, then the driver, the owner of the key is physically proximate to the vehicle, enough to be responsible for what happens so that, when you reach for the door handle and pull it, and the key and the car have already been talking as you've been getting very close to the car, the door is unlocked, and you're able to get in.

So the architecture is as follows. All of them share. The car emits a low-frequency, inherently short-range, RFID-style ping. And that's in the region of 120 to 135 KHz. So the car emits low-frequency, short-range. It requires a relatively large amount of power to generate a low-frequency, short-range signal. You need a larger antenna and/or more power. But cars have monster batteries and lots of power compared to, for example, the battery that can fit in a key fob. So the key fob uses - whereas the car generates a short-range, low-frequency query, the key fobs universally use a UHF, as opposed to LF, low frequency, these use a UHF ultra-high frequency which is inherently longer range, like on the order of a hundred meters transmitter in order to respond. Now, the reason they do that, there are a couple. One is for reliability of transmission, but also because, as Leo, you mentioned, your key also has some buttons. And so...

**Leo:** Yeah, but it doesn't need them, does it.

**Steve:** No. But what has happened is the manufacturers wanted multifunction. They wanted you to be able, at a great distance from your car, to be, you know, if someone says, you know, like you've walked away, and someone stayed back there and shouts, hey, can you open the trunk, you want to push the button and, at a great distance away, because you're taking responsibility for pushing the button, the car receives this much longer range transmission and pops the trunk.

**Leo:** I use this, yeah, to pop the trunk. Or when I can't find the car in a parking lot, I press the panic button, and then the car makes a lot of noise, and I go, oh, there you are.

**Steve:** Yes, exactly.

**Leo:** But I would say it works about, if it's line of sight, maybe a hundred feet.

**Steve:** Yeah. Okay. So the point is this is very asymmetric. The key can be heard by the car from a long distance away. And in these security guys' tests, they were seeing up to 100 meters, but their tests all ran at 50, so about, what, 160 feet.

**Leo:** Yeah, that's probably right, yeah.

**Steve:** Right.

**Leo:** Like WiFi.

**Steve:** Yes, exactly. And in fact it is 315 or 433 MHz are the two frequencies that all of these keys use. They have a small battery, but that UHF signal can reach that range. And of course the car can be very sensitive because it's got as much power as it needs to be listening carefully for its key. So the car can hear the key at a great distance. But the key cannot hear the car until it's right up within a couple meters, typically, maybe even one. So the key has to be much closer. So there's this - oh, and the cars also differentiate the key being outside the car and inside.

**Leo:** Yes.

**Steve:** That is, the car is, on the outside, the car is pinging with one code. And inside it's pinging a different code. So if the key responds to the outside code, the car knows the key is not yet in the passenger compartment. When the key preferentially responds to the inside code, then it knows the key has made it into the passenger compartment, and that enables the start button, you know, the engine start function.

**Leo:** Oh. So I could be sitting in the car, but if the key's outside the car, I can't start it.

**Steve:** Correct.

**Leo:** Ah. That's good.

**Steve:** Yes. I mean, all of this sounds good so far. So...

**Leo:** Now, to open the car I usually touch the handle. I think, I feel like there's some capacitive or conductive thing going on. But that must be my imagination.

**Steve:** Or it might, that might be an additional level of security.

**Leo:** It must be something Audi's doing because I get up close. It's not unlocked until I touch the handle, and then it goes "chunk."

**Steve:** Makes sense, makes sense.

**Leo:** Yeah. That might ameliorate this problem; right?

**Steve:** That won't help you, no.

**Leo:** Oh, crap.

**Steve:** No. Okay. So in their 15-page, beautifully detailed report - the PDF, by the way, is in the show notes, so anyone who wants more can see. But I did copy some of the diagrams they have because they're great. They wrote the perfect paragraph. I couldn't have done it better. They said: "We note that the main reason why relay attacks are possible on PKES systems is that, to open and start the car, instead of verifying that the correct key is in its physical proximity, the car actually verifies that it can communicate with the correct key, assuming that the ability to communicate implies proximity."

**Leo:** Oh, I see the problem.

**Steve:** Yes. "But this is only true for non-adversarial settings. In adversarial settings, communication neighborhood cannot be taken as proof of physical proximity. Given this, any secure PKES system needs to enable the car and the key to securely verify their actual physical proximity. This is only natural, since the car should open only when the legitimate user, holding the key, is physically close to the car.

**Leo:** I'm really - I feel like I wish I had a Faraday cage here or a bag or something I could put my keys in. I'm worried.

**Steve:** Well, people are, I mean, this is increasingly prevalent as higher end or more recent cars are supporting this technology. So the assumption which has been broken is that the low-frequency, short-range communication is range limited. And it isn't. These researchers set up a point-to-point link between the car and near a user's keys.

What they did was they - so they had two units. It's also possible to do this with one unit, although it requires more power. They used two units. Essentially, they had a receiver near the car with a low-frequency loop antenna that would pick up the pings that the car was emitting. They captured that, and they up-converted it from its low frequency to create a 2.5 GHz radio link, just because they had the hardware around to do that. Then they made a second unit which was a 2.5 GHz receiver, which simply down-converted by the same multiple the 2.5 GHz link back down to low frequency and had an antenna to output it.

So essentially it was a link. Anything that the car pinged, the receiver would pick up, transfer to the, I'm sorry, the car receiver would pick up, and then it would transmit over the high-frequency link to their receiver, which would then rebroadcast the low frequency at a much greater distance to the key, which was assumed to not be very far away. So with their system, you can imagine a number of attacks.

For example, somebody parks a very nice luxury car in a restaurant parking lot. And they go into the restaurant. The bad guys are seeing this. They know the make and model of the car, and that it's got the fancy PKES keying system. So one of them has a briefcase which has the receiver and rebroadcaster in it. That person simply follows the owner of the fancy car into the restaurant and arranges to stand near them. That briefcase receives the car's ping over the extended high-frequency link, which then pings the key, even though it is a long distance away. The key doesn't know that it's not right next to the car. So it uses its UHF transmitter to go that, as we know, about up to 150 feet to say, oh, unlock the door. The moment that happens, the bad guy gets in the car. Now his transmitter is sending the inside-the-car signal, which the key receives and responds with its challenge response.

Notice that the crypto doesn't have to be broken. They've simply extended the range because range was the only assumption that all of the security of this system was based upon. They extended the range. The key, operating at a much greater distance, responds to the inside-the-car signal. The bad guy presses the start button and drives away. All 10 cars, once started, never stop, for obvious security reasons. You don't want to, like, suddenly have the engine turn off.

**Leo:** Yeah, because you can have "key not present," and then you're going to be in trouble, right.

**Steve:** Exactly. Now, the alternative approach, which is what some in-the-field devices are doing, is simply receiving the low-frequency signal and using a much larger and higher power rebroadcast in order to reach the key low frequency at a distance, rather than using the somewhat more elegant but necessarily point-to-point system. So, oh, and you can imagine. Imagine if bad guys just put their rebroadcaster briefcase next to the valet key box in a fancy hotel or restaurant.

**Leo:** Yeah.

**Steve:** I mean, all...

**Leo:** They're all there, yeah.

**Steve:** All the keys are there. They're all responding to pings coming from all of the cars. And all the cars in the parking lot, you can simply - the bad guys walks over, sends that car to the rebroadcaster. That particular key responds. Open the door, drive away. And this system works. The reason, I mean, I wanted to take it out of theory and read those news stories because this is actually happening to people now all the time. So the problem is that there is something known as "RF distance bounding," which doesn't use signal strength. It uses time. It uses the speed-of-light just transmission delay round trip between the car and the key fob. We'll talk about that in a second.

Because the first thing anyone can do is put your key in a Faraday cage. And because I knew I was going to do this podcast, I went to Amazon. And I thought, you know, I just googled whatever, or put into Amazon some search terms, and I came up with this cute little sort of pseudo-leather Faraday cage. It's got a metallic sort of cloth feeling interior, lined on all sides. So you slip your key in there and close the lid over it, and it is now radio dead.

Now, unfortunately, I don't have such a key, so I couldn't test it. But it was $6. So anyone who's curious, you know...

**Leo:** I'll test one tonight because I have one from ScotteVest, and I will test it tonight.

**Steve:** Great, great.

**Leo:** I'll see if I can get in my car, yeah.

**Steve:** Yeah, and so the idea is just go stand next to your car and see if it still works or not. I mean, and that's the way anyone can test whether their little Faraday baguette successfully blocks their key's radiation is that, you know, put it in there, and put the key close to the car, and see whether they're able to communicate with each other. So that's the first notion. And of course, unfortunately, and I guess, I mean, I'm less sure that a refrigerator would work, as well. But maybe, if they're all, I mean, it has to be a really closed metal environment. You can use mesh, but mesh is, I mean, the mesh should be tight because what we care about is the wavelength through the mesh. If the wavelength is short relative to literally the size of the hole, that's what determines whether the energy is absorbed by the mesh or is allowed to pass through. What you want is it to be absorbed, that is, for the mesh to be so dense that it's essentially an electrical conductor.

In order for there to be an electromagnetic field inside the so-called Faraday cage, there has to be a differential, an electrostatic differential across the key. But if you're inside of something entirely conductive, then it's going to short-circuit, essentially, the opportunity for there being an electrostatic differential anywhere inside. There'll just be no electrostatic fields inside of a closed conductive container. So that's what you want.

Now, the second thing, I mean, if anyone's, like, really worried, most systems have some kind of dead battery provision. That is, if your battery dies, you will lose the at-a-distance lock and unlock. But most systems can fall back on RFID entry where, if the key is really in close proximity to, for example, the door handle, it'll still open. And then, if it's inside the car, you can start the car. That's in the dead battery case, which says, if you take the battery out of your automotive key fob, you'll lose the functionality at a distance, but no bad guys can hack you any longer. And then you're able to use the near-field RFID fallback.

**Leo:** Interesting. You know, my car says - recently my battery started going, and it said "change your battery." So you're saying, even if the battery had died, or I could take the battery out, I could still get in the car with RFID. There is a - and I've always wondered why. There is like a tap-to-pay logo on the dash. That must be what you would hold the key against then you press the starter to verify that you have a key. So maybe I'll just take the battery out of these. That would mitigate;

right?

**Steve:** Yes. These guys say that these systems typically have a dead battery fallback. And so obviously you lose its ability to transmit at a distance, but you keep its true ability to function in the near field, which is really what you want. And then you've got security.

Now, the good news is manufacturers must be scrambling around to fix this because there's lots of reports of this, and we just disclosed a true serious vulnerability which we know is being actively exploited in the field. I think people are not stealing cars because what are you going to do with a stolen car? You're much better off opening them up and ransacking them as people have been reporting. That seems like an easier way to generate value than being stuck with a stolen car, except at the high-end crime level, where people will know what to do.

**Leo:** Well, I just took the battery out. We'll see if I can get in my car and go home tonight.

**Steve:** And did I see a key inside?

**Leo:** Well, that's what's kind of weird. This has a key. I don't know what this is for.

**Steve:** Yup, because they've also talked about that. I'll bet you you have a key you've never seen on the steering wheel.

**Leo:** Yeah. I can probably start it with a regular key.

**Steve:** Yup, and that is also part of the fallback system. So it may be that that is what you have to do, if your battery is dead, is mechanically start it with that key.

**Leo:** I think that's - they're saying that's the valet key.

**Steve:** Ah, okay, that makes sense.

**Leo:** But that way I can keep this battery part in my Faraday bag and give the valet this and - oy, oy, oy.

**Steve:** Yeah, or just take the battery out and never worry about it again. Now, these things may be retrofit. It's not clear how we're going to get past this. But RF distance bounding uses the actual speed-of-light round trip. The problem is light is fast. And, for example, a one-meter round trip, that is, one meter out and one meter back, you know, remember that light goes at just shy of 300 million meters per second. It's 299792458 meters per second, which is 300 meters per microsecond, or 0.3 meters per nanosecond, which turns out, if we flip that over, to be 3.336 nanoseconds per meter. So round trip is two of those, or 6.67 nanoseconds.

Now, once upon a time that was really fast. But now, nanoseconds, we can do a lot of computation in 6.67 nanoseconds. The problem is that the key fob must have an absolutely stable response time relative to that round trip. That is, if the key fob had a variable response time of a millisecond, like then, yeah, that's a large response time compared to what we're trying to measure, so it won't work.

What we need - and unfortunately they measure these things, and key fobs tend today to be very sloppy because none of these systems use RF distance bounding, which is why all of them are exploitable. So what we're going to need is a next generation of this technology where, rather than relying on signal strength, which we know we cannot rely on, we rely on the actual measured speed-of-light round trip between what it is we're trying to authenticate against and the authenticator. So that, for example, if the key had a known response, a fixed response time of 30 nanoseconds, then at a meter distance the car would expect to have a response within 36.67 nanoseconds when it knows the key is within a meter. If it's longer than that, it'll just decide that the key's too far away. And no amount of hocus-pocus with radio links or amplifiers or anything can possibly improve on the speed of light. If the key is physically distant, unless we come up with hyperspace wormholes, which presumably would remove that distance bound, we're going to have a problem getting into, you know, bad guys are going to have a problem opening cars and ransacking them and/or stealing them.

So what I expect to see is soon cars are going to fix this problem. But unfortunately, all of those people who are stuck here in the middle with this radio technology that is unfortunately not very secure and can be spoofed just due to amplifying the low frequency signal so that the key hears it, and then it responds with its normal long-range signal, that allows cars to be opened and engines to be started. So maybe slip it into a Faraday cage, if that's convenient. Maybe take the battery out. Or maybe not worry about it at all. But we know that people are getting themselves hacked.

**Leo:** Well, and this exploit, I mean, this has always been here.

**Steve:** Yes.

**Leo:** And probably, if you'd thought about it for five seconds, you would have realized it. So I imagine this is, I mean, it's just, you're right, it doesn't happen more often because, well, grand theft auto is not something you really want to go down for.

**Steve:** Right. And reports are that, on eBay, I didn't go digging, but apparently...

**Leo:** Seventeen bucks.

**Steve:** Seventeen bucks you can buy an amplifier that will do the job.

**Leo:** Yeah.

**Steve:** And so I think the biggest problem, as you said, is...

**Leo:** Kids taking the change out of the car and stuff like that.

**Steve:** Or you leave your iPad in plain sight, or your phone or whatever. I mean, so, yeah, it's basically ransacking cars.

**Leo:** Well, I've taken the battery out here. So this will be interesting to see.

**Steve:** Yeah, see if you're...

**Leo:** See if I can drive home.

**Steve:** And let us know.

**Leo:** If I can't get home tonight, I'll let you know. I'll keep the battery in my pocket, just in case.

**Steve:** I know you will.

**Leo:** That's really - but it seems like that would mitigate it. As long as you could use RFID to get - you wouldn't have the distance button push for opening the trunk or beeping the panic signal.

**Steve:** Right, right.

**Leo:** I can live without that. I don't know how I'm going to open the trunk. I'll have to figure that out. All right. I've taken the battery out of both of these. I'd better tell Lisa. Wow.

**Steve:** Yeah.

**Leo:** Wow, fascinating. And of course this isn't the kind of thing a firmware fix is going to fix, I don't think. I think you'd probably have to buy a new car.

**Steve:** The problem is building 6.67 nanosecond timers, you know, into a system that didn't intend them to be there...

**Leo:** It's not easy.

**Steve:** Yeah, it's not easy. So maybe factory retrofit of some electronics, maybe. Or maybe they're just going to say, well, you know, that's not our problem. But we have a new model we'd love to sell you.

**Leo:** Yeah, yeah. Get the new car. Now safer.

**Steve:** And what's really interesting, too, is there is advanced crypto going on. There is symmetric key crypto. In one case it's AES.

**Leo:** Wow.

**Steve:** So that both the car and the key have a shared secret. They have a secret that no one knows. The car generates a nonce, a never-repeating number which probably increments. And so it sends this number to the key, which encrypts it, often using AES, to the proper response. And then the car verifies that that is the proper response, given the known, privately shared key. And then it says, okay, only the proper key could have ever, could have possibly answered this little crypto challenge. So open up.

**Leo:** I suppose they could update the fob; right? And make the fob not a full-time broadcasting tool. Just like has to have some sort of activation thing.

**Steve:** Well, it does. It waits until it gets pinged by the car. And when it gets pinged...

**Leo:** If they could maybe add a button push on the thing so that - you know what I'm saying?

**Steve:** Correct. Correct. See, that's why it's called "passive key entry."

**Leo:** Right.

**Steve:** They kept making it easier and easier to use.

**Leo:** Convenient.

**Steve:** Until now you don't have to fish through your purse, ladies.

**Leo:** Right, right.

**Steve:** You leave it in your purse.

**Leo:** Right. I love that.

**Steve:** Because that was a big - see, everybody does. And that's the Achilles heel is the fact that you have to do nothing at all. And so the key is sitting in a bowl inside the front door in most people's home. They walk in the front door. They put the keys down on a little table near the front door. And now it sits there, unfortunately too close to the car, close enough that the car can hear it if it gets stimulated.

**Leo:** So that's one way your car company could fix it, is issue new fobs that require some sort of active thing. And make it an option. Say, you know, there's a security issue. We know it's convenient.

**Steve:** Do you care?

**Leo:** If you care, we'll give you a new one where you have to push a button to activate the fob for it to work.

**Steve:** Yup.

**Leo:** And that would solve it.

**Steve:** In fact, they could make a firmware change. I don't know if it would be at which end. But you're right, Leo, they could just set it up so that you have to press the Open button, which you already have...

**Leo:** Right.

**Steve:** ...in order, you know, it just doesn't do it automatically. You'd have to do it explicitly.

**Leo:** Yeah, right. I like that. Or put a fingerprint reader on the fob, somebody suggested. That's a little - that's going too far.

**Steve:** Yeah.

**Leo:** All right. Good subject. Good show. Thank you so much. Steve Gibson is at GRC.com. That's where he hangs his hat and his SQRL and his SpinRite. In fact, if you want SpinRite, you should get it right there. Even if you don't want it, you should get it, because everybody's got hard drives. You need it. I'm telling you.

**Steve:** Sooner or later.

**Leo:** Sooner or later, everybody needs it. You can also get 16Kb versions of the audio of this show, fully transcribed by human person transcriptions. Actually all of our transcriptions are written by human persons. But yours is written by the wonderful Elaine Farris. What else? Well, there's all sorts of good stuff. Go visit it. Next week we'll answer questions. So if you have a question, that's also a good place to leave it: GRC.com/feedback. You can also tweet Steve, @SGgrc. And he'll maybe...

**Steve:** I'll see it.

**Leo:** ...see it and maybe record it, and then we'll ask it next week.

**Steve:** Yeah.

**Leo:** We have full 64Kb audio of the show on our website, TWiT.tv/sn.

**Steve:** Maybe not yet going back in time.

**Leo:** Not going back past 400. But soon, soon.

**Steve:** Yes. And mine does go back in time now.

**Leo:** Good.

**Steve:** So for all those people who were unable to get it, I fixed that.

**Leo:** Get the old ones there.

**Steve:** And your new website will have it, too.

**Leo:** Oh, yeah. In fact, we could probably fix it right now.

**Steve:** Yeah, I imagine.

**Leo:** I should get somebody to do that. What else do we have? We have video, if you want to watch Steve gesture and gesticulate.

**Steve:** And hold his little wacky Faraday cage up. I don't think I know anybody with one of these...

**Leo:** I like that you called it a baguette.

**Steve:** Yeah, well...

**Leo:** That's a good name for it.

**Steve:** Little baguette.

**Leo:** Little baguette. We do the show every Tuesday about 1:30 p.m. Pacific, 4:30 p.m. Eastern time, 2030 UTC. If you want to join us, we love it when you watch live. The chatroom is always a great asset to me, although Steve does not participate during the show because he's busy. Although you're often in the chatroom other times. I see you there all the time, so that's nice.

**Steve:** Yeah.

**Leo:** And that's about it. Thank you for joining us. We'll see you next time on Security Now!. Bye-bye.

**Steve:** Thanks, Leo.