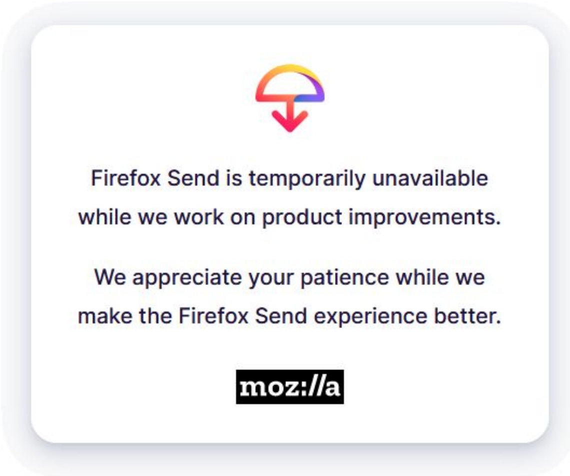
Security Now! #775 - 07-14-20

Tsunami

This week on Security Now!

This week we look at Mozilla's surprise suspension of their Firefox “Send” service, Zoom's latest remote code exploit vulnerability, the latest revision of the US congress' EARN IT Act legislation, the growing tension with “Stalkerware” apps, a Chinese Internet equipment vendor in the hot seat, the challenge of geo-locating illegal drone operators, Fraunhofer's report of rampant router vulnerabilities, SpinRite's move toward increased political correctness, and then we wrap up by looking at “Tsunami”, Google's latest and extremely useful looking contribution to the open source community.

Security News



Mozilla suspends “Send” due to persistent malware abuse.
Mozilla's “Send” service has become my favored solution for sharing files. As we know, files of up to 1 or 2.5 gigabytes (depending upon whether you're signed in to Mozilla) are locally encrypted in the browser and may optionally be password protected so that only the recipient is able to retrieve and decrypt the sent file. And retention controls allow the sender to set time and/or download count before the shared content expires and is removed from Mozilla's cloud servers.

Unfortunately, as with anything that is simple, free and effective (like eMail) it is also subject to abuse by nefarious forces. The bad guys also love Firefox Send because it lets them generate short-term links based on good-looking trusted domains for sharing arbitrary files to unwitting victims.

Thanks to Firefox Send, the bad guys don’t need to set up a file sharing server of their own at a legitimate-looking URL, and they don’t have to worry about making sure their URLs expire automatically after use. Mozilla has been doing that for them. And links that only work once create an extra challenge for security researchers because even if the malicious URL is captured in a log, it's not possible to go back and obtain the original since it's been expired. And since the IP of the server is Mozilla's, it's not one that anyone wants to block. (I was going to say “blacklist” but I'm working to be better.)

Over the past few months, Firefox Send has been used to store payloads for all sorts of cybercrime operations, from ransomware to financial crime, and from banking trojans to spyware used to target human rights defenders. FIN7, REVil (Sodinokibi), Ursnif (Dreambot), and Zloader are just some of the malware gangs and strains that have been seen hosting payloads on Firefox Send.

As a consequence, the cybersecurity industry has tipped its collective hat to Mozilla for suspending the now widely abused service rather than generate the typical vague promises about “seeing about changes” in the future. Some cybersecurity researchers have suggested changes to strengthen the service, such as adding a “Report Abuse” button to make flagging or killing malicious links quick and easy. Mozilla's issued statement reads:

“Before relaunching, we will be adding an abuse reporting mechanism to augment the existing Feedback form, and we will require all users wishing to share content using Firefox Send to sign in with a Firefox Account.”

It's sad that once again we see the Internet's inherent anonymity being abused and then restricted. It was cool to be able to send up to 1GB with a 24 hour expiration without needing an account with Mozilla, even though I have one. But just as Zoom was forced to limit what could be done with full anonymity, so, too, has Mozilla. As we know, even requiring an account is not a very high bar. And it likely won't be very effective. But perhaps it will help a bit.

In light of all this, of everything we see going on around us. I will make a prediction that in some future, access to some set of abuse-prone services, perhaps anything that generates rather than consumes content, will require some form of affirmative government-issued verifiable identity. In that possible — and I would argue probably eventual — future, it will be permissible to browse and read and consume content anonymously. But any public content will be traceable back to its source. We're not there today, and it won't happen soon. But I'll bet that it happens eventually. Perhaps as a half-step, content will be flagged as from a verified source of not, sort of like a Twitter identity. Then users can choose what level of veracity and/or risk they wish to accept.

And speaking of Zoom...

Zoom fixed a new RCE affecting Windows 7 and earlier systems.

This was a bit of an odd chain of events. A private researcher who wishes to remain anonymous (and still is) reported his discovery of a previously unknown remote code execution vulnerability in the Zoom client, which affects versions of Windows up to and including 7 and likely its matching Windows Server 2008 R2, not directly to Zoom, but for some reason to Acros Security, the zero-patch (0patch) guys.

Once 0patch had confirmed and reproduced the problem, including creating a proof of concept and a demo video, they fully, privately and responsibly disclosed the problem to Zoom.

Last Thursday, on July 9th, 0patch blogged about the discovery...

https://blog.0patch.com/2020/07/remote-code-execution-vulnerability-in.html

Earlier this week a security researcher shared a remote code execution "0day" vulnerability in Zoom Client for Windows with our team. The vulnerability allows a remote attacker to execute arbitrary code on a victim's computer where Zoom Client for Windows (any currently supported version) is installed by getting the user to perform some typical action such as opening a document file. No security warning is shown to the user in the course of attack.

The researcher (who wants to keep their identity private) stated that they did not report the vulnerability to Zoom either directly or through a broker, but would not object to us reporting it to Zoom.

We analyzed the issue and determined it to be only exploitable on Windows 7 and older

Windows systems. While Microsoft's official support for Windows 7 ended this January, there are still millions of home and corporate users prolonging its useful service life with Microsoft's Extended Security Updates or with 0patch.

We documented the issue along with several attack scenarios, and reported it to Zoom earlier today along with a working proof of concept and recommendations for fixing. Should a bug bounty be awarded by Zoom, it shall be waived in favor of a charity of researcher's choice.

On the micropatching side, we were able to quickly create a micropatch that removes the vulnerability in four different places in the code. The micropatch was then ported from the latest version of Zoom Client for Windows (5.1.2) to previous five versions back to 5.0.3 released on May 17, 2020. Zoom Client features a fairly persistent auto-update functionality that is likely to keep home users updated unless they really don't want to be. However, enterprise admins often like to keep control of updates and may stay a couple of versions behind, especially if no security bugs were fixed in the latest versions (which is currently the case).

Our micropatches have already been released and distributed to all online 0patch Agents; Zoom users with 0patch installed are therefore no longer affected by this issue.

According to our guidelines, we're providing these micropatches to everyone for free until Zoom has fixed the issue or made a decision not to fix it. To minimize the risk of exploitation on systems without 0patch, we're not publishing details on this vulnerability until Zoom has fixed the issue, or made a decision not to fix it, or until such details have become public knowledge in any way.

And in an update to the blog, yesterday (Monday) they amended the posting to add:

[Update 7/13/2020: Zoom only took one (!) day to issue a new version of Client for Windows that fixes this vulnerability, which is remarkable. We have reviewed their fix and can confirm that it efficiently resolves the vulnerability. With an official vendor fix available to all users, we made our micropatches for this issue PRO-only according to our guidelines. Meanwhile, after issuing micropatches for this issue targeted at Zoom Client for Windows versions 5.0.3 to 5.1.2, we noticed a lot of our users being on all of these versions despite Zoom's highly persistent update mechanism. We had expected most users to be on version 5.1.2., but this indicates many users may still be on even older Zoom Client versions. We therefore ported our micropatch to the remaining supported versions of Zoom Client: 5.0.0., 5.0.1, and 5.0.2. We're now covering all vulnerable supported clients.]

So as an FYI: Our listeners will want to be sure that their Zoom clients are v5.1.2 or later. Since Zoom has an aggressive auto-update facility in place, it's unclear why or how many users' Zoom clients are lagging behind. It's unlikely that this would ever be used in anything other than a targeted attack, but no one wants to be its victim.

The Zero-Patch guys have a bunch of Q&A on their blog posting about this vulnerability, its exploitation and its resolution. So if you're curious, check out the link in the show notes.
I'll also note as an industry we appear to be suffering a bit of definition drift. A "0-day" is defined

as a vulnerability that is first discovered as a result of its being actively exploited in the wild. It's considered of the highest possible importance and priority because it is already "loose" when it is first seen. But I'm seeing the tech press, and the 0-patch guys themselves, describing this Zoom RCE as a "0-day", despite the fact that, so far as we know, it has **never** been abused even once. That's not a 0-day in any way. It's just the discovery, responsible reporting, and remediation of a previously unknown vulnerability. In the best case that's happening all the time. That's what the HackerOne folks have made their business model. Since the term "0-day" has a very specific and important meaning, I think we need to work not to overzealously label things for the sake of a flashy headline, or the audience for this is going to go numb and not appreciate real problems when they do occur.

The EARN IT bill, take II

Recall that EARN IT is the rather tortured acronym for: "Eliminating Abusive and Rampant Neglect of Interactive Technologies Act of 2020." It has just been revised in an attempt to collect the votes necessary to get it passed through Congress.

I struggled through a bunch of legalese and mumbo-jumbo in an attempt to distill what has changed about the bill. As near as I can see, what the bill has changed is definitely not for the better. The original bill employed a 19-person federal commission, predominantly seated with law enforcement, who had the power to determine whether this or that website was "in compliance" (boy, the phrase "in compliance" it chilling) with the committee's "best practices" which, if they were, would entitle them to ***retain*** the legal protections all websites currently enjoy and depend upon under Section 230 for hosting user content.

But that was then. Now, the 19-person federal commission has been abandoned with the new bill handing over that power to the 50 individual states. (And I'll just note that we've seen how well allowing each state to determine its own course of action has worked with what has become the COVID-19 mess.)

Under this revised bill, should EARN IT ever become law, individual state lawmakers will be able to create new laws allowing private lawsuits and criminal prosecutions against Internet platforms, as long as they say their purpose is to stop crimes against children.

The EFF summarizes the intent of Section 230 this way:

> "The whole idea behind Section 230 is to make sure that you are responsible for your own speech online—not someone else's. Currently, if a state prosecutor wants to bring a criminal case related to something said or done online, or a private lawyer wants to sue, in nearly all cases, the prosecutor has to seek out the actual speaker. They can't just haul a website owner into court because of their user's actions. But that will change if EARN IT passes.
>
> Section 230 protections enable the Internet as we know it. Despite the politicized attacks on Section 230 from both left and right, the law actually works fine. It's not a shield for Big Tech—it's a shield for everyone who hosts online conversations. It protects small messaging and email services, and every blog's comments section.
> Once websites lose Section 230 protections, they'll take drastic measures to mitigate their

legal exposure. That will limit free speech across the Internet. They'll shut down forums and comment sections, and cave to bogus claims that particular users are violating the rules, without doing a proper investigation. We've seen false accusations succeed in silencing users time and again in the copyright space, and even used to harass innocent users. If EARN IT passes, the range of possibilities for false accusations and censorship will expand."

The EFF added...

"When we say the original EARN IT was a threat to encryption, we're not guessing. We know that a commission controlled by Attorney General William Barr will try to ban encryption, because Barr has said many times that he thinks encrypted services should be compelled to create backdoors for police. The Manager's Amendment, approved by the Committee today, doesn't eliminate this problem. It just empowers over 50 jurisdictions to follow Barr's lead in banning encryption."

With the amended bill, it will only take one state to inspire a wave of prosecutions and lawsuits against online platforms. And just as some federal law enforcement agencies have declared they're opposed to encryption, so have some state and local police.

The previous version of the bill suggested that if online platforms want to keep their Section 230 immunity, they would need to "earn it," by following the dictates of an unelected government commission. But the new text doesn't even give them a chance. The bill's sponsors simply dropped the "earn" from EARN IT. Website owners—especially those that enable encryption—will no longer be able to "earn" their immunity from liability for user content under the new bill. They'll have to defend themselves in court, as soon as a single state prosecutor, or even just a lawyer in private practice, decides that offering end-to-end encryption was a sign of indifference towards crimes against children.

It turns out that "Stalkerware" is a thing

"Stalkerware" has grown to become enough of a thing that Google, Apple, A/V makers and even the FTC have been pushing back on this growing application segment.

The term "Stalkerware" refers to spyware apps designed to allow an abusive partner in a relationship to spy on their significant other by installing the spyware onto the others' smartphone without their knowledge or consent. Sadly, "stalkerware" is also sometimes referred to as spouseware.

It turns out that its use has steadily increased through the last decade, largely enabled by the proliferation of smartphones. It allows partners to keep tabs on their partners at all times by tracking their phone. And the ready availability of these stalkerware products in official app stores has increased the visibility of these products, opening them to millions of potential users.

According to statistics gathered by Kaspersky, the number of users who had stalkerware-like apps installed on their Android devices increased from 40,386 devices detected in 2018 to more than 67,500 one year later in 2019. So, it's not a massive market, but it is a slimy one. According to independent antivirus testing lab AV-Comparatives and the EFF, detections rates for stalkerware applications on Android and Windows devices have slowly improved, as the issue is gaining more press coverage and security vendors are moving in to address their growing risk. We've briefly touched on this in the past... since these apps are not technically malware, they sort of fall into a gray zone between clearly benign and beneficial apps and clearl malware.

https://support.google.com/adspolicy/answer/9726908?hl=en&ref_topic=29265

Against this backdrop, we have Google's announcement last week of an update to their "Enabling Dishonest Behavior" policy:

> In August 2020, the Google Ads Enabling Dishonest Behavior policy will be updated to clarify restrictions on advertising for spyware and surveillance technology. The updated policy will prohibit the promotion of products or services that are marketed or targeted with the express purpose of tracking or monitoring another person or their activities without their authorization. This policy will apply globally and we will begin enforcing this policy update on August 11, 2020.
>
> Spyware and technology used for intimate partner surveillance including but not limited to spyware/malware that can be used to monitor texts, phone calls, or browsing history; GPS trackers specifically marketed to spy or track someone without their consent; promotion of surveillance equipment (cameras, audio recorders, dash cams, nanny cams) marketed with the express purpose of spying.
>
> This does not include (a) private investigation services or (b) products or services designed for parents to track or monitor their underage children.
>
> Violations of this policy will not lead to immediate account suspension without prior warning. A warning will be issued, at least 7 days, prior to any suspension of your account.
>
> Please review this policy update to determine whether or not any of your ads fall in scope of the policy, and if so, remove those ads before August 11, 2020.

A Chinese Internet equipment vendor in the hot seat
I don't know what to think about Internet appliances from China. It's not as if it's not just as possible for domestic vendors to deliberately plant backdoors in their products. As we know, the security industry remains quite suspicious of our own American NSA and CIA. We have quite strong circumstantial evidence that both organizations are developers of powerful computer subversion and surveillance technology. And to this day we suspect that the NSA may have influenced the design of the default source of entropy used in RSA's earlier BSAFE crypto API... and not in a way that made it stronger.

So it's certainly unfair to paint all of China with a broad red brush when we discover that a Chinese vendor is apparently shipping very high-end networking equipment containing multiple, apparently deliberate, backdoors. Without drawing any conclusion or rendering any judgement, here's the story of one such vendor:
Last week, security researchers Pierre Kim and Alexandre Torres very clearly documented their

discovery of seven vulnerabilities in the firmware of FTTH OLT devices, manufactured by the Chinese equipment vendor C-Data.

https://pierrekim.github.io/blog/2020-07-07-cdata-olt-0day-vulnerabilities.html

“FTTH” stands for Fiber-To-The-Home and “OLT” stands for Optical Line Termination. So taken together, FTTH OLT refers to networking equipment that allows ISPs to bring fiber optic cables as close to the end-users as possible — transiting the so-called “last mile” to our doorstep. These devices are the termination on a fiber optics network. They convert data from an optical line into an Ethernet cable connection that's plugged in at an end user's home, in data centers, or business centers. And these devices appear all over an ISP's network due to their crucial role. This makes them one of today's most widespread types of networking devices, as they are situated in millions of network termination endpoints all over the globe.

So, Pierre and Alexandre confirmed the vulnerabilities by performing a static analysis of the latest firmware running on two devices, but due to the common internal architecture shared by the entire similar family of devices, they believe that the same vulnerabilities impact 27 other of the company’s FTTH OLT models which run the same or similar firmware. Their full report is up on Github and I have the link in the show notes for anyone who wants more.

The seven vulnerabilities are as bad as it gets, but by far, the worst and most disturbing of the seven is the presence of Telnet backdoor accounts hardcoded into the firmware. That's not a bug... it's a feature.

And yes, it keeps getting worse: The Telnet accounts are on the WAN-side interface. They allow those who know the hardcoded credentials to connect to the device using any Telnet server running on the device's WAN (Internet-facing) interface. Pierre and Alexandre said that the accounts granted intruders full administrator command-line interface (CLI) access. Through their static analysis of the firmware of two devices they uncovered four username/password combinations hidden in the C-Data firmware:

- suma123/panger123
- debug/debug124
- root/root126
- guest/[empty]

And this initial backdoor CLI access could then be used to exploit other vulnerabilities. For example, they said that an intruder could also exploit a second bug to list credentials in cleartext in the Telnet CLI for all the other device administrators; credentials that could be used at a later point in case the backdoor account is removed.

A third vulnerability also allowed the attacker to execute shell commands with root privileges from any CLI account.

A fourth bug was discovered in the same Telnet server running on the WAN interface. The researchers said that this server could be abused to crash the FTTH OLT device. Since the server was running by default on the WAN interface, this bug could be used to sabotage an ISP's network if they're not filtering incoming traffic to the FTTH OLT devices.

But the devices also run a web server [of course they do] that's included to power the device's management web panel. Here they found the fifth bug: By downloading six text files from this web server, an attacker could get his hands on cleartext account credentials for the device's web interface, Telnet server, and SNMP interface.

And in case any of the passwords are in an “encrypted” format that's not a problem either, since all credentials are secured by XORing them against a known fixed string:

j7a(L#yZ98sSd5HfSgGjMj8;Ss;d)(&^#@$a2s0i3g

And last, but not least, the two researchers pointed out that all management interfaces on the tested devices ran in cleartext modes, with HTTP rather than HTTPS, Telnet instead of SSH, and so on. They said this opened devices and the ISPs that used them to easy MitM (man-in-the-middle) attacks.

As we know, responsible disclosure is the norm. But Pierre and Alexandre are sure that they cannot explain what they have uncovered as inadvertent mistakes. So they published their complete and detailed finding today without notifying the vendor because the nature of what they have found can only be explained as backdoor functionality intentionally placed into the firmware by the vendor.

They also noted that identifying all vulnerable devices may also be a problem for ISPs, as some of the vulnerable equipment appears to have been sold as a white-label product, under different brands, such as OptiLink, V-SOL CN, BLIY, and possibly others.

I can sympathize with their feelings, but this is still hugely irresponsible. Disclosure is not made responsibly to protect the vendor of the malfunctioning software or hardware. Disclosure is made responsibly to protect the USERS of the malfunctioning product. So, no matter whether or not this company's products may have been deliberately backdoored, the ONLY responsible thing to do was to inform them in private of what was found and give them a fixed hard deadline to update their products' firmware, removing all discovered backdoors, and send out urgent notices to all of their customers and OEMs informing them of the urgent need to update their firmware. Then, whether or not the company complied, only then go public with the details.

Instead, what they HAVE done really does create a HUGE mess. Now we have presumably many tens of thousands of pieces of high-end networking gear spread around the world, located in critical networking positions that cannot be readily taken offline... And everyone in the world now knows exactly how to exploit those devices. These guys could have been heroes, but that's not the path they chose.

And this brings us to an important point: How do purchasers of networking equipment know what they are getting? There's a huge temptation to choose an inexpensive foreign supplier who is offering a lot of functionality at a very attractive price. Maybe it'll be okay. Probably will. But what if it's not?

Drone Operator Locating

We all know what a problem little remotely piloted consumer drones can be when they're operated in the vicinity of a commercial airport. They can and have caused serious disruptions in aviation because a drone cannot be allowed to strike the intake of a jet engine.

As with anonymous postings on the Internet, drone operators have been able to operate under the assumption that they cannot be caught because they cannot be located. They're able to operate a good distance away, and with today's camera-equipped drones they no longer even need line-of-sight to their vehicles. But, if they could be reliably located the word would spread and the problem would be largely solved. A few highly publicized arrests and the game would change.

So far, the only solution that's been applied is the obvious one: Surround locations where drones pose a real hazard with radio receiving stations tied back to a central control point and attempt to use the radio frequency emissions from a drone operator's transmitter to geo-locate that transmitter. In an empty field in the middle of Nebraska that might work well. But a busy commercial airport in any modern urban center turns out to be a very different problem. The air is hugely contaminated not only with avionic radio frequencies including high power radar, but also the typical ground clutter of Cellular phones, Wi-Fi, Bluetooth, IoT and all manner of other RF. To further complicate matters, drone operation radio signals have short duration, their frequency usually hops over most of the band and they have relatively low power.

To address this problem, our industrious academics from Israel's Ben Gurion University of The Negev have been performing some research and have just published a new paper titled:

“Can the operator of a drone be located by following the drone’s path?”
https://orenlab.sise.bgu.ac.il/p/DroneLocation.pdf

So far, their work has been with simulations, but their underlying supposition appears to be holding: The path of a drone being remotely controlled by someone from a distance will inherently contain clues about the location from which that person is viewing the drone. Just one simple example is that the drone’s motion along the line-of-sight to the viewer will be much less obvious to that viewer than any motion perpendicular to their line-of-sight to the drone. So it’s reasonable to suppose that this might influence the drone’s path. Unfortunately, this system assumes that the drone is being piloted visually from a remote location, and not using an FPV drone-mounted camera. So the application might be limited in practice. But they have trained up neural networks and have been able to predict with 78% accuracy the operator's remote location knowing only the drone's flight path. So that’s pretty cool.

I just wanted to share this since I thought it was interesting and clever and not at all immediately obvious. And also since it was new work from our industrious academics at the Ben Gurion University of The Negev.

Rampant Router Insecurities

Despite titling this topic “Rampant Router Insecurities”, this is not what it appears. The tech press has been all hyperventilating over a new 25-page report from the Fraunhofer Institute titled: “Home Router Security Report 2020.”

https://www.fkie.fraunhofer.de/content/dam/fkie/de/documents/HomeRouter/HomeRouterSecurity_2020_Bericht.pdf

> This report analyses 127 current routers for private use developed by seven different large vendors selling their products in Europe. An automated approach was used to check the router’s most recent firmware versions for five security related aspects.
>
> We were able to extract completely 117 of the 127 firmware images. Four firmware images could be extracted partly and six firmware images could not be extracted at all. 116 of 127 (91%) devices are powered by Linux. One was powered by ThreadX and another one by eCos.
>
> The security aspects addressed in this report are:
>
> - When were the devices updated last time?
> - Which operating system versions are used and how many known critical vulnerabilities affect these operating system versions?
> - Which exploit mitigation techniques do the vendors use? How often do they activate these techniques?
> - Do the firmware images contain private cryptographic key material?
> - Are there any hard-coded login credentials?
>
> Our results are alarming. There is no router without flaws. 46 routers did not get any security up-date within the last year. Many routers are affected by hundreds of known vulnerabilities. Even if the routers got recent updates, many of these known vulnerabilities were not fixed. What makes matters even worse is that exploit mitigation techniques are used rarely. Some routers have easy crackable or even well known passwords that cannot be changed by the user. Most firmware images provide private cryptographic key material. This means, whatever they try to secure with a public-private crypto mechanism is not secure at all.
>
> Nonetheless, vendors seem to prioritize security differently. Especially AVM does a better job than the other vendors regarding most of the security aspects. However, AVM routers are not flawless as well. ASUS and Netgear do a better job on some aspects than D-Link, Linksys, TP-Link and Zyxel.
>
> To sum it up, much more effort is needed to make home routers as secure as current desktop or server systems. Additionally, our evaluation showed that large scale automated security analysis of embedded devices is possible today. We used the Firmware Analysis and Comparison Tool (FACT) and it worked very well for almost all firmware images analyzed during this study. FACT is an open source software available on GitHub.

https://fkie-cad.github.io/FACT_core/
https://github.com/fkie-cad/FACT_core
So, is this good? No. Is it the end of the world as we know it? Also no. Everyone knows that

my blanket recommendation would be to use a current build of pfSense which is built on top of state-of-the-art FreeBSD and load it onto a little fanless multi-port appliance PC. You get a super-solid platform that is not being continually updated because there are no known problems. If there were, they'd have been fixed. And the darned thing will do anything you might wish for.

But, everyone already has some solution they like, or at least that they're using. So, what I'll note here is the one thing that Fraunhofer failed to mention: Unless any of these 127 routers is misconfigured to expose vulnerable open ports and services to the public Internet, ALL of these vulnerabilities are **internal** and accessible **only** from the LAN side of the router, not the WAN side. Okay, so a router has a hard coded private key. Sure, that's not as cool as if it generated a random key the first time it was turned on, so that every router of that make and model or firmware edition would be uniquely keyed. But it's not the end of the world. That's not to say that this is good, either. Remember that we recently talked about a router vulnerability that was so egregious that hostile code running on a page in the user's web browser could be used to reach out to a router and cause trouble. But in general, once you have hostile code running loose inside your network perimeter, things are already bad.

Note that ALL of the vulnerabilities cited by Fraunhofer require access to the router. So one way to secure the router would be to block access to the LAN-side management services except from one specific LAN IP address. With pfSense that's trivially done through its web-based UI. I don't recall seeing that generally available in consumer routers. But that way, no system on the LAN can access the router's management. Period. You could have a separate pokey old laptop that's no longer useful for much else. It's IP would be hard-coded and it would be your router-management laptop that's normally turned off and in a closet. With pfSense or some other advanced router, you could even give the LAN interface a second IP on a different private network, like 10-dot. That way the router's management would not even be in the same Ethernet broadcast domain as any of the other devices on the LAN. Or you could use a VLAN and place a cheap little VLAN-aware smart switch in front of the router to block any management traffic that's not tagged with the proper VLAN. Of course that would only work for wired traffic. But that's another solution: bind the management to a wired port so that nothing on Wi-Fi can see the management.

Anyway, you get the idea. Many solutions are available depending upon your network and its configuration. It's true that consumer routers are a problem. Older routers, like older Smartphones that may no longer be receiving updates may be more of a problem. But the problem is not existential and a bit of thought given to strictly limiting all inside access to the router might just pay off... if for nothing else than peace of mind.

Miscellany

Back to YouTube TV
<sigh> So I tried using SlingTV, and I didn't make it very far since I really really really need the recorded content "scrubber" to work correctly. "Working correctly" means that you can pause the content, then jump forward and backward by some amount of time and see a thumbnail of where you now are. It's critical for the way I want to watch content, and YouTube TV works exactly like that on our Roku. SlingTV provides a progressively faster but totally blind scan forward or backward. No fixed-time jumping. It's utterly useless for commercial skipping or for backing up to watch something again. So… just an FYI.

SpinRite Woke!

I was reading a couple days ago about how the Linux team has approved new terminology, banning terms like "blacklist" and "slave." We've been touching on this topic of insensitive language and terminology in our technology and I thought… "You know?... I could do my part, too."

As I've been working on SpinRite's forthcoming technology, I've been thinking about our new "woke" awareness of the challenges faced by those in any highly heterogeneous culture and environment. Even when we're all created equal, we are all still created differently. So, in that spirit, in a spirit of accepting these differences, I realized that labeling a sector as "bad" is really quite harsh. I mean, it's not really a BAD sector, at the moment it's just "checksum challenged." It's "error non-correcting" or just "having a weak bit." It's not BAD, it's just different from its neighboring sectors. And so SpinRite's newly enlightened job, will be to have a gentle conversation with the sector. SpinRite will check-in with it to see how it's feeling. SpinRite will work WITH the sector to bring about the change that will be in everyone's long term best interest. And, you know, four thousand and ninety-six bits is a LOT of bits. Oh my goodness! So, if it should turn out that this particular, otherwise beautiful and perfect little sector, just can't get the hang of holding onto every last one of those, oh-so-many bits, well, then SpinRite will find a nice place for it to go so that it just doesn't need to worry about all that any longer. It's a lot, after all. So it will be able to just rest and relax and live out the rest of its drive's life in peace. It will still be there, but a fresh and brand new sector will take over handling all of its bits for it so that it just doesn't need to worry about all that any more.

So, in the future SpinRite, we're **not** going to have any BAD sectors... NO! We're just going to have some that SpinRite decides have already worked as hard as they need to and it's time to just give them a rest... and give some brand new sectors that have been waiting all this time their own chance to hold onto all of those bits for their owner.

It's 2020 after all, and this really feels MUCH better.

Tsunami

Google has open-sourced a vulnerability scanner for large-scale enterprise networks consisting of thousands or even millions of internet-connected systems.

https://github.com/google/tsunami-security-scanner
https://github.com/google/tsunami-security-scanner/blob/master/docs/index.md

Google has named it “Tsunami.” They've been using it themselves internally at Google and it was recently made available on GitHub. It's not going to be a Google branded product, but it will be maintained, further developed and extended by the open-source community, very much the way Google first made Kubernetes available.
Of course, hundreds of other commercial or open-sourced vulnerability scanners already exist in the world. The difference here is that Google built the scanner with very large enterprise networks — like its own — in mind. This is not to suggest that it would not be just as useful for smaller environments. But, that it is inherently designed to scale well and this is critical: It is explicitly designed to absolutely minimize the production of false positive detections. It turns out that false positives are the bane of IT personnel when traditional scanners are let loose across a large enterprise. Tsunami is designed to run inside giant networks where even the slightest false-positive findings could result in sending incorrect patches to hundreds or thousands of devices, possibly resulting in device crashes, network crashes and countless wasted work hours. So, Tsunami is designed to gracefully tackle networks that include hundreds of thousands of servers, workstations, networking equipment, and IoT devices that are connected to the internet... all while providing the highest possible scanning accuracy with an eye toward minimizing false positives.

Google said it designed Tsunami with the ability to adapt to extremely diverse and extremely large networks without the need to run different scanners for each device type as is too often necessary today. This was accomplished by splitting Tsunami into two logically and operationally separate pieces:

The first Tsunami component is the scanner, known as the reconnaissance module. It scans a company's network for open ports and tests each port, attempting to identify the protocols and services running on each to prevent mislabelling ports and testing devices for the wrong vulnerabilities. Tsunami's port fingerprinting module was derived from the legendary NMAP network mapping engine then added a bunch of new code to that core.

The second component is the the more complex of the two. It takes the scanner/mapper's output as its input, takes each located device's and its exposed ports, selects from a list of vulnerabilities to test, and runs benign exploits to determine whether the device is vulnerable to attack. And this “vulnerability verification” module is the primary means through which Tsunami may be extended. It uses a flexible plug-in architecture to allow the entire vulnerability verification module to grow and evolve over time and through straightforward community extension. And it will allow security teams to add new attack vectors and vulnerabilities to check inside their networks.

You can imagine, for example, when news of the Chinese C-Data fiber terminator vulnerabilities became public, someone quickly adding a new vulnerability verifier to Tsunami in order to see whether there are any of those exposed and vulnerable... and to locate any that are.

Today, Tsunami comes with plugins to check for:

- Exposed sensitive UIs: Applications such as Jenkins, Jupyter, and Hadoop Yarn ship with UIs that allow a user to schedule workloads or to execute system commands. If these systems are exposed to the internet without authentication, attackers can leverage the functionality of the application to execute malicious commands.

- Weak credentials: Tsunami uses other open source tools such as ncrack to detect weak passwords used by protocols and tools including SSH, FTP, RDP, and MySQL.

With this public release of Tsunami, Google is not washing its hands of the project. They plan to continue enhancing Tsunami with new plug-ins to detect an ever widening variety of exploits. Plug-ins for Tsunami will be released in a second dedicated GitHub repository. Google said that Tsunami will focus on meeting the goals of high-end enterprise clients like itself, and the conditions found in these types of large and multi-device networks.

Over on their GitHub documentation page Google posed the question “Why Tsunami?” and provided their rationale:

> When security vulnerabilities or misconfigurations are actively exploited by attackers, organizations need to react quickly in order to protect potentially vulnerable assets. As attackers increasingly invest in automation, the time window to react to a newly released, high severity vulnerability is usually measured in hours. This poses a significant challenge for large organizations with thousands or even millions of internet-connected systems. In such hyperscale environments, security vulnerabilities must be detected and ideally remediated in a fully automated fashion. To do so, information security teams need to have the ability to implement and roll out detectors for novel security issues at scale in a very short amount of time. Furthermore, it is important that the detection quality is consistently very high. To solve these challenges, we created Tsunami - an extensible network scanning engine for detecting high severity vulnerabilities with high confidence in an unauthenticated manner.

Tsunami:

- Supports small manually curated set of vulnerabilities
- Detects high severity, RCE-like vulnerabilities, which often actively exploited in the wild
- Generates scan results with high confidence and minimal false-positive rate.
- Detectors are easy to implement.
- Is easy to scale, executes fast and scans non-intrusively.