A Very Busy Week

Description: Father Robert and I discuss Brian Krebs' forced move from Akamai to Google's Project Shield, Yahoo's record-breaking, massive 500-million-user data breach, and Apple's acknowledged iOS 10 backup PBKDF flaw. A well-known teen hacker jailbreaks his new iPhone 7 in 24 hours. Microsoft formally allows removal of GWX. There's a new OpenSSL server DoS flaw, also more WoSign/StartCom woes as Mozilla prepares to pull the plug. BitTorrent Sync is renamed and more deeply documented. Then we have a bit of errata, some miscellany, and 10 questions and comments from our terrific listeners.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-579.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-579-lq.mp3

SHOW TEASE: It's time for Security Now! with Steve Gibson. 1Tbps DDoS attacks are on the horizon; it's time for you to say goodbye to Yahoo!; iOS 10 is 2,500 times easier to crack; and it's judgment day for misbehaving CAs. It's all coming up next on Security Now!.

FR. ROBERT BALLECER: This is Security Now! with Steve Gibson, Episode 579, recorded September 27th, 2016: DDoSes, breaches, and other records to be broken.

It's time for Security Now!, your respite from the increasingly insecure world in which we live. It's the show where we take a deep look at the hot topics and the newest exploits and the most frustrating worst practices in the security world with our Explainer in Chief, Mr. Steve Gibson. Steve, of course, of GRC.com, the man who created ShieldsUP!, SpinRite, and SQRL, fine security products that we all use. Steve, again, it is a pleasure to work with you, my friend.

Steve Gibson: Hey, Father Robert. Great to be back with you for our second of three recordings with you of Security Now!. And it's interesting, when you were saying "increasingly insecure world," I was thinking, okay, is it increasingly insecure, or are we just more aware of the existing, always have been there insecurities? And I think both are true.

FR. ROBERT: Absolutely.

Steve: And unfortunately I think, as we keep adding features and new capabilities, the lesson is we're introducing vulnerabilities that didn't exist before. So there's both cleaning up the new messes and catching up with the old messes. So oftentimes we're talking about some OpenSSL problem that's been in there for 20 years, and no one had found it. So in one sense we're less secure because now it's been found. On the other

hand, then it gets fixed, and so we're more secure than we were, even though we didn't realize before that we were less secure than we thought. So if that doesn't confuse you, this podcast will.

FR. ROBERT: Steve, this is actually a very common trend inside of any kind of reporting, not just the tech world. But it's the whole idea that, since the news cycles now go 24/7, and not only that, but they go beyond because anybody can be a news source now with social media, you find out about things a lot more quickly, and you find a lot more hyperbole in our standard news sources. And I'm with you. I think a lot of these exploits have existed forever. It's just now they're being brought to light. But there is one thing, and we're going to talk about this later on in the show, that I think goes on the other side, where it is more and more insecure. And that is the Internet of Things, the exploding Internet of Things, as it's really changed a couple of the equations as far as what we think is secure and what is not.

One quick, though, before we go, I can't believe this. This is your Q&A 240. This is 240 Q&A, Steve, of 579 episodes? That's amazing.

Steve: Yeah. I didn't realize that we started that as early on as we did. But I was beginning to get feedback from people. And I thought, you know, it's really great to hear from our listeners. It engages our audience, and it helps me to better understand, like to steer the podcast and understand the sorts of things that we need and should cover. And this week we've got a bunch of stuff. We've got to, of course, talk about Brian Krebs' problems with a big sustained DDoS that finally forced Akamai to say, "Brian, we're sorry, we cannot continue hosting you." That generated a lot of controversy and some interesting social commentary, as well. And of course he got picked up by Google's Project Shield that we'll talk about.

Much abuzz in the news was Yahoo's - they actually broke a record, and not in a good way - half a billion user record data breach. There are some interesting details about that, that I haven't seen much covered in the press. Apple did break something crucial in iOS 10's backup. We'll talk about that. Within 24 hours of getting his new iPhone 7, a teenager, hopefully after he finished his homework, jailbroke the phone. Microsoft formally offers the removal of the Get Windows 10 - I don't want to call it malware - annoyware. There's a new OpenSSL DoS flaw, which essentially allows a remote person to crash a server that's based on OpenSSL.

More problems with WoSign and StartCom relative to Mozilla. We covered this a couple weeks ago, that is, looking at Mozilla's very careful march forward, laying the groundwork for basically pulling the plug on their support for WoSign and StartCom's certificates. BitTorrent Sync is one of the most requested for podcasts, that is, for me to talk about it. The problem is they absolutely refuse to provide documentation for the protocol. There is a new paper out, still doesn't give me what I want, but it's probably enough that I can give it a podcast and talk about what we do know, with the caveat that, well, without the details, there could be something hidden. But their heart seems to be in the right place.

We've got a little bit of errata, some miscellaneous stuff. And then, being a Q&A, as you say, the 240th one, we'll look at 10 user/listener-prompted comments and questions and discuss those. So I think another great couple hours for our listeners.

FR. ROBERT: That's absolutely amazing, Steve. Just the amount of security-related stories that have broken this past week and a half or so have been fantastic. As you mentioned, everything from Yahoo's very, very unfortunate record to Akamai. And actually that's a very serious problem which, as you mentioned in your opening

comments, is actually a problem that's been around for a long time, it's just being amplified by the emergence of new technologies. Now, Steve, Brian Krebs had a very interesting week.

Steve: Well, Brian has been, of course, we talk about him often, he is a leading researcher whose sort of focus is security breaches in the private sector. And he seems very fascinated by the whole underground hacker world. And he's also very outspoken. So he tends to be a frequent target of denial of service attacks. So finally he sought greater protection and arranged a deal with Akamai, the well-known content delivery network, in order to essentially have them host his site. They provided caching services.

The problem with a denial of service attack, which we've covered through the years, is a bandwidth concentration. That is, you typically have widely scattered clients, and they're clients inasmuch as they are targeting a server, trying to flood that server. And so the traffic bound for that single point of service, as it jumps from router to router, heading in toward that server, the traffic is aggregated into larger and larger streams until, at some point, the devices trying to feed that server are no longer able to handle just the raw brute force size of the traffic.

We've often talked about the way routers work. We're talking about buffer delays and queuing packets and so forth. So if you've got a router with the same speed links in and out, and you've got saturated links, like five saturated links coming in, all trying to route traffic through the sixth link, well, the math is simple. You can't squeeze that 5x traffic through a 1x link. There just isn't a way. So the router understands this. The technology is very robust. Its buffer of packets overflows, and it starts dropping packets. What legitimate users experience is an inability for their little microscopic, by comparison, trickle of packet traffic which is trying to access a site. It just can't compete, statistically, with this flood of what looks like valid traffic, but is designed in order to flood these aggregation points.

What a content distribution network like Akamai uses is an interesting technology. It's something we've never really talked about. And that's anycast routing. We've often talked about how the job of a router is, when a packet comes in, it looks at the destination IP. And it's got a routing table which contains a series of masked IP addresses which are associated with different outgoing interfaces. And so its job is to look at the incoming packet and figure out where to send it, that is, out of which interface that packet should then go.

Well, this is done in a routing table by matching the most specific route, meaning the route which matches the packet with as large a mask, that is, all the one bits coming down from the top, and with a small - a hostname. Well, what that means is that, if it's configured correctly, you can put servers all over the Internet, that is, well, I'm sorry. I interrupted myself. All over the Internet with the same IP. And that changes the topology.

Now, even if attackers all over the world are all attacking the same IP address, rather than it physically aggregating at a single location, all of these distributed routers have short routes to local Akamai servers, for example, using Akamai as an example. And so they end up handling the traffic locally and prevent that single point of ultimate failure because no links these days are able to sustain the kind of denial of service attacks that we're seeing.

So I read a lot of the coverage in the news. And people who support Brian and think he's doing a great job were upset that essentially this attack successfully knocked him off the 'Net. I think it was, like, last Thursday he announced that Akamai was suspending his

service. I guess, being neutral in this, I understand, certainly I understand Brian's disappointment that a huge sustained DDoS attack was able to push him off the 'Net. But there is a non-zero cost that any bandwidth provider is bearing when they absorb that kind of traffic.

We've talked a lot in the past about peering agreements between top-tier providers. And the agreement is that there will be pretty much a balance of traffic ingress and egress across peering relationships. But an attack like this completely flips that. And we've talked about, for example, the problems that Netflix has had because that's an example of a service in the evening that dominates Internet traffic in a decidedly one-way direction.

And I remember when I was setting up my relationship with Level 3 about a decade ago, they asked me, you know, what's the ratio of your inbound to outbound traffic? Because they would like to have it be 50/50. Well, no server is. GRC has a lot more people pulling things out than it pulling things in. In fact, there's like zero inbound except just the request traffic going in that generates much larger reply responses.

So anyway, what Brian ended up doing after being off the 'Net for a while is he managed to get Google's Project Shield to pick him up. And this is something we talked about briefly, but there wasn't much - I guess we covered it when it was announced. And it's Google's sort of, in the same way they do everything, they're exploring protecting sites that, under their own criteria, they think deserve to be protected, to be on the Internet. For example, sometimes, as in the case with Brian, DDoS attacks are used to silence individuals or news organizations or dissidents or whomever who are taking a controversial position, or whom somebody somewhere wants to shut up. And so Google has said, for free public news profile sites, we will experiment with this thing we call Project Shield.

So on their page they say: "Advanced DDoS protection. Project Shield is built on Google's infrastructure, creating a multilayer defense system to protect your site against DDoS attacks, including layer 3/4 and 7 attacks." That is to say, 3 and 4 are just sort of raw brute-force bandwidth, just flooding attacks, and layer 7 is protocol level. And, see, that's the problem with some of these later attacks is, in the old days, they just used to be SYN floods. And the infrastructure got strengthened, so it got smarter about allowing endless numbers of SYN packets, that's S-Y-N, short for synchronized, the first packet in a TCP connection setup.

So what's happened is attacks have - there are still layer 3 and 4 attacks going on. But layer 7 means this is like just a valid HTTP query asking for content. And if enough devices scattered around the Internet ask for a server's content, making each of them a valid request, once again, you get just too much bandwidth. In this case, the incoming bandwidth probably wouldn't have a problem, but there wouldn't be enough outgoing response bandwidth that allowed everybody to get their answer. And so, like, 99.999% of the queries would be bogus. And so a little 0.0001% query has almost no chance of getting a page pulled up. So that's a different approach to the denial of service attack issue.

FR. ROBERT: You know, we've had people on from CloudFlare and AlienVault, even Akamai, come onto This Week in Enterprise Tech. And one of the questions that they get most often is why does the DoS attack still work? Why does DDoS still work? I mean, it was one of the first that was popularized. It was the first to hit the mainstream consciousness. Shouldn't we have solved it by now? And they all have the same answer, which is, well, the problem is, it's not really an attack. It's not an exploit. All it is...

Steve: It's overload.

FR. ROBERT: It's just an overload. It's legitimate traffic that you're just ramping up. And in this particular case they were able to use a lot of compromised edge devices that were able to send legitimate traffic from multiple IPs. So there was no blocking upstream.

Steve: Correct.

FR. ROBERT: Which is what we hear sometimes when you have an event that's under attack.

Steve: Correct. Now, there are sort of two issues here, I think. That is, in the press, the press has enjoyed jumping on the whole IoT problem. And we're already getting new acronyms on this podcast for various issues of Internet of Things problems. But there's another problem. And I think it's the superset of the IoT problem. That is, if you had a houseful of insecure IoT devices connected by a modem to the Internet, 1200 baud or 9600 baud, well, evil as they might be, they couldn't do any damage.

The other problem is, the related problem, the enabling problem, is consumer bandwidth is skyrocketing. Until I lost my pair of T1s, where I was limping along at 3.4Mb, I don't know how I survived now, I switched to a cable modem. Now I've got 300 downstream and 50Mb upstream. I mean, one individual, one entity has that kind of bandwidth. And of course we're seeing this escalation of bandwidth as consumers are demanding greater levels of connectivity in order to support the kinds of services we now want. But with that comes the problem that then those individual endpoints are individually far more capable of participating in devastating DDoS attacks.

So, yes, them being distributed is important. Them having easily compromisable devices is important. But ultimately, if those things can't talk with high bandwidth to the Internet, that doesn't do them any good as attacking components. Well, that problem's been solved because all of our consumer bandwidth is just going up as fast as it can grow.

FR. ROBERT: You know, Steve, also the attackers are getting far more sophisticated in terms of how they hide their attacks. It used to be when you owned a device, all you cared was that you had root access. Now you can make that device do anything you wanted it to. Now they realize, no, I want to own a device, and I want it to stay owned for as long as possible. So they will not max out an outgoing connection. They will not max out a device so that it stops responding to its legitimate purpose.

In other words, they want the person who buys the X label consumer Internet of Things security camera to be owned, and not even notice that, yeah, a little bit of your upstream seems to be leaking out every month. And that's really what's enabled this because if you get 1,000, 10,000, 100,000 of those devices, there is no mitigation service on the planet that can stop that.

Steve: Well, and I don't know how much our listeners look at their own traffic. I have a cute little chart which monitors the SNMP counters in my pfSense router. And it turns out many routers offer SNMP, Simple Network Management Protocol, that allows you to watch. When, I mean, I have got multiple iPads. I've got, yeah, you know, I've got a couple TiVos and computers that are on all the time. Point is, there is constant activity. This is like, my network, even when nothing appears to be going on, it's, I mean, I'm looking at my switch and router lights. They're never not flickering. They're just flashing all the time.

So exactly as you say, Father, if you add a few light bulbs, IoT light bulbs, or a couple cameras, how would you even know? As you say, it's able to hide in the noise. And unfortunately, we all now have very noisy networks. And I know that some people are like, okay, I'm going to figure out what all of these streams are. Well, if you don't have a life, okay, go for it. But, I mean, some of this stuff is just unknowable. It's going off to random IPs. And it's like, well, I don't know where it went or what's in there. And of course now it's encrypted, more and more, so you can't even see what's going on.

FR. ROBERT: You know, I've always suggested that people, I mean, because Wireshark is free, and you can get a gigabit tap for relatively little, or use a hub, if you can find one on eBay and drop [crosstalk].

Steve: Or just use port mirroring.

FR. ROBERT: Or use port mirroring if you've got a smart switch. And I just tell them, watch your traffic for 24 hours. Just collect packets for 24 hours when nothing is happening, when you're gone on vacation. And you will be surprised how many packets leave your network. And unfortunately [sic] a lot of that is not malicious; a lot of that is just housecleaning that the various devices on your network will do. But as you mentioned, that's a noise floor. And so unless something goes above that noise floor, my tools are not sensitive enough to know, hey, you know what, this one device, it keeps beaconing out to this IRC channel, and then it does stuff. And it's not smart enough to realize, oh, that's command and control. I should probably shut that down.

Steve: Right.

FR. ROBERT: And they'll never know. Those devices will always be owned. Those devices will always be compromised, as long as the people who control those botnets don't do something really stupid like max out their attack power.

Steve: Yeah. And so you're right. These are, while they're valuable for attackers to have, they don't want to lose them. And they're so easy to get now, that is, the fruit is so low-hanging, that all they have to do is generate a trickle out of each of their 10,000 webcams, and that gets the job done at the other end, unfortunately.

FR. ROBERT: Yeah, yeah. A fun little attachment to this. I did manage to get a visit to one of these DDoS protection outfits, one of their datacenters down here in Northern California. And it's actually, it's a wonderful, wonderful thing to see how they do it, how they use anycast, which by the way, is anycast part of the original Border Gateway Protocol specification? Or was that tacked on later on?

Steve: Oh, I was confusing it with CIDR. I know that the addition of CIDR came later because they wanted to create more networks, instead of just having class A, B, and C. I don't know whether it was part of 1.0. My guess would be it was not because it's not the way the Internet was originally set up. As we know, the Internet purists don't like NAT, the idea that we're going to have a single point that represents multiple devices sharing an IP. The Internet purists say no, that is breaking the rules. We designed this network so that every device had its own IP. And of course IPv6 will allow that to happen once again because finally we have enough bits.

But so my guess would be that something like anycast was an extension added later, when the need grew which didn't initially exist. Because it doesn't feel like this kind of addition to BGP would have been - there wouldn't have just been a reason for it, day one.

FR. ROBERT: Right. And so I use anycast if I'm going from outside in; and I would use VRRP, the Virtual Router Redundancy Protocol, on my inside to give my clients a redundant gateway. That actually makes sense. We've actually got a good comment in the chatroom from [indiscernible] who says, "So now even our dumb networks have to be secure? What's next?" Oh, he's joking, but it is actually a good point, which is in the old days, when you were you talking about when we were still on dialup modems - in fact I remember when I first got my Practical Peripherals 14.4, it was this little white box with an LCD screen in the front. Oh, my gosh, it was heaven. It was so fast.

Steve: And it was also flat so that the telephone could sit on top of it.

FR. ROBERT: It had to because I actually had the one that still had the little dial thing. I had to wire the thing in myself. But our dumb networks, way back when, they didn't have the bandwidth to become a nuisance. Whereas today my home connection has more power than my corporate connection did 15 years ago. I've got more bandwidth. So my dumb network is suddenly a bigger threat. It's like saying, well, the little kid with the BB gun can't do much, but now we're going to give him a bazooka. That's where we are.

Steve: Yeah. Exactly.

FR. ROBERT: All right. Let's get away from bazooka and DDoS. We understand that this is a problem. We understand that the ISPs are dreading the day when they start to see commonplace 1Tbps attacks, which is coming.

Steve: Yeah. And I don't see any mitigation unless we, I mean, we sort of talked about this last week. And I did get a lot of feedback from our listeners when I mentioned that the only thing I could see, if it was not legal to allow, or ethical to allow infected machines to be tampered with, then to hold ISPs responsible and disconnect them from the Internet. A lot of people like the idea. Some people said, eh, you know, ISPs are far from perfect. I'm not sure - in fact, one person sent me his own story, where Comcast had been doing that a few years ago, and he got disconnected, though he was absolutely sure that he was not at fault. They just messed up their detection algorithm one way or the other.

So the only thing I can see is that, somehow, these attacks are raising the visibility, and we're getting more buzz about needing a solution. Problem is, we're steeped in this technology. And if you can make valid, just too many valid queries, then filtering those really becomes a challenge. The only thing I could imagine would be much more caching so that, for example, the queries being made would end up hitting the cache and not go any further. But then you just do cache busting. You change the URL a little bit. The cache doesn't know if it's valid or not. The query has to go through in order to see, well, maybe it is valid, and you're back there again.

So, I mean, from everything I've seen, anything someone can propose, we've got, you know, there is an attackable workaround. And the reason, of course, going back to first principles, is it wasn't designed to be resilient against attacks. It was designed when universities and major businesses were connected to this experimental network, and everyone was amazed it worked at all, rather than, okay, we're designing something for a 50-year future horizon where the world is unrecognizable from what it is today. Well, that wasn't what they were trying to do. They were just trying to say, hey, can we send a packet from Northern California to Southern California? And they were shocked when it worked. So don't...

FR. ROBERT: That's actually the source of most of our exploits these days, which is

something that just worked, and they never thought that it would be misused.

Steve: Right.

FR. ROBERT: We trust, you know what, security folk, I mean, networking folk, by nature, before we got into all the security stuff, we were trusting folk. And now we've seen the error of our ways.

Steve: Yeah.

FR. ROBERT: Steve, let's get away from massive DDoS attacks because I want to go back to a good old-fashioned cluster breach.

Steve: Old-school security breach.

FR. ROBERT: Old-school security breach. This is just someone let everything out. And, yeah, we've heard about a lot of breaches over the past couple of years. But this one is of note, not just because of size, but also because of the circumstances behind the disclosure of this breach. Of course, this is [crosstalk].

Steve: Well, yes. So, and I'm really not - I'm not interested in the politics. The problem with that is that there's so much subject to interpretation. Just so people know what that is, the argument is that executives at Yahoo!, Marissa and company, were aware of this breach for as much as two years and didn't talk about it publicly. So of course that's really misbehavior. When we talk about problems that arise, and I don't mean to use LastPass as anything but a good example because mistakes can happen. There was a strange problem with, I think it was just Firefox browsers with the way their extension interacted with something. They had it fixed before the person - it was Tavis Ormandy who found the problem, and he was creating a timeline for, like, managing this problem. And it was fixed before he had posted the timeline.

So that's what you want. We all understand that everyone can have a problem. The question is how quickly and responsibly do you deal with it. That may be a softer standard than some people would like. People would like no problems ever. But we can't do that. So the best we can do is immediately remediate by, for example, informing those who might be affected. In this case, as many as 500 million Yahoo! accounts were breached sometime in the past, and they didn't tell anybody.

Now, the part that I found interesting was a little bit of the technical details because the good news was that Yahoo! has been using a very strong password-based key derivation function known as bcrypt, which is very difficult to accelerate and speed up, by design. It was designed for this purpose, so that it would be a memory hard problem that was resistant and resilient against brute force attacks. The bad news is that, if you parse Yahoo's announcement, it suggests that most - but on the other hand, these are people who didn't tell us for two years, so I'm not sure that their adjectives should be believed completely - but not all of their passwords were hashed with bcrypt. We don't know how many passwords were hashed with not bcrypt. And we don't know what not bcrypt they were using. That is, was it MD5? What did they obsolete and strengthen? The good news is they did strengthen something. The bad news is we don't know.

Now, we've seen, and we've covered in the past, instances where a company exactly with Yahoo's profile implemented stronger password-based key derivation; but, for the sake of backward compatibility, left the old hashes there, too. Well, okay. So as users signed in, they would migrate the old hash, they would verify with the old hash, and then they

would rehash with the new stronger algorithm. What they should do is then delete the old hash. We know that there have been cases where that was not done. But that's what you have to do in order to evolve from an old hashing scheme to a new one. That is, there needs to be a period of overlap.

And because the whole nature of hashing is that you cannot, like, on the server side, Yahoo! can't simply change the MD5, if that's what it was, to a bcrypt. They have to wait for that user to log back into their Yahoo! account. And that's the other thing, too. I mean, we don't even know how many of these were not just throwaway because Yahoo! is the classic, oh my god, I want to post something to this blog. Must I create an account? Well, I'm not giving them my real email. So you just get a throwaway account. Yahoo! has been a great source of those.

So the idea being, though, that in order to advance the hash, they have to wait for somebody whose account is currently weakly hashed to provide the plaintext password to them. They then hash it with the old algorithm to verify that that's a proper login. Then they take the same plaintext and rehash it with the new one and thereby upgrade. So again, it's frustrating that there's so much still not known. So the takeaway for everyone is the greatest danger would be probably that you've had a Yahoo! email account for a long time, like ever. Did you ever make one? And did you ever reuse your password?

And unfortunately, everyone used to. Monkey123 was popular for a reason. And so if it's both old, it is probably the password, like what used to be your password, "Oh, this is my password, and I use it everywhere I go." It's like, well, we've known for a long time you can't do that. But if there's any other instance where you may have reused that password, that's the vulnerability because, from this breach of personal information, it's probable that, from what they said, a percentage of those accounts were weakly hashed. And if those were hashed with a weak algorithm, MD5 for example - and Yahoo's been around long enough, it probably was once MD5 - then if you had reuse, that's probably where you're vulnerable. And so you definitely want to change the passwords on any accounts, any non-Yahoo accounts that might possibly have ever shared it.

FR. ROBERT: You know, Steve, I'm like you. My Yahoo! accounts were always my throwaway accounts.

Steve: Yeah.

FR. ROBERT: They were the ones that I had just because I wanted a place for junk to go. And I have five of them, and none of them have been checked within the last seven years, would be my guess. But I went back in, and I changed all the passwords. When I was at DEF CON, though, there was a very nice gentleman who was explaining to me that he specialized in looking for those throwaway accounts because, he says, it's got a couple of points that make it very dangerous. First, throwaway accounts typically don't have really strong passwords because that's something that someone wants to remember. So it is a reused password. Secondly, throwaway accounts, because they are discounted as to their importance, people forget...

Steve: [Crosstalk] passwords?

FR. ROBERT: Well, people forget what services they had attached to that account.

Steve: Ah.

FR. ROBERT: So if I sign up for a service with a throwaway account, and then I just

forget about the throwaway account, that email's probably still in there. So they'll know what services they can get into by requesting a password reset, and it will go back to the throwaway account. And it was fascinating. I talked to this guy for about 45 minutes, and he was telling me about all the different ways that he could use what most people consider digital refuse to ultimately get into that top layer of accounts, the stuff that was actually still active.

Steve: Well, and I did see one interesting piece of advice that I appreciated regarding this. And that was don't retain old email only because you can. Following on from your example, Padre, it is important to recognize that those confirmation emails and so forth, that is a rich repository of your history which is probably not entirely obsolete. Probably not entirely useless. So the fact that these gargantuan email services allow you the luxury of never having to permanently delete something, that's, I mean, that's a mixed blessing. It can be useful, but it can also bite you.

We've talked about, I want to say Mail Home. MailStore Home is my favorite little local archiver. And it's what I do for all of my various GRC-based accounts and the other stuff I'm doing, is it's a beautiful indexed database. And so I run it periodically. It sucks down my email and indexes it and archives it and makes it searchable here. So I still have all the benefit of everything being available, yet not the liability of that everything being out in the cloud. And here's a classic, perfect example of why that's not safe. So if you've got those Yahoo! accounts with lots of old email sitting in them, somebody who can get in there can learn more about you than you would like them to.

FR. ROBERT: Right. That might actually be an interesting exercise for our audience, to look through their throwaway accounts to find out what's been left in the deleted items folder, or what's been left in those folders that you filed away on Yahoo!. See how many other accounts those can actually lead into because I'm betting there are quite a few. In fact, on mine I realized that one of my very first throwaway Yahoo! accounts I used to verify a Google account because Google originally, way back when, required another email address for you to be able to establish the account. And so my Yahoo! account had the information for the Google account. The Google account actually had the information for an Exchange account. And then the Exchange account led into a folder that could give you access to my password hash. Which I was looking at it, going, okay, that's convoluted, and I know where those are. But still, that's way more information than I thought I was giving away.

Steve: Well, and we're not good at saying "what if." As we've talked about, it's instructive to say, although a little dark, perhaps, to say, what if I die in the next five minutes? Literally, like, what if I die? Can my friends and family get to the things that they may need to? That is, planning ahead. Similarly, you challenge yourself by looking at your email berg and say, what if a bad guy got this? I mean, put yourself in that position. Browse through that. What if this was in the hands of someone malicious? What are the consequences? And again, just exactly as you were giving an example of, it's potentially frightening.

FR. ROBERT: And I think you're right, we just - we don't like to think about that. Or maybe it's not even that we don't like to think about that. We can't think of that. We just - we're not built to think of worst-case scenario. Go figure. Steve, is it just the standard mitigation practices that we're talking about here? We've got people in the chatroom saying, well, okay, if I change my password, if I use a strong password, is that enough? Or do I consider my accounts dead now?

Steve: I guess, okay, there are several ways to look at this. One is how do you feel about Yahoo!? There are alternatives to Yahoo!. So does their behavior inspire confidence

in their future custody of any of your email?

FR. ROBERT: And that answer should be no.

Steve: Yeah. It really should be no.

FR. ROBERT: If it took them two years to tell me about a massive breach that they've known about for at least that long, then no. I mean, I can't actually - there is no way to delete the accounts. You can empty them out, but they stay there, which is wonderful.

Steve: Yeah. So, okay. So to take those accounts then out of service, you empty them all out as much as you possibly can. Then you change their passwords to something insane that you deliberately don't write down, and you will never be able to get back in there again. And you're not going to have a problem with sharing your password. Basically you're saying, Yahoo!, no more. I mean, the reason LastPass still has my confidence is the way they respond to any problems found. That's the flipside of Yahoo!, where I'm proud to say I've never had an email account because it just always seemed a little too strange to me. I mean, I just never took them seriously. So, but I also had the advantage of my own domain and email server, so there wasn't much pressure on me to go looking for other solutions.

But still, yeah, I think you're right, Padre. I think that no one - maybe there's some reason you must keep your Yahoo! email. In that case, yes, certainly change the password. If you haven't logged in for a long time, knowing what little we know, which is distressingly little because they're just not being as forthcoming even now as they should be, logging in hopefully promotes you to a stronger hash. And at the same time then change it to a state-of-the-art, 20-character, mixed-character-class password, managed by a password manager, and you're doing everything you can. That and maybe just sweep out all of the debris that Yahoo! has been holding for you because other people can get to it, potentially.

FR. ROBERT: Yeah, we've got - the chatroom is starting up a micro flare here right now because of people saying, look, I use Flickr. And we've got PC Guy who is saying, "Yahoo's not just email, people." And I get that. But I think you're dead-on, Steve, which is every company will suffer a breach. If you're on the Internet, at some point, something will leak that you don't want to leak. The question is how do you respond to it? And if a company's willing to be 100% transparent, as soon as they know that there's a breach, to let us know, to let us know that we need to change our authentication credentials and to keep us up to date about what was taken and what was not, then I'm willing to stick with them. I'm willing to say, okay, yeah, that was a mess-up. But you've learned, and you're going to do better in the future. There's nothing about the way that Yahoo! handled this that makes me want to continue using their services because I'm just thinking...

Steve: And not even providing full disclosure.

FR. ROBERT: Precisely. Yup.

Steve: Even now not telling us what we need in order to make a more informed decision. That's what we would want, how to make the best decision. Well, we have to have information, and there's still, oh, well, you know, don't worry about the man behind the curtain.

FR. ROBERT: Because the man behind the curtain has your password. Right. Steve, it's

not all DDoSes and breaches. There's a little mischief going on in the latest version of iOS.

Steve: So, yeah. We've discussed ElcomSoft a number of times in the past. They're a relatively well-known security firm who sells iPhone-cracking commercial software. So a security researcher at ElcomSoft, Oleg Afonin, he discovered a flaw in the password hashing used to protect iOS 10's backups. And I would love to know what's really going on here, and I'll explain what makes me so curious. But what he discovered is that iOS 10's backups, that is, the brute-forcing of the password - we've just been talking about password-based key derivation function, where you deliberately use a technology to make them slow. He realized that, with moving from iOS 9 to iOS 10, it was 2500 times faster, which is to say easier, to brute-force a local iOS 10 backup than it had been under iOS 9.

Now, unfortunately, because of the nature of his commercial enterprise, he irresponsibly disclosed his discovery publicly, without first notifying Apple. They immediately did get in touch with him and said, "What? What?" And then he told them what he knew. But the cat was out of the bag first.

So here's what he wrote. And I'm quoting it because I don't want to paraphrase this because what he said is really interesting. He said: "When working on an iOS 10 update for ElcomSoft's Phone Breaker" - one of their products - "we discovered an alternative password verification mechanism added to iOS 10 backups. We looked into it and found out that the new mechanism skips certain security checks, allowing us to try passwords approximately 2,500 times faster compared to the old mechanism used in iOS 9 and previous."

FR. ROBERT: Wow.

Steve: "This new attack vector is specific to password-protected local backups produced by iOS 10 devices. The attack itself is only available for iOS 10 backups. Interestingly," he writes, "the 'new' password verification method exists in parallel" - this is why I would love to know what's really going on - "exists in parallel with the 'old' method, which continues to work with the same slow speed as before." So, okay. What was Apple thinking?

So continuing: "By exploiting the new password verification mechanism, we were able to support it in our latest update, ElcomSoft Phone Breaker 6.10. Since this is all too new, there is no GPU acceleration support (yet) for the new attack. However, even without GPU acceleration, the new method works 40 times faster compared to the old method with GPU acceleration. This," they write, "is extra-troublesome" - and here's another juicy nugget - "because decrypting a backup is currently the only way of cracking modern non-jailbroken phones. And even a jailbreak will not expose the Keychain's protections, but decrypting a backup will. So within the iPhone cracking/break-in community, backup encryption is the golden goose."

And then he provides in his blog posting a little table showing under iOS 9, CPU only, using an Intel i5, they can do 2,400 passwords per second brute force. Also iOS 9, but adding GPU acceleration, in this case an NVIDIA GTX 1080, that 2,400 jumps to 150,000 passwords per second. And now, okay, so iOS 9 with state-of-the-art GPU acceleration, 150,000 password guesses per second brute-forcing, cracking the backup password encryption.

FR. ROBERT: That's pretty good, 150,000 per second, yeah.

Steve: Now we move to iOS 10 with just CPU, that is, they've weakened it so much that, back to that Intel i5 that was only able to do 2,400 passwords per second on iOS 9, six million passwords per second.

FR. ROBERT: Holy...

Steve: Just with the i5. So if we assume this scales linearly, you know, look at the 2,400 to 150,000. What is that, about 70 to one? So that probably puts us at, what, 4.2 billion passwords per second. So pretty much the entire IP space of the Internet per second you could do brute-forcing under iOS 10 with GPU.

FR. ROBERT: Okay. So if I understand it, the original, the hard encryption, along with every guess, it was a little bit of work that your system had to do. It actually basically had to do some math. Are you saying that they took that part out in the backup? You no longer have to do the math? You can just brute-force the guesses?

Steve: Well, he's being deliberately obscure, and this hasn't been published because Apple's - he says, you know, he wants to sell his software as much as he can before Apple fixes this. And they're running around frantically in Cupertino right now, saying, "Oh, shoot," or worse. So all he says is certain security tests were bypassed. But what's curious is that what this says is, I mean, if we take what he wrote, and I'm sure it's accurate, exactly as written, that for some reason they're doing both. Now, their reasons for both, we were just talking about the notion, for example, of Yahoo! moving from an old hash to a new hash. So maybe they thought this was better, but they broke it. Or I don't know. But somehow the new one is far weaker.

So any iOS 10 backup is apparently dual-hashed. The password is double-hashed with the old algorithm. And the new algorithm, normally it's new and improved. In this case it's new and catastrophic. But the fact is you don't need to worry about the slow old one because you've got a much easier fast new one that you can use to do your brute-force attacking. And both of them are hashing the same backup password. So either one, the fastest one to finish gives you what you want. And in this case it's the new one which would finish much quicker.

FR. ROBERT: Wow.

Steve: It's like, again, there's a mystery here. We don't have the details. Maybe we'll get them someday when someone reverse-engineers this, figures out what's going on. And we know that Apple is not big on telling us in detail about their mistakes. They just say, "There's a new 10.0.3. Please download it immediately."

FR. ROBERT: To their credit, though, Apple is - they've acknowledged, like, okay, yeah, this was a mistake.

Steve: Yes.

FR. ROBERT: This is something we're going to fix.

Steve: They did immediately acknowledge it, yeah.

FR. ROBERT: Yeah. If it had been another company, they might say, well, we'll fix it in two years because we don't think anything's going to happen bad between now and then.

Steve: Well, and so what this does mean, too, is that any local backups made by this iOS

10, those need to be considered dangerous. That is, that backup needs to be destroyed, like maybe immediately, and then just refrain from backing up locally until you've got 10.0.3 or whichever update fixes it. I think we're at .2 right now. So, yeah, maybe .3. But you want to make sure that that backup is destroyed because it's a statically sitting, brute-forceable complete image of your phone which significantly also contains the Keychain. And if you crack the backup, you crack the Keychain, which even a jailbreak won't give you. And then you've got literally the keys to the kingdom. So any backups, any local backups made under iOS 10.2, well, 10.0 to 10.2, need to be considered a ticking timebomb. You want to make sure those don't persist.

FR. ROBERT: Right. And even after you update your software, you really should delete those old backups because they will still sit there.

Steve: Exactly.

FR. ROBERT: Especially if they get moved over to, say, a time machine. Yay. All right. More iPhone news because it's not just about really fast hashing here.

Steve: Well, so I love this one because the people covering the story picked up on the fact that Luca Todesco is a teenager. And so the headlines were how quickly he did this, apparently in less than a day. And so I could sort of imagine the conversation in the evening in his household, where his parents say, "Now, Luca, you can crack your new iPhone 7 after you've finished your homework." And he says, "But Mom, a new crack is worth up to $200,000." And his parents say, "Well, yeah, that's nice, dear. But geography is important, too. You need to know where Aleppo is if you're going to be President someday."

So anyway, the story here is that, within 24 hours of obtaining his new iPhone 7, a well-known teenage hacker who has successfully - he's got a reputation in the hacker community, which is why everyone believes him. He's not published details. He's just - he's achieved it. He says he wants to polish it and do a better job with handling all the details of the attack. There was a classic one years ago where you could simply go to a web page, and the attack was entirely supported by Safari. So a web server was able to deliver payload that would, just visiting that page, would jailbreak the phone. I don't know that this one can be reduced to that, but that was referred to in some of the coverage here, where he was saying, you know, "I just - I want to clean it up and polish it. Yes, I got my 7 jailbroken."

And what's interesting is that he wants a jailbroken phone, not to sell the exploit or to do evil, but because he is truly interested in poking around. He is a teenage security researcher. And it's very valuable for him to have root on his phone. In fact, in his little video that he produces he brings up a root console which he's then able to use to poke around. So he's annoyed that Apple won't give developers an official means for having that level of access to their phone. So he gets it for himself after dinner, and hopefully after doing his homework, and then plays with hacking his phone after school.

FR. ROBERT: That's not a bad hobby, a hobby that gets an extra $200,000.

Steve: Well, this is the kind of kid you want to hire.

FR. ROBERT: Yeah; right?

Steve: Absolutely. Somebody who's actually there, really reverse-engineering and digging in. I mean, I have no doubt that Luca knows his stuff. And I don't have any

doubt that he'll have no problem getting a job.

FR. ROBERT: Right, right. And there actually is a little bit of a tradition of doing the right thing that we've seen the last few years. You may remember Zimperium X. They're the ones who found the Stagefright bug.

Steve: Yup.

FR. ROBERT: And they could have sold that for seven figures. They really - in fact, they could still be selling it because it's the gift that keeps on giving. It's the exploit that will never be completely patched. But they disclosed it responsibly to Google for $1,337. And the only reason why they asked for that amount was so that it would spell out LEET.

Steve: LEET, of course.

FR. ROBERT: Yeah. So some people really just want to be curious and then do the right thing. And hopefully Luca's like that.

Steve: Yeah.

FR. ROBERT: All right. We've been hammering a lot on Apple. And I think we need to do some fair time here and talk a little bit about Microsoft because...

Steve: Well, yeah. And this isn't really a hammer. This actually surprised people. And I wanted to make sure that our listeners know. I meant to look because I think we're right on the cusp of - let me refresh the Never10 download count. Yes. Oh, my god, no: 1,999,138 downloads. I was thinking by podcast time we would have crossed two million. It's slowed down to only 3,666 downloads per day. So maybe by the end of the podcast we will have crossed two million downloads. But of course it's been phenomenally popular because a lot of people said no, thank you, for whatever reason, I want to stay where I am.

Somewhat controversially, I never fought with the GWX, the Get Windows 10, because, I mean, and I looked at doing so. And we've discussed this on the podcast in the past. There is no API, no means for a program to say to its own system, where it's running, please don't download this update. Because if you think about it, what would malware want to do? So of course any malware would want to hook onto an API that gave it control over updates to Windows that might raise Windows' awareness of that malware. So this is one area where Microsoft is adamant at minimizing programmatic control.

Well, so I hit that. I understood why there wasn't an API. And I said, okay, I can't prevent this Get Windows 10, which was being offered through the Windows Update, which was supposed to be security, controversially, but wasn't, as we all know. So all I did was I leveraged the existing registry options, which at least told Windows, the GWX, don't do anything. Just sit there and squat on some hard disk space, but don't even think about updating the system to Windows 10. I don't want it.

So the good news is Microsoft just released a formal remover. For anybody who wants to find it, there is a link in the show notes, or just look for, and here's the key number, it's 3184143. So it's KB (Knowledge Base) KB3184143. And they describe it as "This update removes the Get Windows 10 app and other software related to the Windows 10 free upgrade offer that expired on July 29, 2016. For a complete list of the software removed by this Windows Update, see the update replacement information." And on that page you'll find four links for Win7 and Win 8.1 in each of 32- and 64-bit system flavors.

And I just haven't had a chance, but I'm planning to post this on the Never10 page formally, just so that people who are, for whatever reason, wanting Never10 - actually, I've been told, because I've argued that it's not useful anymore, it's like, why are people still downloading it? Well, one of the things it does is it intelligently finds the one update that fixes the Windows Update update update update update problem, and it solves it for you. And so people are saying, yeah, we just use that now as the easier way to find that Windows Update update update update update. And, like, okay.

But anyway, so I will update the page after the podcast today with this link there because anyone who wants Never10 certainly would be interested in just running this and finally removing that lingering debris which otherwise isn't removed. And I don't know that this needs to be sought. Actually, when I turned on my Win7 machine which I use for Skype, I did see that there were some updates. I failed to see whether this one was being offered. So maybe this is automatic. People may want to go do it right now, when they hear this. Or maybe Microsoft will ultimately be just pushing this out in the channel, and the GWX will just dissipate from their systems on its own, which would be the right thing to do. So I imagine that's probably what they're doing. But at this point it looks like it's a go-get-it if you really are serious.

FR. ROBERT: By the way, Paul Thurrott did mention last week on Windows Weekly that you still can get the Windows 10 upgrade for free if you are on 7 or 8 or 8.1; that, even though the offer is expired, you can still do it. I just rolled back all my machines from 10. I had a few on 10.

Steve: I heard that yesterday. You were talking about this.

FR. ROBERT: Yeah. It was...

Steve: Or I guess on Sunday.

FR. ROBERT: I just, you know, it was one of these things where I was willing to put up with so many of the little quirks and foibles of Windows 10 because I actually do kind of like the OS. A few things like the sticky borders and the fact that the registry edits don't work anymore really bugged me. But what finally did it was the fact that I have active hours set on my machines so that it doesn't do any updates while I'm working. And evidently this last big update ignored that. Either Microsoft thought it was so important, or it was just not set up properly. So this machine went into its update - no notification, no ability to cancel it, just a notice saying "Do not turn off your machine" - in the middle of a show. It delayed the show by 30 minutes because this had all the tools that I needed to get to, and there was no way to stop it. There was no way to cancel it. And if you turn off a Windows machine in the middle of an upgrade, it will bork it.

Steve: Thank god you're not a heart surgeon, Father.

FR. ROBERT: I know; right? But then I got home, and I thought, okay, maybe I messed up active hours. Maybe that was me. And so I checked, before I started that night's video editing, I checked my active hours, and they were still on. Like, okay, I'm good. And in the middle of the video edit it did the same thing. It dropped into the update screen. I'm looking at it, I'm screaming at it, going, "I didn't even save any of my work." I lost an hour and a half of work because Windows just decided now is a good time for it to update. And I'm thinking, you know what, it might be a bug, and I'm willing to say Microsoft could fix it. But no. If your OS thinks that it's more important than the work I'm doing on the OS, we can't use you. I guess I'm a Never10 now.

Steve: Well, it'll stabilize. I don't have any doubt that it'll stabilize. We've seen this history of Microsoft going from good versions to bad versions. Normally, it's an every other one. I think they've got two turkeys in a row in this case. The problem, of course, is now they're saying there's not going to be an 11. We're just stuck with 10. But the other new thing about this is that they're moving towards the software as a service, and this now becomes an OS as a service because, one way or the other, if people are not going to upgrade, they're going to figure out a way to turn us into revenue streams. So I'm happy at 7.

FR. ROBERT: Okay. Let's propeller-head a little bit because OpenSSL has a little issue that I think our Security Now! folk need to know about.

Steve: So the problem with - okay. The problem with OpenSSL is it is old, and that's not good. And huge, and that's not good. And bloated, and that's not good. And the work product of countless, literally countless developers, that's not good. And incredibly difficult to maintain. The good news is, it is, for better or for worse, it is our industry's go-to de facto standard SSL/TLS and related privacy and encryption and authentication development and testing platform. That's where these technologies go to get their first life. And so it's sort of the galactic standard for everything, which again is a mixed blessing, which is why we're now seeing new packages. Amazon has one, for example, because of AWS and everything that they're doing. They just said, okay, we're not going to have that blob which is just - it's become unknowable on our servers. And the other problem is, you know, every single feature, every possible widget and gizmo and gadget which anybody had an itch to scratch added to the formal specification, it's in OpenSSL.

So the good news is, if you want it, it is there. The bad news is, if you don't want it, it is there. And we know what a problem complexity is. So this is yet another instance where, again, a mistake was found. It was immediately fixed. So again, the kind of response we want from an important mission-critical package like OpenSSL is what we got. What we're seeing, though, is that its role is changing. It's now the standard testing bench for new stuff where it makes sense to just graft another barnacle onto this thing in order to test the protocol.

But increasingly people are saying, eh, you know, we just don't need all of that for our web server, so we're going to do another one. And we're going to, by only implementing the features that actually ended up being in use - that's the other thing that OpenSSL represents is the entire exploration through history of Internet connection privacy and authentication. So obsolete stuff is there because everything is added in a forward and a backward compatible fashion. So stuff that nobody uses any longer is there. And I'm not saying it shouldn't be. We need one of those. I just don't think it ought to be used in production as much as it is. But it's the default de facto standard still for the Internet because maybe we'll need to turn that switch on, or use that.

So in this case I talked a couple years ago in detail about certificate revocation. One of the protocols that is really interesting, in fact, where I ended up coming down on the whole issue is stapling. OCSP stapling is the solution. The idea there is, okay, OCSP is the Online Certificate Status Protocol. And it provides a means by which a web browser can, when it receives certificates from a server, can, in real-time, on the spot, query the issuing authority for the validity of that certificate. So there are extra fields in the certificate providing the URL for that certificate's OCSP server.

The browser then gets it. And if you've turned that on - and we've talked about this. Firefox has supported it, and a lot of our listeners turned it on for a while. Google had some problems, surprisingly, with their own OCSP servers not being reliable enough. They just, they often didn't answer. And that was one of the controversies was, well, if

we don't get an affirmative yea or nay, what do we do? And for the sake of user experience and the fact that OCSP servers weren't at that time, and maybe even now, reliable enough, it would be a fail open. It would be, well, we couldn't get a "No, it's definitely bad." So we're going to go ahead and accept it.

Okay. Stapling solves this problem. It is a beautiful solution. I think it's where we're ultimately going to be. And that is that the server that is issuing the certificates also provides a recent, a sufficiently recent OCSP status reply with the certificate. So the website says, here's my identity and a recent revalidation of that assertion.

So the beauty is that changes the connectivity. Instead of there being a triangle where your browser goes to the server, gets the certificate, then your browser goes to the CA and gets that, instead now we're reusing the existing connection. And the server, the web server, notices that its OCSP validation is getting old, and what could that mean? That could be minutes or hours or days. But it's going to be a relatively short time because then it reaches out to the CA and updates and gets a new, timestamped, signed by the certificate authority, reassertion of its certificate's continued validity.

So that's the way the system works. What this means in terms of implementation is that a browser which is configured for it is able to explicitly ask the server to provide a stapled OCSP status response. And so what happened was, of course this all got implemented in OpenSSL. That's where it first started breathing, and it has since. Turns out there was a little mistake made in the code where - and I would call this a "classic edge case." So it's on the server side. A malicious client could leverage this edge case by causing, essentially - in fact, I'll read from the OpenSSL post: "A malicious client can send an excessively large OCSP Status Request extension." So that's an extension to the TLS protocol, some additional fields, that says I'm aware of OCSP stapling. Please send it back to me if you can.

"If that client continually requests renegotiation, sending a large OCSP Status Request extension each time, then there will be an unbounded memory growth on the server. This will eventually lead to a denial-of-service attack." Now, that's not a DDoS. That's a DoS. And we'll clarify that in a second. They continue, saying: "...through memory exhaustion. Servers with a default configuration are vulnerable, even if they do not support OCSP." So the fact that they're using that version of OpenSSL that offers the feature, even if they don't turn it on, they can still be brought down.

So there's three different tracks of OpenSSL. There's 1.1.0, which needs now to be updated - and that's, of course, 1.1 is the latest - needs to be updated to 1.1.0a. Then there's 1.0.1, and those users need to go to 1.0.1u. And 1.0.2, and those users need to go to 1.0.2i. I get nightly security reports from my Unix machines, and they immediately informed me that there was a new version of OpenSSL available. So it's been propagated. It was immediately - and it was a trivial thing to fix.

So basically it was a memory leak where it was discovered that a client could deliberately cause OpenSSL to request blocks of memory over and over and over, never freeing previous blocks, by both leveraging this flaw in the OCSP handling, coupled with renegotiation of the connection. So those two together sort of slipped by the original testers, and this got fixed. So essentially what it does is the system just keeps giving the memory to OpenSSL, which is running with tremendous privileges in the system so that it'll just - that process bloats and bloats and bloats and bloats until memory requests start failing for other legitimate processes, and the so-called denial of service in this case is that the web server, which has grown to the entire size of the server's memory, no longer has any memory available to serve as additional requests.

FR. ROBERT: The ever-expanding memory blob.

Steve: Exactly. Yeah. So it's not one of our horrible remote exploit, end-of-world meltdown problems. But not good. But immediately fixed.

FR. ROBERT: Right, right. Now, if you didn't do that, if you didn't always reserve that memory, you would run into Jelly Bean problems; right? I mean, eventually, because you'd start releasing memory that still contained sensitive bits.

Steve: Well, but we don't really know what's in there. For example, the OCSP, if it's just the OCSP response, that's public domain. So it's an assertion signed with the certificate authority's public key. So that's freely available. I mean, any browser can ask for one from a certificate authority. So if that's what is there, then it's just a mistake. But maybe the information contains something sensitive. But in this case it really doesn't matter. It's going to bring the server down. Somewhere in the renegotiation logic they forgot, when they allocated a new block of memory to service the renegotiated connection, there's just, like, one line of code missing, which was release the old connection's outstanding allocation.

So, you know, easy to do. Mistakes happen. And these guys did fix it as quickly as they could. And again, I want to make sure people understand. OpenSSL is the reference standard. But due to all of its history, it's no longer really becoming what you want to use in a production environment. You want to test new things there and then selectively move them maybe over to a much leaner armature that you'd then use for actual work, just because who knows what we haven't found? That's the thing. We continually find problems in it. And it should be no surprise to anyone. It's just too big. It's too old. It's too complicated. And we've talked about the alternative TLS stacks, which are 1/200th the size. I mean, it's hundreds of thousands of lines of code compared to 6,000. It's like, okay. If this 6,000-line system solution does everything we need, thank you. We're done.

FR. ROBERT: And speaking of done, let's move on a little bit because this next story is actually something that I'm very interested in because we've talked a lot about it over the last couple of weeks. And that is, what do you do when a CA misbehaves? Of course certificate authorities are the way that we learn how to trust on the Internet. And of late we've seen some CAs do some very, very peculiar, if not downright malicious behavior. Now, Steve, you know this. You know that trying to revoke the authority of a CA is very difficult. In fact, it's been almost impossible. It's one of the biggest reasons why people are saying we need to move over to DNS DANE so that we can start doing self-issuance of certificates and just forget the CA system altogether. But we've got sort of a good news/bad news thing with WoSign. Can you tell me about that?

Steve: Yeah. We talked about this in detail a couple of weeks ago. And the framing of our discussion then was I wanted to, and did, share Mozilla's thought process, essentially, with our listeners because, exactly as you say, there is big financial impact when a major browser like Firefox or Chrome or IE or Edge decides to pull the plug on trust with a certificate authority. Essentially, they're out of business because they're no longer - who's going to buy a cert from them when there's a competitive marketplace, and there are other certificate authorities which are trusted by all the browsers. So it's game over. So politically, from a bureaucratic standpoint, this has to be done carefully.

So a couple weeks ago we sort of went through phase one of that. And what popped up, and I caught this in a post by someone who follows me, Vincent Lynch, who's very much involved in and follows the certificate authority industry very closely, he summed it up, saying Mozilla now believes that StartCom - and remember there's also this weird WoSign/StartCom connection, where we discovered that changing a post parameter at

StartCom would cause it to issue a WoSign certificate. So it's like, uh, what? I mean, this is just all stinky.

So anyway, he said: "Mozilla now believes that StartCom purposefully backdated an SHA-1 certificate for a payment processing company," in flagrant violation of the CA browser industry, the so-called CAB Forum, rules. And so Vincent linked to today's update from Mozilla. And again, this is all out in the open, open for public comment. They don't want to hide anything because they recognized the consequences of them doing this are severe. But if we don't hold certificate authorities accountable, then the system really collapses.

So Mozilla wrote: "Today, Mozilla is publishing an additional document containing further research into the backdating of SHA-1 certificates, in violation of the CAB Forum Baseline Requirements, to avoid browser blocks." Meaning that they now have clear evidence that WoSign backdated certificates. Remember that there are "valid after" and "valid until" fields. And no browser today will accept an SHA-1 signed certificate that appears to have been issued after the start of this year. That is, with a valid after date in 2016. So that can cause problems. And we've talked about the problems that that can cause. So you can imagine that there would be pressure to issue a certificate with a previous date. And in this case it was December 20th, I think. I remember it was like 10 days before the end of the year this certificate appeared.

So Mozilla continues, saying: "It also contains" - that is, this write-up - "some conclusions we have drawn from the recent investigations and a proposal for discussion regarding the action that Mozilla's root program should take in response. Taking into account all of the issues listed above, Mozilla's CA team has lost confidence in the ability of WoSign/StartCom to faithfully and competently discharge the functions of a CA. Therefore, we propose that, starting on a date to be determined in the near future, Mozilla products will no longer trust newly issued certificates issued by either of these two CA brands. We plan to distrust only newly issued certificates to try and reduce the impact on web users, as both of these CA brands have substantial outstanding certificate corpuses." And should that be corpi? Anyway.

Okay. So once again we see this careful, methodical march to removing trust. And so the nice thing about, I mean, they're doing the right thing. They're not saying we're going to retroactively distrust. We're not going to yank the certificates, the roots out of our store. Rather, we're going to add some code to see, if the certificate is otherwise valid, when was the valid-after date? So essentially putting them out of business on Firefox, and other browsers will likely follow because, again, this is community connected so that they are no longer able to issue certificates in the future which will be trusted.

But that also protects the investments made by everyone who has an existing certificate. And what that means is that, when it's time to renew, they will go somewhere else. And Mozilla did say that maybe a year from now they will revisit this. But just to remind people, it wasn't just this flaky website and this one backdated certificate. When we covered this a couple weeks ago there were, like, six different separate isolated problems. Among them, many certificates had been misissued. And they didn't report that in their required auditing, and only revoked the ones that they were explicitly notified of they had to revoke, rather than, as they should have, going back through their own records and retrospectively revoking anything which they could determine had been misused and was subject to abuse. They didn't do any of that.

So, I mean, goodbye. I'm not going to miss them. And again, well, you know, websites can just go get the certificate from someone else. We have to hold certificate authorities accountable. And so we're seeing played out here in public the necessary bureaucratic

drama of creating a careful case and then ultimately saying, okay, we're sorry.

FR. ROBERT: You know, Steve, I'm with you. I think CAs absolutely need to be held accountable for what they do. The question I would have, though, is looking into the future, how do you do that? I mean, this is an isolated case. And this took a while. I mean, this company did a lot of really bad things before finally...

Steve: Yeah, over and over and over.

FR. ROBERT: Over and over. And they got little slaps on their - actually, not even slaps on the wrist, really, until finally there was enough oomph in the community saying, okay, we have to do something about this. But you can't do that for every CA that goes off the reservation. I mean, it just takes too long. And in the interim you've got this massive security hole because the entities you're supposed to trust are not trustworthy. And then I could see this evolving because we already know that there are, have been, a couple of CAs who have been in bed with nation-state level entities.

Steve: Yup. Yup.

FR. ROBERT: And I could see this becoming a cause to fracture the Internet, where you have nation-states saying, well, if you don't trust our CAs, we're not trusting your CAs. And, I mean, unfortunately, that sounds childish, but that's not unheard of. This sort of tit-for-tat happens all the time.

Steve: Yeah.

FR. ROBERT: So is the only way out of this to move to DNSSEC and DNS DANE?

Steve: Ultimately, and we talked about this last week, everyone knows what a fan I am of DNSSEC. And DANE is one of the many, many benefits that we will get once we have DNS secured throughout the entire system. I wouldn't - I don't begrudge anybody, any entrepreneur, the opportunity of starting up a certificate authority. But it's not as if the field is so rarefied that we need another one in order to, for example, bring the price down. It's not like we're in a monopoly situation where everyone's having to get theirs from VeriSign, for example. So it's like, I'm not going to miss them. And they probably have loyal followers who are going to miss them. But unfortunately the trust was misplaced. And we have a system - and I agree with you, Father. What you're basically saying is the system is based on trust, and it's not a strong enough assertion.

FR. ROBERT: Right.

Steve: Trust is not strong enough. And as you note, it's also not completely resilient in the face of fracturing, different types of fracturing of the Internet. But today it's what we have. And those guys can go away. Their customers will simply move to one of the other 400 certificate authorities and I'm sure get the same kind of price that they would have from these guys. So they had an opportunity to make some money. They did for a while. They weren't responsible. And because this is entirely resting on trust and behavior, I mean, that's the obligation that comes with basically printing money.

The certificate authorities perform a very valuable service. Everybody knows I'm a huge fan of DigiCert. They are my CA. I'm never moving away from them. I am so happy with the job they do. And they earn the money they're making. But they're selling bits. And it's like, wow, that's a great job if you can get it. And so unfortunately WoSign just, you know, they blew their opportunity of selling bits for dollars. It's like, okay, sorry. With

that printing money capability comes the obligation of printing it responsibly.

FR. ROBERT: They killed the cow to have a steak, basically. And actually I'd go one step further. It's not just that the system is built on trust that may not be there. Unfortunately for most of the world, the CA system is built on ignorance of the process.

Steve: Yes. Yes.

FR. ROBERT: You say "CA" to most people who don't watch Security Now! or the TWiT.tv network, and they'll just sort of glaze over and say, uh, I have a padlock in my browser. That's okay; right? And that's it. That's the extent of their knowledge.

Steve: I will never forget the podcast, I don't know which number it was, but we're at 579 today. But it was many years ago when, between podcasts, I had for some reason looked into the CA root in a Windows machine. And I remember when there were 12 certificates...

FR. ROBERT: Remember that.

Steve: ...in there. I mean, VeriSign was there. Global Trust was there. And a couple other companies that sort of, I didn't know they were in the CA business. But, I mean, you didn't have many more fingers than there were certificates in a Windows machine. And one week, between podcasts, I looked in there, and the scroll thumb went [sound effect] down into this little bitty thing. And I thought, what? And I started dragging it down, and hundreds of trusted roots go by. And with the next podcast I said, "Oh, my god, Leo, what has happened?" And then we did a podcast talking about the consequence of this explosion of certificate authorities, that it is a Trust Everyone-based model. And unfortunately, the more everyones you have, any single point of failure cripples the entire system. So again, the only response to that is a zero-tolerance policy.

FR. ROBERT: Zero tolerance and zero trust. Is there another way to say zero trust? Huh. Trust very few people? Trust just a few? Maybe not. We'll figure it out later on. All right. We do need to push on because we're not going to get to any Q&A in this Q&A.

Steve: No, we're not. We will do that tomorrow for next week's podcast.

FR. ROBERT: But we still do have something funnier. We get to talk about BitTorrent Sync. And this is one of the services that many members of the TWiT family were asking us to cover on Know How. And it's interesting. I used it once or twice. It wasn't really my flavor anymore. It was renamed. Does that also mean that it's a new being?

Steve: Okay. So, yeah, they renamed it Resilio, I guess as in resilience, Resilio, R-E-S-I-L-I-O. So Resilio is the new name for BitTorrent Sync. And from the first day of announcement, our listeners have said, "Oh, my god, Steve, what is it? Give us an analysis, like you do when you have the information, so you can explain it to us and tell us we can use it because it looks wonderful."

And so I immediately got in contact with the BitTorrent PR guy. And I said, "Hey, look, I'm not buying anything. I know who BitTorrent is. What I want is the whitepaper for the documentation." And instead he said, "Oh, you know, we just put a brand new monument in the front of our building, and it's four inches marble, and it's a beautiful italic font." And I said, "No, no, no, no, no. What is the protocol?" "Oh, well, you know, we've got this great team of blah blah blah blah." I mean, and finally I told him, "Stop sending me your press releases. All I want is the technology." Never got it. They

published a non-whitepaper whitepaper about a year ago. Still nothing.

Now they've done it again, which is what got it onto the show notes. There's a whitepaper. Actually it's not white. It's fancy. It's got marble italic font, just like the monument in front of the building. They call it "Resilio Sync Security & Privacy Brief." Notice it doesn't say "privacy specification." The good news is it's not too brief. So there is a huge demand for this podcast to say something about what used to be BitTorrent Sync and is now BitTorrent Resilio. I think there's enough there for me to at least discuss what they're willing to say.

Again, that falls far short of, okay, here are the protocols. For example, Telegram could say everything that these guys have said. And from that it would look wonderful. But when I actually saw the block diagram of the Telegram architecture, I said, "Holy crap, this is the biggest crock I have ever seen." And not long afterwards the rest of the security community agreed. And so the problem is, as we know, the devil is in the details. And for whatever reason, they're not publishing it. And the only reason I can imagine is competitive. I don't think they think there's anything wrong. But unfortunately, those who create it are not those who need to judge it. And I imagine they don't want competition. They don't want the compatible products to be created.

There was some effort a couple years ago at reverse-engineering it. There's something called the Initial Protocol Specification, actually hosted in a forum on their site, dated summer, July of 2013, so three years back, where someone took some time to dig around and work on it. I sort of thought more than that existed today, but I thought there was a working BitTorrent Sync-compatible implementation. But just in looking briefly, I didn't see it anywhere. But so I just wanted to put it on people's radar. I will find some time, in a future podcast, before long, to dig into it. And we will finally do a podcast, not on BitTorrent Sync because we waited long enough for it to become BitTorrent Resilio. And I'll share what there is to share.

But I'm still annoyed because - and as you said, Padre, it's closed, and not your cup of tea. Not mine, either. There's something called Syncthing which a lot of people seem to be liking. And I believe that's completely open. And that's, I mean, that's what you want. So if these guys got competition from a knowable open alternative, I can't think of anybody who would deserve it more.

FR. ROBERT: And the thing is not just that it's closed, it's that what they've opened, they've let us take tiny little peeks into how this handles encryption. It's not impressive.

Steve: No.

FR. ROBERT: It's not.

Steve: Well, that's just - it's the PR guy. Oh, yeah, well, you know, we've got military-grade encryption and John McAfee said it's wonderful. Okay.

FR. ROBERT: Let's see. John McAfee was the man who invited me to come to a strip club for Black Hat at Vegas.

Steve: How well does he know you, Padre?

FR. ROBERT: Well, no. This was an invitation to all the journalists. And who went? Ian, Iain Thomson actually went.

Steve: Of course he did.

FR. ROBERT: And he was asking me if I went. I'm like, are you kidding me? John McAfee asked me to come to a strip club. I'm like, okay, this is like a red flag had a baby with warning tape. Just no, you stay far, far away from that. And it turned out, yeah, it was really that bad. It was a really skeevy club. I guess they're all kind of skeevy. And he showed up, like, two hours late and then didn't want to talk about anything. So, yay.

Steve: Well, and you know, Showtime has a documentary, produced it. My TiVo sucked it in, and I keep forgetting that it's there. But it was called, oh, shoot, it had a short name, and then "The Dangerous Life of John McAfee." And so if anyone - I did tweet about it. I think it aired for the first time on Saturday night on Showtime. And so for what it's worth, I'm glad it came up because, if anyone's curious, it's a two-hour documentary. John has said it's all lies, full of lies. So it sounds like it was probably going to be fun.

FR. ROBERT: I mean, yeah, love him or hate him, he's a very interesting person.

Steve: Yes. He's entertaining.

FR. ROBERT: He is entertaining.

Steve: I did want to correct the record, just for the sake of accuracy. I misspoke last week when we were talking about OSes and routers by saying that pfSense and a Ubiquiti router were both FreeBSD based. I know better. Ubiquiti uses Debian. We've talked about that in the past. Someone said, "Uh, Steve, no." And it's like, okay, you're right, sorry about that. So I just wanted to close that dangling mistake.

FR. ROBERT: Naturally, naturally.

Steve: Two bits of miscellany. I did want to put also on people's radar a science fiction program on NBC beginning week after next, on October 3rd, called "Timeless." Time travel. I don't know much. I'm not recommending it. But for anybody, I mean, I'm going to record it and hope that it's engaging and fun. Time travel is neat stuff. I think the bad guy is the actor who played Luka on "E.R." So, yeah, I'm dating myself. But it just looked, you know, it's going to be made - it's a network show. I'm sure it's not going to be deep sci-fi. But it looks fun. Apparently the plot is that a state-of-the-art time machine is stolen by this bad actor with the intent of going back in time and messing up the past. So, but there's a prototype of it also. And so the good guys use that and go chase them around. So anyway, for what it's worth, it's called "Timeless." I'm not vouching for it, but I wanted to make sure people knew about it. Looks like some fun special effects and an interesting cast.

FR. ROBERT: Yeah. I want another time travel series because we haven't had many. I mean, "Heroes" kind of had some time travel in it. But before that it was "Seven Days," which I really enjoyed. I was sad when they canceled it. And before that there was "Timecop," which was what time travel would be if everything was really cheesy and hokey.

Steve: Oh, that was a classic movie. I've seen it about three or four times.

FR. ROBERT: Oh, but they turned it into a series.

Steve: Yeah. And we all remember when Spock and Kirk and McCoy went back in time.

One of my favorite, you know, and we met Edith Keeler, who had her soup kitchen.

FR. ROBERT: That was "City on the Edge of Forever"?

Steve: Yes.

FR. ROBERT: That episode?

Steve: Yeah. And there was that time gate, that weird sort of organic-looking thing that - yeah.

FR. ROBERT: And it was McCoy who went through. And immediately as he went through they lost contact with the Enterprise because the Enterprise no longer existed, so they had to go back and fix the timeline.

Steve: Don't you hate when that happens?

FR. ROBERT: I mean, seriously. I can't tell you how many times that's happened.

Steve: Yeah. And actually he mistakenly OD'd on cordrazine or something. So he went out of his mind. He went crazy. He jumped through the portal. And so then Spock and Kirk had to go back and rescue him. Anyway, I'm sure everybody already knows about that particular episode. It's one of the best.

FR. ROBERT: Oh, by the way, the chatroom just called me out. I just realized one of my favorite shows, "Dr. Who," is a time travel series. So, yes, we do have one currently. My bad. Totally my bad. Sorry about that.

Steve: And Padre, I meant to ask you this before, whether by any chance you're watching "The Strain."

FR. ROBERT: I am not.

Steve: It's now in its third season. It's on FX. And I really like it. As they say, there's no accounting for taste. I'm not suggesting that everyone is going to love it. But it holds up. They've built a complete coherent mythology essentially around the vampire myth, but brought it into the present day with some useful acting, some fun writing. And it's in its third season, and it's really good. So I just...

FR. ROBERT: Oh, and Guillermo Del Toro. Okay, all right. No, I'm in, I'm in. I'm totally in.

Steve: So the first two seasons are available on Hulu and, hint, elsewhere. I don't think anybody - now, again, it's not going to be everyone's cup of tea. I get it. But if you don't mind a little gore, but something really engaging, it's just, you know, a plane lands at the airport, and no one gets out. And they wonder what's going on inside. And the CDC is brought in. So it's state of the art. And I've noticed in this third season they recognize they have something good. And so now we're getting - the writers are writing back in some of the older back story, which is equally good. I've just - I'm impressed.

So FX has "The Americans," that I also really enjoy. And it's going to start its fourth or fifth season. And I only know about "The Strain" because they were advertising it during the breaks of "The Americans." So I just wanted to put it, again, on our listeners' radar.

FR. ROBERT: This might be something that I have to get onto my computer because I'll be taking a trip over to Rome. And I'm actually responsible for some of the entertainment nights.

Steve: Ohhh.

FR. ROBERT: Maybe we can be in the Eternal City watching "The Strain." I'm sure that will go over well.

Steve: Give it a try. I would love to know what you think because it's - I really enjoy it. I mean, it's just - it's well done. And I like things that are well done. And, oh, my god, that's an awkward segue to start talking about SpinRite. But Anthony Cunningham sent me a long DM. And he said: "Hi, Steve. As I listened to last week's Security Now!, Father Robert made a comment that I thought I might be able to help with, plus it would work as a SpinRite story.

"Close to the end of the podcast, he mentioned he wished you could get SpinRite to work on cell phones. And I would like to note that that request might be possible already. A previous SpinRite testimonial talked about using it on a virtual machine to do multiple runs at the same time. At the time that got me to try it. And what do you know, it worked. I also used it on a live Linux USB distro that was at the time not working right, and it fixed that as well. Recently the cell phone I use at work as my podcast streaming player was getting rather slow and laggy, so I thought, 'What the heck, I'll try SpinRite on it and see what happens.'

"Using VirtualBox on Linux, I plugged in the phone to the USB, had VirtualBox set it as a raw disk, spun up a VM with SpinRite, and pointed it at the cell phone. SpinRite did its thing, and about an hour later it was done. I unplugged the phone, rebooted it, and after that it was running faster and more stable than it had for over a year. I thought to myself, this was great. I'll try it on my Nexus 5 next. But that, unfortunately, was a no go. The phone must be able to be seen as a mass storage device in order to be used as a raw disk in VirtualBox, and for some reason the Nexus 5 will only read as a non-mass storage USB device. I hope Father Robert can try this out and see if it helps."

So, Anthony, thanks for the great DM and another example of how SpinRite can be leveraged, in this instance in a VirtualBox, in order to run on a cell phone and fix it.

FR. ROBERT: That is amazing. I am trying that tonight on my OnePlus One because I know exactly what's going on. Every once in a while it will just kind of hang. And it's just the garbage collection has not been done properly over the course of the lifetime of this device.

Steve: Level 2.

FR. ROBERT: Yup, Level 2. All right. That's my task for tonight. Actually I built a SpinRite-only machine a while back for Know How. We never did the episode. But it's sitting there. It's like one of my break-in-case-of-emergency projects. I've got six or seven in case I'm just really lazy one week, I can just [crosstalk].

Steve: Queued up, right.

FR. ROBERT: But I made a tiny little machine that has just a hard drive docking station. And it's got all the different formats.

Steve: Nice.

FR. ROBERT: All the way from mSATA to the M.2 to SATA to the large version. And so you just plug in the one you want, turn it on, and then walk away.

Steve: Nice.

FR. ROBERT: So I'll try that with this. Who knows? Maybe it will fix my OnePlus.

Steve: Cool.

FR. ROBERT: Steve.

Steve: That's the podcast.

FR. ROBERT: That is the podcast. You know what, it is absolutely a pleasure. I am so sorry to everyone who was hoping to hear the Q&A. We will get to it next week, which is actually tomorrow.

Steve: Hey, what we promise is two hours of engaging content, whether it's sourced from our listeners or from the events of the week. And in this case, now, as our listeners may or may not know, you're traveling, as you just mentioned, to Rome next week. So you and I are recording next week's podcast tomorrow evening. Which means there won't be a week's worth of news aggregated. So I think it's very safe to say that next week's podcast, actually recorded not more than 24 hours from now, will be pretty much all user sourced content, and nothing that the industry has brought to us, because there won't have been any time. As I said, we don't have a time machine. So we'll have to work with what we've got.

FR. ROBERT: Steve, I think that's the only way we're ever going to get to Q&A because any time I work with you, I just - I have way too much - I know, sometimes I know I'm like, you know what, just let Steve do it. Let Steve do it. And then I just want to get in there. I just want to have so much fun. Thank you for the knowledge you drop on us each and every single week.

Steve: Well, this was great, and it's a pleasure working with you. And we'll do it in a little over a day for next week's Security Now!. And if anybody is interested, if the change in schedule allows you to watch it live, we're recording at 7:30 Pacific time, so 10:30 p.m. Eastern, on Wednesday, that is, tomorrow. And so if that allows you to watch it live, what is that, is it still live.twit.tv?

FR. ROBERT: It is indeed.

Steve: Cool.

FR. ROBERT: Of course, you know Steve Gibson. He is the mastermind behind GRC.com, the creator of treasures, gems like ShieldsUP!, SpinRite, and of course SQRL. And he's just - he's our Explainer in Chief. He is the man that everyone at TWiT turns to for the latest and the greatest in security. And, well, we couldn't do it without him. Steve has audio versions and transcripts of the show at his site at GRC.com. And you can also find the show wherever fine podcasts are aggregated, including at our show page at TWiT.tv/sn for Security Now!.

We do Security Now! every Tuesday at 13:00 Pacific, actually it's 13:30. You can watch

live at live.twit.tv or download episodes from TWiT.tv/sn. You can also subscribe to the show to get it automatically downloaded to your device of choice in your format of choice, as well as downloading directly from the page. We want to give you every chance possible to get your dosage of security goodness. Until next time, I am Father Robert Ballecer, the Digital Jesuit, in for Leo Laporte, saying go be secure.

Steve: Thanks, Padre. Talk to you soon.