Securing the Vending Machine

Description: This week we follow up on the Win10 ZIP extraction trouble, discuss some welcome Android patching news, look at SandboxEscaper's latest zero-day surprise, examine the Hadoop DemonBot, follow up on U.S. DoD insecurity, look into the consequences of publicly exposed Docker server APIs, look at a DDoS for Hire front end, check out the mid-week non-security Windows 10 bug fix update, look at the just-released Firefox v63, and examine a new privilege escalation vulnerability affecting Linux and OpenBSD. We also handle a bit of errata, some sci-fi miscellany, and a bit of closing-the-loop feedback from a listener. Then we answer last week's puzzler by exploring various ways of securing those vending machines.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-687.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-687-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. More zero-day exploits in Windows 10. A problem with Docker. Google's plan to fix Android. And Steve has the answer to his conundrum from last week, our vending machine problem. All coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 687, recorded Tuesday, October 30th, 2018: Securing the Vending Machine.

It's time for Security Now!, time to protect yourself and your loved ones online, as we do each week at this time with this guy right here, Steve Gibson of the Gibson Research Corporation, GRC.com. Hi, Steve.

Steve Gibson: Yo, Leo. Great to be with you again.

Leo: Yeah. You were just up the road apiece.

Steve: I was in wine country this morning.

Leo: Nice.

Steve: I awoke among the grapes.

Leo: Did you have a nice visit?

Steve: I did. I've mentioned to you before the geek group that we were - it was the MRC, the Math Resource Center gang at my high school. So that was 45 years ago because we were graduating class of '73. And one of our group is sort of a wandering nomad, and he had taken a picture maybe six months ago of a group that excluded me that was our group. And I said, okay, I am never going to miss another opportunity, because it's been 45 years since we've all hung out.

And so as it happens, there was an opportunity that came up yesterday. And so I prepared the podcast, I spent all day Sunday working on the podcast, getting it ready for today, so that I could take yesterday off, which is normally my podcast prep day, and spend it up in Napa. So I got there by early afternoon, and we just hung out and caught up on 45 years of all of our shared history. And it was great. So I spent the night, woke up, and flew back down so I could be here in front of my blinky lights. And it's good...

Leo: It's always nice to sleep in your own bed, if you can, I have to say.

Steve: I can't wait. I can't wait. So, yeah. Although I do travel with my pillow because that's - Lorrie and I joke that I have three feathers in my pillow. It's the limpest pillow you have ever, ever seen. And everybody...

Leo: Hard to get a limp pillow at a hotel these days.

Steve: Yeah, everybody's got these foam things, and your neck is all cricked.

Leo: I don't like a foam pillow.

Steve: And you've kind of got to get on top of it somehow. It just doesn't work.

Leo: Oh, I'm with you. I like feathers. Give me feathers.

Steve: So basically I travel with a pillow. This is Episode 687, and I titled it the answer to our last week's teaser because, oh, my goodness, has our listener audience had fun with this. I mean, I was getting feedback from every available channel in every orifice from every venue. I mean, it was just...

Leo: But did anybody get it right?

Steve: A couple people did. There was of course the inevitable crazy over-engineering where, yeah, we've got certificates flying in every direction. You've got to put a catcher's mitt up when one's flying by and then grab it and then go home and come back. Anyway, so we will wind up with kind of a careful walkthrough of what would be the minimal solution if it weren't for an intriguing problem that the minimal solution has, and then just sort of inch forward a little bit to how to solve that, which then ends up with a really

elegant, nice solution. So this podcast is titled "Securing the Vending Machine." But of course...

Leo: And you'll restate the problem before we give the answer.

Steve: Yeah, of course. So of course lots of other stuff happened. We do have a follow-up from Microsoft on last week's mentioned Win10 zip extraction trouble. We've got some welcome news from Google on the Android patching front. Believe it or not, our apparently depressed hacker who goes by the handle SandboxEscaper...

Leo: She's at it again, yeah.

Steve: She's depressed again and has dropped another zero-day surprise on us. We've got a Hadoop-based bot called DemonBot. I also want to follow up a little bit on that U.S. GAO report on the DoD lack of security, how distressingly insecure those systems are. There's been some action - it's a bureaucracy, so good luck. But still I guess nothing happens without some hope. We're going to take a look at the consequences of publicly exposed Docker server APIs, which you just have to scratch your head. It's like, wait, really? You're saying that the Docker service API, there are public exposures? Oh, yes. And so of course hackers have jumped on those.

Also I've got some fun DDoS for Hire news. We're going to look at the going rate, in terms of dollars per minute, of purchasing DDoS for Hire. I'm sure none of our listeners would ever blast somebody off the Internet, but now we know exactly what it costs. And there are six, no, sorry, nine different plans you can choose from, like the copper, the gold, the VIP executive plan. Anyway, we've got all that. We're also going to check out a midweek Windows non-security bug fix, which they normally drop a couple days after the second Tuesday of the month where those are all the security fixes. Then they've got, well, we've also, you know, nonsecurity stuff, and lots of stuff there. Well, large in terms of number, nothing really in terms of breathtaking.

Also Firefox just jumped up to v63 with some welcome changes that we have forecast previously. There's an I don't think much to see here new privilege escalation vulnerability which affects Linux and OpenBSD. We just have to talk about it so that it's covered, but it's not anything that should worry people too much. Well, based on the profile of your system. We've got a little bit of errata. I want to talk a little bit about Peter Hamilton and the Salvation Sequence, just because what he's done - and there's no spoilers in this. You know I don't do those. But what he's developed is really fun, in a Peter Hamilton-ish way. I got a little bit of closing-the-loop feedback from a listener.

And then we're going to wrap by taking a careful look at this interesting security architecture of a smartphone wanting to be used to purchase things from a vending machine. Of course last week we talked about how in this particular instance the app was easily reverse engineered, and some white hack gave himself 999 euros, which was probably the maximum number that the app would accept, and it worked. And of course he notified the vendor, who has taken some action. And so I left it as a puzzler, as our listeners know, for what would be the solution to that? So I think another great podcast for our listeners.

Leo: Nice. Should be fun, securing that vending machine. Of course, listening to this show, if you get your employees to listen to the show every week, you'd probably be okay. But most employees are not listening to Security Now!.

Steve: No. So I fact, when I was catching up with my high school gang in this little informal reunion we had yesterday, I mentioned that I've been doing this podcast. We were in the 13th year. And one of them said, "So how long is it?" And I said, oh - and in fact I think that one of them guessed, like, oh, about like 15 minutes? I said, "Oh, no, it's two hours." And they said, "Two hours? Who's going to listen to that?" And, like, "Nobody does a two-hour podcast."

Leo: Oh, yes.

Steve: I said, "Well, in the beginning it was 20 minutes, and it's sort of gotten out of control."

Leo: Yeah.

Steve: So anyway, we have a fun Picture of the Week. It's been in my library of things to use for the podcast when nothing is particularly apropos for whatever topics we're talking about. Anyway, this shows a robot, kind of a classic "robot" robot, android-y kind of robot, sitting in front of a keyboard and screen, looking back over its shoulder at a guy who's just apparently walked in the door holding a cup of coffee. And the robot is saying to him, "Hey, man. Would you tick this 'I'm not a robot' box for me?" So anyway, yes. Little reversal.

Leo: Get it?

Steve: A little reversal.

Leo: It is kind of timely. Google has just updated their - you know they had that reCAPTCHA thing. You see that everywhere now.

Steve: Yup.

Leo: And they've just put out version 3 that is a little, they say, more effective. I see it a lot now.

Steve: Well, it's nice, too, that where they're able, as we've talked about this before, sometimes all you have to do is just click the "I'm not a robot" button.

Leo: Right. Because they know so much about you.

Steve: Yes, exactly. And they go, okay, yeah, well, we thought that, but we just wanted to make sure. It's like, okay, okay.

Leo: And then there's the ones that I really hate. Used to be, you know, numbers, letters, and all the things. The new one, you've seen this, is the click...

Steve: Oh, the grid.

Leo: Yeah. You have nine images. Click the ones that have a traffic light in it or a car in it.

Steve: Exactly. Or show a pedestrian or something.

Leo: And it's so hard.

Steve: I get them wrong often.

Leo: All the time.

Steve: I'm not a robot.

Leo: Because they get the top of the head. Is that part? Is that picture good? Is there a pedestrian in that picture? It's just a hair.

Steve: Yeah. For a while we were doing those street numbers, like random photos of people's numbers on the side of the house.

Leo: That was to help Street View because they were actually - those were - they couldn't always read those numbers in their Street View cars.

Steve: That's right.

Leo: Yeah. They're no fools.

Steve: Just abuse your humans.

Leo: Yeah, that's really what it is. We know you're not a robot, but could you help out here?

Steve: So at 5:27 p.m. last Tuesday afternoon, the day that we were finishing up, I mean, last week we did the podcast in the morning.

Leo: Right.

Steve: In the afternoon Microsoft acknowledged that zip file problem in Windows 10.

Leo: Oh, boy.

Steve: In a posting with a long title. Believe it or not, this was in answers.microsoft.com. The title of the posting was "If you copy files from a .zip file without extracting them, they might not be copied or moved correctly, even though it looks like they have been."

Leo: That's a very long starting headline there.

Steve: End of quote, yes. So they explain. And it's useful for the exact what it is that they've done here. They wrote: "There is a known issue in the Windows 10 October Update where the consent prompt 'Do you want to replace these files'" - which is really handy, okay, that's me saying that, they didn't say that - "is missing when copying contents from a zip file. With the Windows 10 October 2018 Update, if you copy or move files from a .zip file without first extracting the contents into a new destination folder" - and I guess it really wouldn't be new - "that contains duplicate filenames or is write-protected, you don't get a 'Do you want to replace these files' prompt. It will appear that the files were overwritten, when in fact the copy action for those files is not executed, and files have not been overwritten."

So, okay, that's a little news over what we understood to be the problem last week. There were reports on Reddit of overwrites. It looks like it's just a - it's like it thought it gave us the confirmation, and we said, no, don't replace them. But it just skipped that step. So it looked like they were overwritten, but nothing happened. So then they explained: "This failure can occur in the following three scenarios: copying from a compressed file to a regular folder, moving from a compressed folder to a regular folder, or copying from a compressed folder to a protected folder."

And then they said: "Note: While the copy action for the duplication file names does not complete and no files are overwritten, the 'move' command will also silently fail and might remove or delete the moved file." That is, you're moving something, and as we know, moving means essentially copy and then delete from the source. So in that instance there is a potential file loss problem where, again, it doesn't actually perform the copy, but it thinks it did, and so it removes the source of the file.

So anyway, they said: "We recommend you fully extract the zip folder before you copy files to a new destination folder to avoid this issue." They said: "And if you deleted items at any point in this process, you can recover files that were not copied to the destination folder or were unintentionally recycled by doing the following," and then basically they explain about the recycle bin. So apparently that's where things go. Of course, I've never been a recycle bin advocate. I always - one of the first things I do when I'm setting up a new system is I turn that all off because I just - I've got backups of my backups in any event. So if I really lost something, I could get it back. I just like it to go away.

But in any event, so that's the story. And they said, and this is kind of curious, they said: "Microsoft is working on a resolution and estimates a solution will be available in early November for this issue." Okay, well, tomorrow's Halloween in the U.S., so the end of October. So November starts on Thursday. But as a consequence of that timing, we don't have our second Tuesday, the infamous - it is infamous now - Patch Tuesday until mid-November, essentially, November 13th.

And this is a potential data loss situation. So it's unclear whether they might push an out-of-cycle fix for this, or if they're going to just say everybody hold your breath for two weeks, because two weeks from today is when this will be fixed, unless they decide, okay, this is something we have to deal with sooner. So don't know yet. And they're not saying.

I mentioned some good news on the Android updates front from Google. And this is a consequence of The Verge that recently obtained some copies of confidential Google vendor contracts. So this is not a formal announcement. All we really had was back in May a statement of Google's intent. Back then what The Verge coverage was, they said: "Google says it will require Android phone manufacturers to roll out security patches on a 'regular' basis, though it isn't clear who that requirement will apply to or how rigorous the mandate will be."

Then The Verge says, and this is back then in May: "On Wednesday, during a talk at Google's annual developer conference that was caught by 9to5Google via XDA Developers, the company announced that many more users would receive regular security patches, thanks to new agreements it's making with partners. David Kleidermacher, Google's head of Android security, reportedly said: 'When you have billions of users, it's a large target. [Uh-huh.] And so it deserves the strongest possible defense. We've also worked on building security patching into our OEM agreements. Now this will really lead to a massive increase in the number of devices and users receiving regular security patches.'"

And then The Verge finished the coverage back then, saying: "Unfortunately, there are no details beyond that. We reached out to Google to learn how frequent the security updates will be and who they'll apply to," they wrote, "but the company didn't immediately have answers for us. It sounds," they wrote, "like the requirement will apply only to new phones launching on Oreo or later that take advantage of Google Play services, so likely nothing in China. Even then, it isn't clear whether it'll apply to all of Google's partners."

So that was then. Now, today, with the benefit of these leaked confidential documents, The Verge has said: "Every month, a security team at Google" - as we know, they're sort of giving us some background - "releases a new set of patches for Android. And every month, carriers and manufacturers," they wrote, "struggle to get them installed on actual phones. It's a complex, longstanding problem. But confidential contracts obtained by The Verge show many manufacturers now have explicit obligations about keeping their phones updated written into their contract with Google."

Leo: Hallelujah.

Steve: Yeah. "A contract obtained by The Verge requires Android device makers to regularly install updates for any popular phone or tablet for at least two years."

Leo: I wish it were longer, though. That's...

Steve: I know. I thought the same thing. It's like, okay. Because here we were just recently talking about old routers and how they've been abandoned, but they're still in use, and that's a problem. So The Verge wrote: "Google's contract with Android partners stipulates that they must provide 'at least four security updates' within one year of the phone's launch. Security updates are mandated within the second year, as well, though without a specified minimum number of releases. David Kleidermacher, Google's head of

Android security, referred to these terms earlier this year" - and this was what I told you that they had said back in May - "said that Google had added a provision into its agreements with partners to roll out 'regular' security updates. But it wasn't clear which devices those would apply to, how often those updates would come, or for how long."

The terms cover any device launched after January 31st, 2018, okay, so that's all of this year, after the first month of this year, that's been activated by more than 100,000 users. Okay, so failed devices, I guess, aren't important enough, or small user bases, or small manufacturers maybe. Then starting July 31st, so the middle of this year, the patching requirements were applied to 75% of a manufacturer's "security-mandatory models." Then starting on January 31st, 2019, so again, one month into next year, Google will require that all security mandatory devices receive these updates.

The Verge wrote: "Manufacturers have to patch flaws identified by Google within a specific timeframe. By the end of each month, covered devices must be protected against all vulnerabilities identified more than 90 days ago. That means that, even without an annual update minimum, this rolling window mandates that devices are regularly patched," or at least within 90 days, within three months. "Additionally, devices must launch with this same level of bug fix coverage. If manufacturers fail to keep their devices updated, Google says it could withhold approval of future phones, which could prevent them from being released."

So of course our listeners know we've spoken of this often. And in fact, I mean, we've despaired at the severity of some serious problems that have surfaced on Android which Google's own devices are patched for, responsibly and almost immediately, I mean, as quickly as one can. But the fact that all the other phones with the same level, the same version of Android, may never be patched, thus leaving huge numbers of users exposed to what then become publicly known vulnerabilities. And, boy, if this podcast has shown us anything, it's that bad guys are now, I mean, this is like things that are patched are a source for some subset of hackers to jump on, reverse engineer what got changed, what was patched, and then turn that into an exploit and go after all the devices that have not yet received that same patch.

So that's great that we're seeing Google recognizing that their own Android ecosystem is suffering from the fact that they inherently and by design don't have the same kind of grip that Apple does over the iOS ecosystem; but that, frankly, for this kind of product, with the reality of today's vulnerabilities, a tighter grip is necessary in order to protect the user. So anyway, props to The Verge for getting a hold of that contract. And now we have some sense for how this actually looks in the real world.

And SandboxEscaper, Leo, she's just still not having a good time.

Leo: Aw.

Steve: Her tweet read: "Here's a low-quality bug that is a pain to exploit, still unpatched. I'm done with all this anyway. Probably going to get into problems because of being broke now, but whatever." And, you know, the problem is I took a good look at this because there's a proof of concept with all the source code up on GitHub. It's DeleteBug1.rar under her GitHub account. And it's a nice piece of work. I mean, like I wouldn't hire her, but somebody ought to. This person should be employed. I mean, this takes some skills to find these sorts of things. So, I mean, you'd have to make sure, I don't know, maybe have her talk to somebody about her apparent depression. But boy, I mean, there is some skill there.

So just to recap, back in late August our listeners will remember when she was apparently having another bad day. She posted to Twitter to expose and to detail a proof-of-concept exploit for a local privilege vulnerability in Microsoft Windows Task Scheduler, which was due to its handling of an advanced local procedure call, the ALPC service. And as we know and covered, almost immediately after the proof of concept was released, even though it wasn't a remote execution-style vulnerability, I mean, not the end of the world, as we said at the time, even malware, I mean, this would even be useful for any malware that could get into your system in order to elevate its own privileges to do more damage.

And that is exactly what happened. It wasn't long after this zero-day dropped that we found malware using exactly this in order to give itself privileges, system-level privileges, and deal with the fact that a user properly using a non-privileged account could still get malware installed into the system. Which non-privileged users, it's the reason they don't have privilege is it's kind of pain not to, but it's supposed to prevent this. So we know that it didn't take long for what many in the industry felt was an irresponsible disclosure of a zero-day.

So it's happened again. What she's describing as a low-quality bug is something she found in a new DLL, something known as the Microsoft Data Sharing Service. It's dssvc.dll. It's a new service introduced in Windows 10. So unlike the previous proof of concept, which went way back into Windows history, so it affected all versions of Windows, this one that she's just dropped now is Windows 10. And both of the descendant servers, Server 2016 and 2019, are vulnerable to a misuse of what she has found and proven. So it does not affect 7 and 8.1 or their server versions.

The service, this Microsoft Data Sharing Service, runs with LocalSystem, which is to say root or kernel, full kernel privileges, because it serves as a data brokering agent between applications. That's what this does. And so if you are able to abuse this, and she has figured out how, the abuser is essentially a process, this DLL running in the kernel. So the proof-of-concept exploit code is deletebug.exe, which people can download. Don't run it, though. What it demonstrates...

Leo: Okay.

Steve: Yeah, please don't run it. She uses this to delete a critical system service, a pci.sys file, which you have to have to boot. So it immediately renders a system unbootable. So, yes, again, don't try this at home. But it convincingly demonstrates the fact that this is doing something it should not be able to do. And unfortunately, all the source code is there. Nothing is hidden. And she just said, here you go, I'm having another bad day, so I just found this, and I'm letting the world know.

So it generated, as her actions have in the past, some feedback. And I was sort of curious, so I read through the Twitter stream a little bit. A Mitja Kolsek retweeted a response from Microsoft Security. And he tweeted: "Hey, girl. In case you missed this," and then reiterated the tweet. And he said: "Please continue with your work in this area and get paid by Microsoft. What you're finding are not," he said, "easy-to-find bugs. And your proof of concepts and write-ups are of high quality."

Leo: That's awesome.

Steve: And I completely agree. I mean, it's top drawer. It's really good work. And then he said: "There. You should know this. Cheers." And Microsoft Security response was:

"Yes, this vulnerability is in scope for the Windows Insider Preview bounty program." So...

Leo: Get your money here.

Steve: Get some money. And then I did get a kick out of, unfortunately, Peter Stevesant also - and he's tweeting from @binaryfraud2017 is his handle. On October 26th he responded to both of those previous tweets. And he said: "Well, when you see how much Microsoft pay for exploit, you should sell them to" - and then he give the URL, and we've talked about them before, http://zerodium.com, which of course is the zero-day reseller. And he says: "They will consider your work much better." And I guess he's meaning you will be better paid for that. But in any event, dropping zero days like this, I mean, they're beautifully engineered. And unfortunately what that means is they are immediately snatched up by any malware that wants to be able to bypass the whole Windows privilege system. That's what this is doing.

So, yes, it doesn't immediately allow bad stuff to get in. But, boy, I mean, if you click on something in email, like a phishing attack, like for example that you were just talking about from this previous sponsor, Leo, the bad stuff, you've given it a chance to run. Well, the whole idea of having privileges on accounts is only so that most people don't have them. And this just cuts through that. And, yes, Microsoft will fix this immediately. I'm sure we'll see this fixed two weeks from today on November 13th. But there's two weeks between now and then, and then there's also the patch delay. Because, boy, after October's disaster, I don't think anybody's going to be in a big hurry to fix this.

Oh, and I forgot to mention, those little micropatch folks at 0patch.com, they do have a fix for this already. So if you're in an environment where you think you would be exposed to this kind of problem, you don't have to wait for Microsoft. And I've talked about these guys before. It's numeric 0patch.com. And they're able to respond within hours of these sorts of disclosures to fix these - they call them "micropatches" to fix these little vulnerabilities until Microsoft performs the official update. So one is available for this, too.

Okay. So Apache Hadoop. I just love saying it, "Hadoop." It's just...

Leo: So does Mary Jo Foley. She loves saying it, too.

Steve: So it's a collection of open source software utilities which facilitate using a network of many computers to solve problems involving massive amounts of data, huge datasets, and computation. And you get to say "Hadoop" in the process. It provides a software framework for distributed software and processing of big data using the MapReduce programming model, whatever that is. I've never had an occasion to dig into that.

Leo: That's a Google big data engine.

Steve: Ah, okay. And I'm just quoting from Wikipedia here. "Originally designed for computer clusters built from commodity hardware, still the common use" - so that means you get to take like bunches of off-the-shelf stuff and just put it all together, and Hadoop somehow corrals it all and manages it with MapReduce, whatever that is. And Wikipedia says: "It is also found used on clusters" - and here is the problem - "of high-end

hardware. All the modules in Hadoop are designed with a fundamental assumption that hardware failures are common occurrences and should be automatically handled by the framework." Which is like a dream; right? You just throw a whole bunch of stuff that's kind of limping, and Hadoop fixes it, works with what it has somehow. And if something dies, it goes, oh, well, and works around it.

Okay. So as a consequence of this Hadoop cluster nature, it turns out that Hadoop clusters - and I'm going to say "Hadoop" as many times as I can during this - Hadoop clusters are often very powerful, well connected, and placed into the Internet cloud, where unfortunately they are accessible either deliberately or by mistake. So it turns out that Hadoop has something called "YARN," which is an abbreviation for Yet Another Resource Negotiator. And it sounds like something Hadoop would need, right, because it's about resource negotiation. And apparently there were some before. This is yet another one.

So YARN. It provides cluster resource management for enterprise Hadoop deployments, has had a known flaw in the handling of its REST API, which is an HTTP-based API, for which a Metasploit proof of concept was published after a 21-line, which is to say not very long or complicated or hard to understand, Python source proof of concept was posted onto GitHub - get this - seven months ago, last March. I have a link to this exploit.py file, and it's just like, oh, you mean that works? And it turns out, yes, unfortunately, seven months later it still works. There's a Metasploit module which in its description it says: "This module exploits an unauthenticated command execution vulnerability in Apache Hadoop through ResourceManager REST API." And so I have a link there, too.

So now what we know is that, unfortunately, there exist on the Internet publicly exposed Hadoop clusters which have not been patched in seven months, since this was known and made public. So Radware, the security guys, have reported that an entity that actually identifies itself as DemonBot, does not self-propagate in worm style, which is to say that infections do not themselves look for other vulnerable systems. But they have identified a growing number of servers - they cited the number 70 servers - which use, and I love their term for this, they called it the "spray and pray" tactic to blindly find and infect millions of exposed publicly available and of course unpatched Hadoop instances.

They're using them to run cryptomining ware. Oh, and also being used to source extremely powerful, thus DemonBot, extremely powerful and debilitating DDoS attacks which unfortunately these Hadoop instances are really well equipped to do because they are typically on high-end hardware and on very well connected to the Internet pipes, so large bandwidth, often on strong servers and able to launch high bandwidth attacks. So very much as with the millions of compromised routers and other insecure publicly exposed and vulnerable services, attacks such as these will be part of the Internet terrain until we figure out how to build secure systems out of the box, if we ever do.

And it's funny, too, because again, as I was introducing my old friends from high school to the idea that I was in the 13th year of a podcast, they said, one of them said, "You haven't run out of stuff to talk about after 13 years?" And I said, oh, no. And in fact I think that may have been the dialogue that got us into, "And how long do you talk every week about this stuff?" And I said two hours. And they said, "What?"

Leo: Somebody's doing the math right now, yeah.

Steve: So two weeks after that rather horrifying Government Accountability Office, the GAO report detailing glaring cybersecurity issues in weapons systems at the U.S. Department of Defense, the DoD has announced that it is expanding its existing Hack the

Pentagon bug bounty program to include hardware assets. And Leo, I really don't understand how that's going to work because how do you hack an F-35?

Leo: Well, god, I hope you can't. That's all I'm saying. If you can, they'd like to know. I think that's fair; right?

Steve: Wheel one of those over here, and let me plug my USB stick into it. But somehow they're saying they're going to do this. So they're going to be tapping the Synack, the HackerOne, and the Bugcrowd platforms, which are the existing bug reporting bug bounty organizing platforms, to attract more white hats to this effort. And to me this feels like, okay, I'm not really convinced that this is going to work. But they said, since the Hack the Pentagon program was kicked off in 2016, the bug hunters have found - and I wasn't aware of this, so that's just two years ago - 5,000 - which is a little distressing, too - 5,000 code vulnerabilities.

And six public-facing bounty challenges have been run, including the most recent, which was Hack the Marine Corps in August. Other sessions have focused on the Air Force, Army, and the Defense Travel Service. A three-year, 34 million - and this is what this new contract is, and this is bureaucratic jargon, they called it the "indefinite delivery, indefinite quantity" - contract package covering these three bug bounty managing companies will crowd source vetted hackers to probe the DoD's websites, hardware, and physical systems. And again, it's like, okay, but, you know...

Leo: I guess they're going to let them in.

Steve: An atom bomb? That doesn't sound - so according to the contract's performance work statement, the government expects its military contractors to run at least eight limited-time challenges and five continuous challenges during the first year of the three-year contract, and more if an option in the contract is exercised. Each program will last between three months to a year, and they might overlap. And I was thinking about this, Leo. It seems to me that virtually all of the interesting and critical systems are classified; right?

Leo: Yeah, yeah.

Steve: And again, what I was wondering was - because we know that there's inter - I don't know if you'd call it interagency or interdepartmental or division. We know that there's competition between the Army and the Navy and Marines and so forth. So I wonder what would happen if, instead of involving external white hats, if you instead opened the systems up to interdivisional hackers. So the Marine hackers could try to get into the Army's systems and vice versa, if that might be something that would be feasible because, again, I just - it's hard for me to believe that I can sign up to Bugcrowd and be vetted and be given access to hardware which is, you know, "Here, Steve, hack this system. See if you can get in." It's like, really?

So, okay. Anyway, it'll be interesting to see how this pans out. To me it seems like it would make sense if you kept it within - because, well, the reason I was thinking about this is that we know that for hacking you really have to have an adversarial posture. You can't have programmers check their own code. You've got to have programmers who are unfamiliar with the system and who are truly motivated to get into somebody else's system trying to do that in order to find this class of problem. And so it just seemed to

me, if you had a situation where you could set up a true competition, but also sort of keeping it in the family, that is, or within people who have security clearances, then maybe you could get the same effect, but also be able to pull it off. Because I'll be surprised if people without high-end security clearance really ever get to touch high-end hardware. So be interesting to see.

And I was a little surprised, Leo, when I heard that Docker Engine APIs were exposed. And the good news is not a lot of them. So as we know, Docker creates sort of a universal packaging approach that bundles all of an application's dependency into a container so that you don't - so to solve the whole dependency hell problem you create a Docker container that is just sort of a drop-in solution. You just put it into an environment that runs containers, and it will run because it brings everything it needs with it. So these Docker engines are available for Linux on the CentOS, Debian, Fedora, Oracle Linux, Red Hat, SUSE, and Ubuntu. And also Windows Server has a dockerd service, a Docker daemon that's able to also do runtime Dockers.

So it turns out that, believe it or not, and our regular listeners will believe it, this Docker Engine is a server which interacts with its local environment via an API. That API exists exposed in some cases on the public Internet. Or as Trend Micro put it in their security advisory: "Misconfigured Container Abused to Deliver Cryptocurrency-mining Malware."

They wrote: "We recently observed cases of abuse of systems running misconfigured Docker Engine-Community" - that is, the community edition, there is a community and an enterprise sort of classes of these - "with Docker Application Program Interface" - that is, API - "ports exposed. We also noticed that the malicious activities," they wrote, "were focused on scanning for open ports 2375 over TCP and 2376 over TCP, which are used by the Docker Engine daemon, dockerd. The intrusion attempts to deploy a cryptocurrency-mining malware," which their software, Trend Micro, found as Coinminer on the misconfigured systems.

So this Docker API allows remote users to control Docker images like a local Docker client does. "Opening the API port for external access," they wrote, "is not recommended" - yeah, no kidding - "as it can allow attackers to abuse this misconfiguration for malicious activities."

Now, unlike routers, where by policy these services are open and vulnerable, which we've been covering really pretty much this year, the Docker Engine itself isn't compromised or abused, and the Docker Enterprise platform is not affected. The Docker Enterprise platform has very strong authentication that cannot be bypassed. But Trend Micro wrote that they found "rare instances of abuse" on the Docker Community version. I mean, and so people have to really work, it turns out, to make this exposure public. It's not clear whether they have to explicitly open ports to the public, or if they install a Docker Community instance on a public server, if just essentially sheer negligence ends up exposing these systems.

The good news is there are not hundreds of thousands of these instances. There are hundreds of them. As in, like, a low number of hundreds. Trend Micro reported a peak of, looks like from their graph that I have in the show notes, maybe about 425. It looks like it peaked around early this month, early October, and then it sort of jumped up and down based on their scans. So anyway, I was just sort of surprised that there were any instances of this. It does look like, though, since Docker is very popular, and it's certainly in use in hundreds of thousands of systems, it looks like only in a few hundred because this malware is scanning aggressively. They're seeing a scan rate of 50,000 packets per second, looking for open ports 2375 and 2376.

So it is attempting to move laterally. If it gets into a network, it scans all networks it's accessible to. So the problem, of course, is that it'll have, if Docker is there in an

enterprise where there is a publicly exposed instance, once that gets infected, it is acting as a worm which is turning around and scanning into the Intranet, looking for additional instances where it may be able to jump. So they've only seen - it peaked around 425 instances. But who knows what those instances may have found when they pivoted and then looked inward into organizations that may be using Docker pervasively.

So again, the good news is, unlike routers, the Docker designers clearly understand the importance of not letting this API be exposed, and by default it appears that it's not. But even then you just can't keep people from misconfiguring systems. And Leo, you are going to love this next piece.

Leo: Okay.

Steve: What is the going rate for DDoS for Hire? As a result of Fortinet's threat research, we now know in at least one instance they have screenshots of the front end of a DDoS for Hire service powered by what's known as the Bushido Botnet. And I have to get a little close to the screen here to read this. In fact, looks like in the compressed version I can't. But I know that there was the Bronze, the Silver, and the Gold. For $20 - oh, yeah. I have it down below.

The $20 for the Bronze plan, you were able to purchase that, and you got 900 seconds of DDoS, which is 15 minutes. And the Bronze plan gets you one concurrent attack using what they call their "standard" network. Then this goes all the way up. You have the Bronze, the Silver, and the Gold. But then you can also go from the Standard to the VIP. So you also have Bronze, Silver, and Gold in the VIP package. So the highest end plan is the $150 VIP Gold plan. So again, you have three tiers and three metals, Bronze, Silver, and Gold. So a total of nine different plans ranging from $20 to $150.

The $150 VIP Gold plan buys you 7200 seconds, which is two hours, offering two concurrent attacks using their VIP network, whatever that is. And then through the browser UI you're able to select which types of plan you want. And two pages down in the show notes, just because I thought this was really fun, is the current gigabits per second offered by their DDoS attack tools. And when this screenshot was taken, it was 424 Gbps.

Leo: It says "gigabytes." Capital B. Isn't that a byte?

Steve: Yeah, I think it's probably gigabits.

Leo: That's an error, yeah. That may be right.

Steve: Yeah, I think it's bits because that's normally what we think of in terms of network speed is in bits. And but still, that's almost half a terabit per second of flooding bandwidth. So, I mean, that's goodbye target. I mean, that's meltdown. And at the time, again, that this was taken, this site is the 0x-booter site. And of course these are all couched, not as, oh, we're not selling you attack. This is to, like, stress test your network. Okay.

Leo: Oh, yeah, sure.

Steve: 424Gb is not a stress test. It's the center of the sun. So at the time that this page, this web interface, the snapshot was taken, they had 16,993 bots present in their network waiting for marching orders. Where would you like us to aim the attack?

Leo: Wow.

Steve: So Fortinet writes: "Distributed Denial of Service offerings, often disguised," they said, "as legitimate booter or stressor services," they said, "continue to increase in the cyber underground market. This relatively new 'crime as a service' trend has created an entry point for novice DDoS attackers offering a simple option to anonymously attack nearly any website, forcing it offline for a small fee." They wrote: "Due to the public release of the source code of some popular bots" - and of course Mirai we've spoken of is among them - "building a botnet to provide these services is now simpler than ever. A quick Google search returns lists of resources for botnet builders, usually with complete step-by-step instructions." Because why not?

"Being able to reuse and even modify the source code has enabled cybercriminals to create their own versions that implement new functionalities." Oh, and "Fortinet will be delivering a talk during the upcoming Botconf 2018 being held this coming December 4-8 in Toulouse, France. During this talk, Fortinet will be detailing the reuse of Mirai source code and the effect it has had upon the development of other botnets in their presentation titled 'Mirai: Beyond the Aftermath.'"

So anyway, they wrote: "During our regular monitoring, the FortiGuard Labs team recently discovered a new platform" - and this is the one we've been talking about - "offering DDoS for Hire services called 0x-booter. First appearing on October 17th" - so only two weeks ago, October 17th - "0x-booter is available to anyone who signs up on the website. This service comes with a simple-to-use interface which enables practically anyone to learn and use the service. Initiating a DDoS attack is made through a web user interface which eliminates the need for direct contact between the user and the botmaster. In the attack hub interface, the details of the host or domain, the port, the attack duration, and the type of attack can all be selected before launch."

So, yes, now it's just a matter of going to a website and saying, "I'm annoyed with these people. Zap them for me, please." And if you are willing to fork over some cash, ranging $25 for a couple minutes, to sort of evidence your displeasure with someone, more for a couple hours, you can now do that in DDoS for Hire. The world we live in today. Wow.

Middle of last week, Microsoft dropped their post-Patch Tuesday, they normally do it a few days later, like on Thursday, bug fix, which are not the security updates, but just fixing their grab bag of stuff. And this was an opportunity for me to verify that this system that I've been writing about, or rather that I've mentioned, which was Windows 10 Home, which I updated to Pro in order to set a delay. I went back when I was putting this together on Sunday and checked for anything new. Got any updates for me? Nothing. So I thought, oh, that's interesting.

So I set the delay down to zero and asked again. And sure enough, I got KB4462933, which was available, but had been held off because that's what I had asked for. So I knew it was there, so I was able to verify in fact. And the list of what it fixes is really, I mean, it's really extensive. But to give you a sense for it, it addresses the redenomination of local currency that the Central Bank of Venezuela implemented to enter the Bolivar Soberano into circulation. Because of course you'd want that. Addresses an issue that prevents the clock and date flyout from appearing when the region format is Spanish and the sorting method is traditional. Because not having that flyout would be a problem.

Addresses an issue that causes the GetCalendarInfo function to return a wrong value for the Japanese era. Addresses an issue in which applications have handle leaks when using client authentication certificates with the TLS protocol. This issue occurs when the FreeCredentialsHandle call occurs before the DeleteSecurityContext call in the application code. And things like that. Addresses an issue in which Scheduled Tasks configured to run on a specific day of the week don't execute at the expected time, and so on.

Anyway, so, you know, there's things, I mean, and that's like a tenth of all of the stuff that it fixes. So just a grab bag of things for Windows 10 that they had on their list of things to fix that they did not put into the security update because these are arguably not security-sensitive things. Just fixing little cruft around the edges of Windows 10. So I've heard no reports. That was released on Thursday. No reports of it causing unexpected problems. I haven't had any, and I did immediately reset my hold off day counters back to, I think it's two weeks for security and a couple months for the feature update because I definitely want to wait till everybody else has gotten any arrows in their back, if they're going to, before I hurt this system that I get to talk to our Security Now! listeners over every week. So anyway, that.

Firefox has made its move to v63, which now brings the built-in tracking protection that we were foretelling of. In their update they said it shouldn't be hard to own your life online. So they have now built-in tracking controls. And after you update, and you go to a site which is attempting to track you, up comes this really nice multipage tutorial. It's interesting because I have - I can't think of it. I'm blanking on it. Oh, uBlock Origin. I have uBlock Origin installed. And so I was somewhere, and I wasn't seeing anything that happened. And so I turned off uBlock Origin, and suddenly Firefox jumped in and said, oh, look, there's something to do here. So uBlock Origin sort of was in there preemptively. When I turned it off, then Firefox said, oh.

And I got this whole, like, look at this little shield that appears in your URL bar. And we've got these little, I mean, I got a whole little tutorial. So Firefox users who update will be told about all this. However, it is still not enabled by default. I had gone in and manually enabled it in the earlier build, before they surfaced it out on the UI. So what v63 does is it brings it out to the UI as "enhanced tracking protection," they call it. So if you go under Options, you know, go into Menu, choose Options, then select the Privacy and Security tab on the left-hand side, under Content Blocking, at the section at the top is Content Blocking. There's two checkboxes, which is all detected trackers and also third-party cookies, probably both of which you want to enable. And this is very fine-grained.

So you should be aware if you turn these things on because they're being careful. It's there now. It's in the UI. But it still defaults to off. So I think our listeners who are Firefox users should probably flip that on. But be aware that it might upset some sites that could be dependent upon third-party activities, some forms of tracking, in order to function. The good news is you have highly granular control to turn that off. And also - I poked in a little bit and dug around. Under this tracking protection, where you customize it, there is an option, and I showed a picture of the screen that I got under Block Lists. And you can currently choose.

They're using the Disconnect.me block list, which is highly regarded. And so there's two. There's basic protection and strict protection. They're recommending basic. And the description says "Allows some trackers so websites function properly." In other words, Disconnect.me has found that, if they use absolute strict blocking, there are some known problems with that. But again, a user who really, if that's what you want, you can choose the non-default strict protection by clicking that link, and then select Disconnect.me strict protection.

But then sort of be aware that, if something doesn't seem to be working, again, you are able to still do per-site overrides in order to allow it when it's necessary. And as we know, we've seen some benchmarks where blocking content really does produce a startling increase in speed because so many pages have become now so heavy with just stuff that they're dragging along. So I know that we've talked already about turning on content blocking and how much faster it makes things. So Firefox is moving forward.

And Leo, I've heard you on other podcasts with your guests talking about sort of this growing distinction that we're sort of presuming will exist as Firefox increasingly profiles itself as the anti-tracking, privacy-focused browser; whereas Google just can't with Chrome.

Leo: They can't, no.

Steve: Because that's their lifeblood. I mean, it's what's paying for all the free services that we get from Google. And as you say, yeah, it's their business. So it hasn't stopped people from making Chrome the number one browser on the 'Net, but it's nice to see that Mozilla and Firefox are finding a niche for themselves because I still like the way they handle tabs. I imagine I would switch to Chrome if they ever got tabs right, but that doesn't seem to be something that they're focused on at this point. Oh, and I had it in the show notes, also. If you put in the URL about:preferences#privacy, that takes you directly to that page.

Oh, and I did mention - I don't think this is worth worrying much about, but it did get some coverage around the Internet, a local privilege escalation vulnerability in Linux and FreeBSD. It's been fixed. So watch for updates from your platform's packager. An Indian security researcher discovered a flaw in the X.Org X Windows Server package which impacts OpenBSD and most Linux distributions, including Debian, Ubuntu, CentOS, Red Hat, and Fedora. Exploitation of the flaw would allow a lower privileged user to create or override a file or files anywhere on a system, including files owned by privileged users. And in this example the etc/shadow file for passwords was used as an example. However, the attacker would require a console session to exploit the issue. But SSH could be used to get a console session remotely.

So I guess the concern would be probably not an individual user at home. But if you had a system where you were depending upon the system privileges preventing unprivileged users from getting to sensitive areas of a system, where an unprivileged user could arrange to bring up a console, this could be a problem. What was found was that the log file option on the X Windows Server config was vulnerable to a format string vulnerability which could be abused by a local attacker. X.Org immediately fixed the problem and issued updates.

So it doesn't feel like it's a super critical problem for, as I said, for the typical end-user at home, but maybe in an enterprise scenario it would be something that people would want to get onto immediately. And again, it's fixed. Apparently the server exists and is invokable, even if it's not in active use throughout those distributions. So it's there and vulnerable to exploitation and so worth removing it if you don't need it. Or if you do, then updating it so that you're not in trouble.

Also, just a little bit of errata, two pieces. An anonymous listener in, wow, Cheektowaga, New York? Is that how you pronounce it?

Leo: Yeah, Cheektowaga.

Steve: Cheektowaga. Cheektowaga. Never said that word before, Cheektowaga, New York. And apparently I just misspoke. He said: "As you said it, slash dot dot doesn't do anything special." And, yeah, that would be true. He says: "Slash would be the root directory, and the parent of the root is the root because it really has no parent." So, he says: "You probably meant to say dot dot slash," which is definitely what I meant to say. So I guess I misspoke. So I meant to say dot dot slash because, as we know, the dot dot backs you up a level in the directory hierarchy. So dot dot slash, dot dot slash, dot dot slash moves you successively back up toward the root in the tree. And I guess I said "slash dot dot" by mistake. So thank you for that correction.

And a number of people also noted that in my discussion of Real-Time Operating Systems, RTOSes, remember FreeRTOS was the one that I was talking about, I guess it was last week, one of the things I failed to mention was that real-time means real-time. That is, it's often one of the other characteristics of it, and I failed to talk about it at all, which led some people to say, "Steve, what you really were describing was an embedded OS; wasn't it?" And so they certainly have a good point.

One of the other things you get in what is a real-time operating system is real-time guarantees of response. For example, there's no such thing in Windows. It's like, it's sort of best effort. If things aren't working right then, or if another process is hogging all the processor, for example mining, if you have a miner, a coin miner running on your system that is misbehaving, the rest of your system is really sluggish because it's sucking up all of your processor resources. In a real-time OS, one of the things that microkernel, that minimal set of services guarantees is guaranteed response. Meaning, for example, that you often have hardware interrupts that need real-time service.

And so to do real-time audio playback or whatever, like running a robot arm, so you want to involve the processor in the servo control loop of a robot arm, so what it needs is you absolutely have to provide a guarantee that within X number of microseconds or milliseconds, you guarantee that the interrupt service routine will get run in order to perform the actions that you're requiring in real time. So that's another set of characteristics unique to a real-time operating system, separate from an embedded OS which may not necessary provide those, and certainly any of our consumer OSes where it's just like, well, gee, things seem slow. I guess I'd better get a faster machine or give it more memory or something.

And also, Leo, I wanted to take a moment, since I thought we would have time, and we do, to talk a little bit about "Salvation," which during my trip I took an iPad Mini with me, which is now my favorite reading device. And by the way, for anyone who doesn't know, the Kindle for iPad got the feature I most love on Kindle, which is the ability to smooth scroll. It's why I use iAnnotate, and they also have an iRead, I guess maybe it's iReader. That's my PDF reader of choice because it doesn't page the PDFs. It just lets me smoothly scroll through them, which I really prefer. And until a recent update, the Kindle app on iOS, well, for the pads, or for the phone, was still a page at a time. They added the long-sought feature of smooth scrolling, which I really appreciate.

Anyway, Leo, you went through the Commonwealth books, and there have been several of them. First was "Pandora's Star," and then "Judas Unchained," which was a fabulous pair of books, where we learned about the Commonwealth. That was Hamilton's reality where what we learned was that, in this Commonwealth, the Commonwealth was basically built on wormhole technology. Some of the main characters had figured out how to create a portal which could be used to link planets. And so cleverly, what I like about Hamilton, and I'm now seeing this pattern, is he'll take a fundamental innovation and build a fully realized reality around what if that was true?

And so there were cool things, cool consequences of what if you actually had wormholes that were expensive to maintain because they had to use to vast amounts of energy, and

huge complexes had to be built around them in order to maintain this spatial energy warp of some kind, to zero the distance between distant places, the presumption being this is a lot of effort to do, so we're not going to have many of them, and so they're expensive. So what you do is you bring back trains. And that's what he had. He ran train tracks through these wormholes, and so that's the way you efficiently used this expensive yet cool transit system, the idea being that you'd have switchyards on both sides where you'd be staging both passengers and goods, and thus keep these very expensive to create and maintain wormholes busy, by using trains.

Okay. So that was the Commonwealth. In this new, and I guess it would be called the "Salvation reality," the similar sort of innovation are quantum entanglement portals. And so they're like portals which are, unlike wormholes, lots of, like, I mean, you need installations. You need sort of like mountain-size installations with lots of power and reactors around them and so forth. These things, these quantum entanglement portals, they're almost free. They're cheap. So imagine you have this portal of varying diameters, which is inexpensive. And you create it, and then you pull them apart.

And so they use quantum entanglement, I mean, we know what that is in theory. So his extension is, okay, they're just doorways that you're able to separate in physical space, yet it's still a portal that has zero space if you go through it. So of course they're used on a planetary scale. But it turns out what he realizes, okay, if you actually had these, we would use them for all kinds of things. So, for example, in this future, virtually all transportation has gone away. It's just on foot now, and you have hubs of portals that link you to other hubs after you just cross through a hub room. And so after crossing through three or four interlinked hub rooms, you're pretty much wherever you want to be.

And the very wealthy have homes where the home is a hub room, and the different rooms of the home are on different areas of the Earth or on different planets. Like the kids' room is on the moon, and it's a big crystal dome where the kids have their toys, and they're looking up at the Earth. And when you walk through the door, you're now back on Earth. Anyway, the one thing that I thought was so clever about him is imagine you want to bring a large portal somewhere, that is, by foot; but it's a large portal, and so you can't.

Well, again, being so clever, he has this concept of threading where in a backpack or something you can carry, you bring a small portal which is quantum entangled to a portal back wherever, at the headquarters. And once you get to where you want a big portal to be, you engage a device which pushes a very wide and very narrow portal through the small portal. Now you have an elongated, a very long and narrow portal, had to be narrow to be able to come through the small diameter portal. So now essentially you have a portal that's a long slot. And so the next phase is something, some equipment, a threading, a piece of threading equipment at the other end, now pushes a full-size portal through the slot. And now you have a large-diameter working portal wherever it is you went.

So again, that gives away none of the plot, but it's just some of the cool technology that he has created in this very richly realized reality. And props to him. I'm having a lot of fun with this book, which I had a lot of time to read during my recent trip to visit my high school friends.

And Sunday, as I was pulling all this together, I found a neat note from actually somebody in your town, Leo, in Petaluma, Ben Willerson. The subject was "It really does fix SSDs! Thank god!" And I realized he probably meant that maybe literally. So on the 23rd he wrote: "Dear Steve and Leo," he says, "everyone says they love the podcast. I haven't said it before, so add me to that list. My daily commute is hell," he says. "Since

Leo is in Petaluma, he knows. Although you guys don't fix that, you make it more tolerable."

He said: "My SpinRite story will not be as important to you as it is to me. It could not be. But I hope you find this since others need to know what happened. Several years ago I switched to an SSD to get more space, but also since I figured a solid-state storage drive would be failure proof. What could go wrong? It cannot crash. There's nothing but reliable solid-state memory inside. So I was less careful about keeping backups. I did more at first, somewhat.

"But everything was working until last Monday morning when I turned the PC on after the weekend, and it said 'Missing operating system.'" He said: "I had done a huge amount of work using that computer, and all that data was only there since I had stopped backing up." So I see. So he'd done a huge amount of work that was only on that computer because he hadn't backed up in a long time.

He said: "The first thing I tried was moving the SSD to a different computer to see if I could at least get back the data. But the other computer wanted to format the SSD, and it was not accessible at all. My next thought was SpinRite. Like you have said, it can be used for maintenance to keep this from ever happening in the first place. But it has also recovered other people's SSDs." And he said: "I was thinking about that security camera crashing story," which we talked about a couple weeks ago.

He said: "The subject of this email gives away the ending. I bought SpinRite from your website, and it was running on that SSD within a few minutes. I don't know whether you believe in the power of prayer, but I do." He says: "I write this letter because of the relief I felt when the entire drive reappeared, with nothing lost, after SpinRite finished with it. Bless you and bless SpinRite. Thank you." And Ben, thank you. Thank you for sharing your story with us.

Leo: Nice.

Steve: And one little bit of closing the loop from a listener, and then we'll take our final break, and we'll talk about securing vending machines. And this was a good point. Andrew Cooper in Sydney, Australia said - his subject was "Inside the Intranet should not be considered safe." And he said: "Steve, first let me say that I've been listening to Security Now! almost since it began. I really enjoy the way you cover topics from the high level right down to the technical nitty-gritty.

"However, something's been niggling at me for a while. There have been many times when you've been discussing a vulnerability in some protocol or service that you've implied, or explicitly said, that as long as it's not exposed to the public Internet, then it's okay. I heard it again today in your discussion on the LIVE555 server on SN-686. This may have been true in the past, but it hasn't been for a while.

"For many years now, the larger security-conscious organizations have known that it's not safe to trust the Intranet. It started over 10 years ago with talk of APTs (Advanced Persistent Threats) and the 'disappearing perimeter,' and today has developed into a mindset that informs all security discussions and decisions. Microsoft's phrase is 'assume breach.'

"While this mindset is especially true for medium and large organizations, it should also be a consideration for smaller networks. Even if your employees or family members are 100% trustworthy, a well-executed social engineering attack" - and again, Leo, what you were talking about at the beginning of the podcast with this new sponsor - "could result

in some code running on a PC, or even a phone, that can then use a vulnerability in an internal service to dig its way deeper into the network and pivot to other targets." And of course we know that was what the Sony attack - exactly was the profile of what bit Sony.

"Of course, this mindset needs to also consider the value of potential targets and the likelihood of attack. It's probably overkill for most home networks and even some small offices; but I don't think it's helpful to encourage the thinking that if it's" - he wrote "being," but he meant "behind" - "the firewall, it's okay. That said, thank you for all the hard work you put into the podcast every week. Keep it going. Regards, Andrew."

And point taken. And so I wanted to share that with our listeners because certainly - and I did mention that earlier in this podcast, that for example in the Linux privilege escalation attack, if you had somebody you didn't trust on the Intranet, and you had a Linux machine where you were depending upon its security boundaries, that could potentially create a breach. So I will be more careful about that in the future. Thank you, Andrew. And a good note for our listeners, as well.

Leo: Indeed. Okay, Steve. Let's hear it. I've got to know.

Steve: Okay. So first, again, the idea being the minimum solution.

Leo: Yeah.

Steve: So first what looks good, but doesn't work. So imagine you designed a system with no WAN communication required for spending money from the smartphone. That sort of seems like what we were talking about. Remember, basically there was an app on which you could have - where the app could hold the balance. And apparently it was a negotiation just between you and the vending machine. You selected what you wanted, and the vending machine communicated over Bluetooth or near field and essentially deducted the money from the wallet that the app contained and said, "Enjoy your candy bar." And of course the app was reverse engineered. The guy was able to give himself 999 euros. And so, whoops, this wasn't very secure.

So fixing that problem is not hard. And so what we saw was apparently a system where the phone wasn't connected to the Internet, the vending machine wasn't connected to the Internet, and that the sole communication link was between the phone and the vending machine. The problem was the wallet was not secured. So securing the wallet is trivial. We have well-established, simple, and inexpensive hardware which can protect a secret key from being disclosed, the so-called TPM, right, the Trusted Platform Module, which is a version of an HSM, a Hardware Security Module.

And we didn't talk about what is the attack surface. Do we want to protect the system from the vending machine itself being physically attacked? And you could argue that, well, if you're going to crack open the vending machine, you can take the candy bar, so you don't have to buy it. But a larger attack would be to steal some cryptographic secrets which would then allow you essentially to have the equivalent of attacking all the vending machines that shared that secret. But it's just not difficult to protect a secret. Our iPhones do it. Our motherboards that have a TPM chip do it. And it's no longer expensive. Those chips are on motherboards, and they're available for pennies.

So let's assume we can keep a secret in the vending machine. That is, proof against attack. Well, that means that all of the vending machines could securely share that one secret, and that that one secret doesn't need fancy public key crypto. It could be just a

symmetric key. It's just a secret key which is either used to encrypt the wallet or to sign the wallet. But doing either of those things completely prevents the wallet from being tampered with. That is, it would completely defeat this attack where any reverse engineering of the app could in any way alter the wallet so that the alteration was not detected.

If it was encrypted, the wallet would just be an opaque blob. It would just look like static that you couldn't do anything with. If it was signed, then you could see the contents, but you couldn't touch it. You couldn't alter them, or that would break the signature. And the point being that when the wallet was handed over Bluetooth or near field to the vending machine, it would just reject it. It would say sorry. It would use its secret key to check the signature and say there's something wrong with the wallet. Go deal with headquarters in order to get your wallet repaired, and then come back if you want your candy bar.

So that problem, that is, this tampering with the wallet problem, is easy to fix. But it doesn't address the actual security. There's still a flaw here. And that is, in this model, even if you've made the wallet tamper-proof, you haven't solved what is known as or could be known as the "double-spending problem" because nothing would prevent an attacker from making a copy of the wallet before they buy the candy bar. So they take the wallet and make a copy of it; right? Then they buy the candy bar. The vending machine deducts the cost of the candy bar, gives them back a re-signed wallet with a lower balance, and they simply replace the wallet that has now less money in it with the copy that they made before they bought the candy bar. They've essentially reset the wallet's balance, and the whole vending machine system would have no way of knowing that had happened. So you just buy another candy bar and do that forever.

So while we address the first issue of wallet tampering, we haven't really solved the problem of creating a secure vending machine system. The existence of the double-spending problem demonstrates that there is no means by which the user's balance can only be stored in the app, in this case in the smartphone. That is, you absolutely have to somehow involve another party. There's just no way to prevent double spending if you can copy a wallet that has money in it and reset it after you've taken the money out.

So if every vending machine did have a connection to the Internet - and again, we don't know that that's the case. We're assuming it's not because, if every vending machine was on the Internet, then this sort of double spending could be trivially prevented by having the vending machine essentially reach around the user to contact headquarters, and the user's balance would actually be maintained by the vending machine operator headquarters. And so, again, then any kind of fraud of this sort could be prevented.

Essentially, the app would then be reduced to the role of identifying the user whose account and balance is maintained somewhere where the vending machine is able to access it and deduct the amount from the wallet there. But we're assuming this is not the case, that is, that the vending machines are being kept simple, and they're not connected to the Internet, which is why they were so easily hackable by this first-generation attempt at an app.

So let's assume the vending machines do not have a connection to the Internet. All they have is a connection to the user who's standing in front of them with a nearby connection, with a Bluetooth or near field connection. If that's the case, we still need a third party. Not the vending machine, not the user, but we need to get in contact with headquarters. And since we cannot trust the app, we have to use the app as an intermediary. It's the link between the vending machine and headquarters.

So now we have a system that once again works securely. The user says, hey, I want a candy bar. So the vending machine prepares a transaction which it signs with its secure

key. It's got to have that because there has to be something that the app doesn't know, and that's the secret stored in secure hardware, in a Trusted Platform Module chip, for a few cents these days. So the transaction description is signed by the vending machine's network secret. I mean, it could be the same secret in all the vending machines. It doesn't have to be anything fancy. There's no need for a per-vending machine secret. It could have one if it wanted, but unnecessary. We're wanting to keep this simple.

So it signed that with its secret and uses the Bluetooth or near field link to give it to the phone and say, if you want the candy bar, you've got to check in with headquarters and give it this signed transaction. Again, the app can't mess with it, can't fuss with it, because it's a cryptographic signature. It could see it if it wasn't encrypted. It could be encrypted if they wanted the additional security. And when I was thinking this through, I was thinking, and you know, it's a vending machine; right? It's got limited quantities of candy bars.

So if it's going to be sending a packet back to headquarters, it ought to send its current inventory along with it. So that way headquarters can keep track of how many candy bars are in the machine and send someone out to refill the vending machine if it's getting dangerously low. But anyway, extra feature. So the phone has this transaction. Oh, I forgot one part, very important. The vending machine also has a nonce. It generates a - it doesn't even have to be encrypted, really. It could just be a counter. Doesn't really matter because it doesn't have to be secret. But it absolutely has to have a nonce - that is, as we know, that's short for "number used once" - a token which it embeds in this transaction, and that's to prevent replay.

Because what happens is this transaction is going to go to headquarters, along with the user's ID who wants the candy bar. And so it uses the smartphone link. We know the smartphone is on the Internet. Vending machine doesn't have to be. User's smartphone is certainly on the Internet. So it sends it to headquarters, who looks up the user, checks their balance, looks into the transaction, says, oh, candy bar, 25 cents. Deducts that from the balance, assuming that there is a balance there. Authenticates the transaction and re-signs it with, again, this shared secret that headquarters and all the vending machines have. So this authenticated, approved, and re-signed transaction comes back to the phone.

Of course, the user just - this all happens instantaneously from the user's perspective. The phone now uses its Bluetooth or near field link to send the approved transaction back to the vending machine that checks the signature. The approval could only have been created by somebody that had the secret also. And it could be a different secret, if you wanted to have multiple secrets. But again, no need, keeping it simple. Checks with its secret, verifies it, and then turns on the motor that rotates the screw and drops the candy bar out for the purchaser to pick it up.

At that time, the nonce which had been issued on the transaction which was pending is removed from an outstanding purchases list, probably only one at a time, but it might have two or three people buying things, I guess. So you might have multiple outstanding transactions, but you probably don't need them. But the point being that, by having a nonce, which it has now deleted from its outstanding nonces list at the time that it accepted the approved transaction, the app is no longer able to resubmit an approved transaction because the nonce no longer will be accepted by the vending machine because it already issued a candy bar from an approved transaction with a nonce.

So anyway, a nice little fun problem in crypto protocols and engineering, where we engineer a system where the vending machine's cost is kept low because all it has is a very inexpensive secret key, unhackable because it's in its hardware. It does not need to have the additional complexity and cost of its own connection to the Internet, yet it's able to interact with a nearby user's phone which does have an existing connection,

transact through an app where the app is reduced to an intermediary that has no ability to alter anything happening. It can see it, if it wants to, or it could be encrypted if there's some benefit.

And as a little extra feature we threw in the ability for the vending machine to continuously inform headquarters of how much stuff it has in its physical goods inventory so that headquarters can dispatch someone out to refill the machine if it's getting dangerously low. And so our listeners can compare the system that I proposed with their own solutions and see if they had a better idea. I can't think of anything better than that.

Leo: So a couple of people did get that, though, figured it out.

Steve: Cool.

Leo: No, you're saying. Oh, you don't know.

Steve: Oh. Oh, no. Yes, there were some people who I saw. Certainly the gang that hangs over in crypto land...

Leo: They figured it out.

Steve: ...at GRC, they were on the ball completely, yeah.

Leo: Wow. That seems like the obvious way to do it.

Steve: Yeah. There are, I mean, and that's what we've seen with this sort of stuff is we've got some - what I love about crypto is that they are simple little modular things like signatures and asymmetric or symmetric keys and hashes and, you know, some cool components that you get to assemble in different ways to solve, usefully solve problems. But they're unhackable once you apply them correctly. And unfortunately, I mean, the reason we have this interesting little thought experiment is that, boy, the company that did it, gave it the first try, really didn't put much time into it apparently because, yeah, they had a very insecure and very easily hackable solution.

Leo: A friend of mine runs ice cream vending machines is his business. And the way they know when it's time to refill, they have a little scale in the freezer. And they know the weight of the machine.

Steve: Clever, clever, yeah.

Leo: And that's sent back to the home office. And of course you might say, well, then, how do they know which ones to bring? Well, the truck always has all of the stuff. And there are certain pretty predictable patterns as to which ice creams sell faster than others.

Steve: Ah, nice.

Leo: Yeah, he's got one of those - if you see it, if you go to a convenience store and see those, they're like, I think it's Hershey's freezers there with the ice cream treats in there.

Steve: Ah, right.

Leo: Little scales on the bottom so he knows when it's time.

Steve: Nice.

Leo: Send a guy out to refill.

Steve: Low-tech, but effective.

Leo: Yeah, it works. He also bought all of the pay phones from AT&T because it was too small a business for - this guy's a brilliant guy. It was too small a business for AT&T. But payphones are still widely used, especially in poorer communities. He retrofitted them with cell phones, and it's a good little business. It's a little cash cow, yeah.

Steve: Nice.

Leo: Poor guy's got to roll those quarters, though. But other than that. It is time to wap this up, as they say.

Steve: Yes, Donald.

Leo: Thank you [Elmer Fudd laugh], thank you so much, Steve Gibson.

Steve: Daffy. Daffy.

Leo: Daffy. I don't know who it is. Hello. It's Elmer Fudd, actually. [Elmer Fudd laugh] Steve is at GRC.com. That's his website, GRC.com. And that's where you'll find this show, 64Kb audio plus human transcriptions, human written transcriptions.

Steve: And the recently blessed SpinRite, Leo.

Leo: And the recently blessed SpinRite, the world's best hard drive maintenance and recovery utility. Also lots of good free stuff, too. He's very good about that. GRC.com. He's @SGgrc on the Twitter. That way if you have something to tweet to

him, you can do it. He even accepts direct messages from all sources. If you want video of the show, you can get that at our website. We have video and audio at TWiT.tv/sn of all the 687 episodes on record, and a few that aren't on record. And you can also subscribe. Best thing to do really is to get your podcast application to subscribe to Security Now! so you don't miss a minute. And you will have it all forever. My friend, we have wapped this thing up.

Steve: We have come to the end of another podcast. Until this time next week - oh, my goodness, and that's Election Day in the U.S.

Leo: Oh. I want to see that "I Voted" sticker on every single person.

Steve: That's a biggie, yup.

Leo: 100%.

Steve: I'll be proudly wearing mine.

Leo: Yeah. Thanks, Steve. We'll see you next time.

Steve: Okay, my friend. Bye.