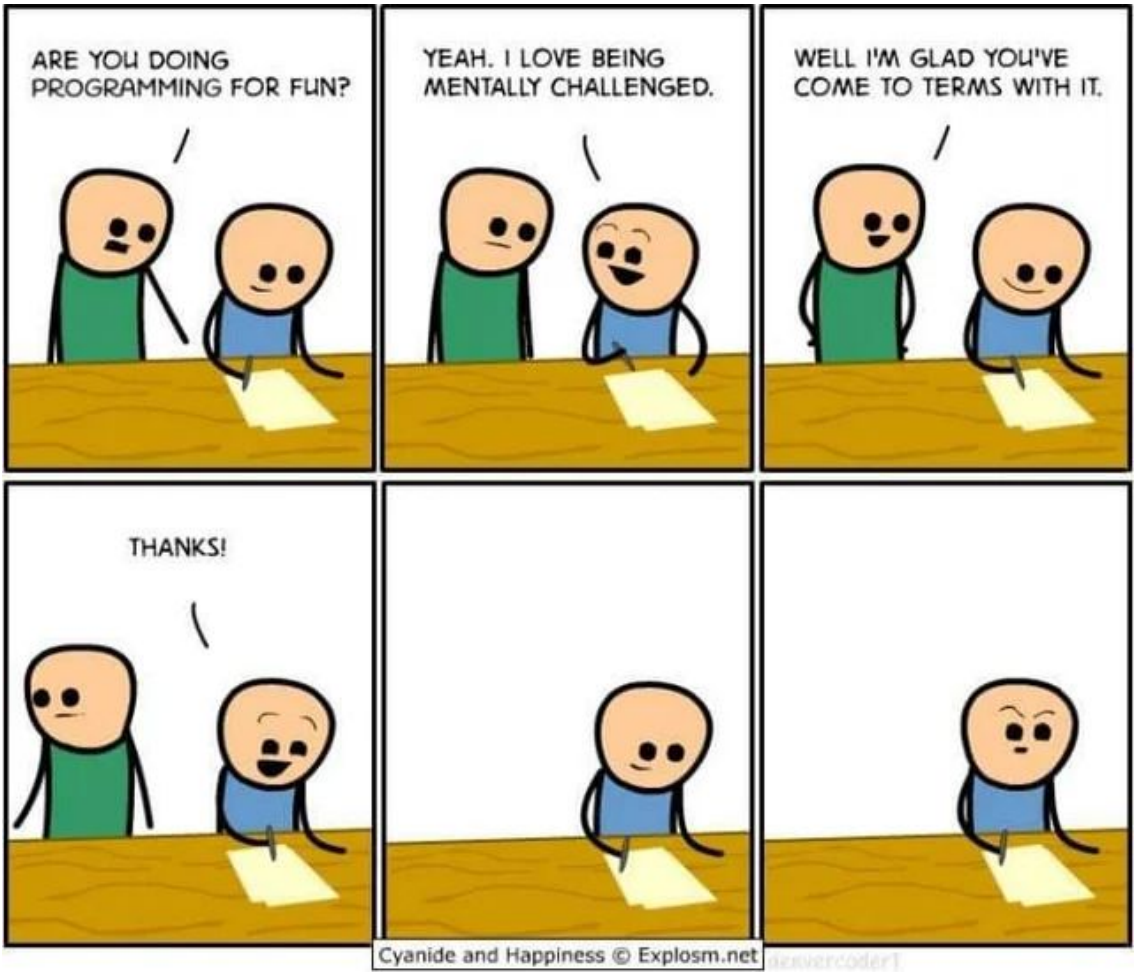
Security Now! #790 - 10-27-20

The 25 Most Attacked Vulnerabilities

This week on Security Now!

This week we examine a recently patched 0-day in Chrome and a nice new feature in that browser. We look at the site isolation coming soon to Firefox and Microsoft's announcement of Edge for Linux. We have some movement in the further deprecation of Internet Explorer, and a potentially massive SQL injection attack that was recently dodged by more than one million WordPress sites, despite the fact that some admins complained. Then we have a bit of miscellany, closing the loop feedback, and an update on my work on SpinRite. And we end by looking at the NSA's recently published list of the top 25 network vulnerabilities being used by malicious Chinese state actors to attack US assets.



Browser News

A Critical 0-day in Chrome

The Hacker News summed it up by writing: "Attention readers, if you are using Google Chrome browser on your Windows, Mac, or Linux computers, you need to update your web browsing software immediately to the latest version Google released earlier today." And even though that was last Tuesday, and even my Chrome's have since updated, our listeners may just want to double check that they, too, are now running v86.0.4240.111. However, there's **much** more to last week's emergency update than what drove it. But we'll start with that...

Last Tuesday's release closed 5 five vulnerabilities, 4 rated high severity and one medium. And one of those 4 high severity vulnerabilities was a 0-day that was seen exploited in the wild by attackers who used it to hijack targeted computers.

That nasty one was CVE-2020-15999. It's a heap buffer overflow in Freetype, the widely used open source font rendering library which is part of Chrome. Various bounty payout were or will be made for the other 4 vulnerabilities. But this biggie was discovered in-house by Google's Project Zero researcher Sergei Glazunov. But even so, it was subject to an accelerated 7-day public disclosure deadline since the flaw was under active exploitation. It only took one day for Google to begin pushing the update.

And since this flaw was actually in FreeType, not Chrome, Sergei also immediately notified the FreeType developers, who quickly developed an emergency patch to address the issue, also the next day, on October 20 with the release of FreeType 2.10.4.

Without revealing details of the vulnerability, Ben Hawkes, Project Zero's technical lead warned via Twitter that while the team has only spotted an exploit targeting Chrome users, it's possible that other projects that use any earlier versions of the FreeType library — and there will be roughly a gazillion — might also be vulnerable and are advised to deploy the fix included in FreeType version 2.10.4. Ben tweeted: "While we only saw an exploit for Chrome, other users of freetype should adopt the fix discussed here: https://savannah.nongnu.org/bugs/?59308 -- the fix is also in today's stable release of FreeType 2.10.4."

What we do know, thanks to what Sergei has shared, is that the vulnerability exists in the FreeType's function "Load_SBit_Png," which processes PNG images embedded into fonts. It can be exploited by attackers to execute arbitrary code just by using specifically crafted fonts with embedded PNG images. Since web fonts can be specified by a web page, and since the browser will download and render glyphs from those fonts, turning a theoretical FreeType flaw into an active exploit would not be difficult. Back when I created the "Off The Grid" cipher, which was based upon a Latin Square, I wanted to allow the user to choose from among a library of highly recognizable and blocky fonts. So I purchased a bunch of fonts and used the webfonts system to render them in the user's browser. It's not difficult to do.

Sergei added: "The issue is that libpng uses the original 32-bit values, which are saved in 'png_struct.' Therefore, if the original width and/or height are greater than 65535, the allocated buffer won't be able to fit the bitmap." Sergei has also published a font file with a proof-of-concept exploit. All of this increases the urgency of updating anything that uses the FreeType font renderer in a way that would allow an attacker to provide their own malicious font.

So, the “Load_SBit_Png” FreeType function:

1. Obtains the image width and height from the image header as 32-bit integers.
2. Truncates the obtained values to 16 bit to store them in a `TT_SBit_Metrics` structure.
3. It then uses the stored and truncated values to calculate the bitmap size.
4. And it allocates memory of that size for an image backing store.
5. And it then passes ‘png_struct’ and the backing store handle to a libpng function.

We also know that this bug was introduced in the June 9th, 2015 release of FreeType v2.6. So it's been in every subsequent FreeType release for the past five years.

Compared to that beauty, the other four are yawners, even though three of them are also ranked as severe. One's an implementation bug in Blink, another is a use-after-free bug in Chrome's media system, and the last is a use-after-free bug in PDFium. The remaining medium severity flaw is another use-after-free issue in Chrome's printing subsystem.

Over the past 12 months Google has patched three 0-day vulnerabilities in Chrome. Nearly one year ago was a critical remote code execution vulnerability patched last Halloween night, and the other was a memory confusion bug that was fixed in February.

Thanks to Chrome's mature self-maintenance system, the browser, which would normally be the first target for malicious abuse, is self-mitigating. But that may not be the case for every other use of FreeType. Anywhere an attacker can access a FreeType library built after June of 2015 and arrange to render their own font glyphs under FreeType, is potentially exploitable. For example, FreeType is the font renderer in Android, iOS and macOS. Java uses FreeType, as does the Sony PlayStation, many open desktop operating systems and video games. Like I said, it's pretty much everywhere today. So keep an eye out for FreeType updates and pay them some heed.

Chrome 86 is now blocking slippery notifications

I've mentioned before that I was taken aback when some site I visited prompted me to enable and allow all site-based notifications before it would allow me to proceed into the site. It said something like: “Please click ‘Allow’ to enable this web site and proceed...” Knowing that this was bogus, I, instead, clicked my browser's [BACK] button and chose the next useful-looking search engine link. No thanks.

The problem is that most users won't realize that this is entirely bogus and unnecessary, and that they are then giving that site permission to harass them with notifications which appear to be originating from their operating system since the browser, whether they are at that site or not, forms a conduit for subsequent messaging spam.

Fortunately, Google has noticed this too. What's more, they have observed sites using notifications for active malicious purposes including to send malware or to mimic system messages to obtain user login credentials. Google's Web Platform Product Manager PJ McLachlan

said “Abusive notification prompts are one of the top user complaints we receive about Chrome.” So, starting with v86, Chrome will be automatically suppressing website notification spam on all sites which have shown a pattern of sending abusive notification content to visitors. PJ said that “Our goal with these changes is to improve the experience for Chrome users and to reduce the incentive for abusive sites to misuse the web notifications feature.”

Google has had this on their radar since v80 and has been working to refine its operation. And they've become a bit crafty. Google’s web spiders will subscribe to push notifications if push permissions are requested in order to detect websites which are misusing notifications to subsequently spam their previous visitors. If such behavior is noticed, the spiders will use Google's Safe Browsing blacklist service to evaluate received notifications and automatically flag all sites abusing notifications for malicious purposes.

I've never been a big fan of off-site browser-pushed notifications. I suppose there are valid use cases. And as we become more browser-centric, receiving asynchronous notifications might make sense. But mostly I want to interact with a site by choosing to visit it.

“Site Isolation” coming soon to Firefox

Site Isolation is a state-of-the-art browser security feature which offers enhanced protection against some forms of security flaws by reducing each browser tab's attack surface. In practice it increases the difficulty of malicious code on one site from accessing the resources of another.

By enforcing the same-origin policy, browsers prohibit websites from accessing each other's data. But mistakes happen, security bugs occasionally arise, or new and creative ways are found to bypass these permissions.

So, site isolation provides another perimeter of defense to make such attacks much less likely to succeed. By placing web pages from different web domains into different operating system processes, the browser is able to leverage the underlying operating system's native and time-hardened inter-process isolation.

As we discussed about two years ago, Google added site isolation to Chrome in mid-2018, with the release of Chrome 67. And after seeing the feature's success, the following February 2019, Mozilla announced their own plans to bring site isolation to Firefox. Mozilla named their internal re-architecting project Fission, since it splits Firefox apart into separate processes. But this is much much easier said than done. In each case, it has required time-consuming rewrites of large portions of each browser's internal architecture and it required about two years for each project.

But Firefox is finally emerging from the other side of that effort. According to an update to the Project Fission wiki page, Site Isolation can now be enabled inside versions of Firefox Nightly and, in time, it will be move into the stable release channel.

To enable:
1. Access the about:config page
2. Set the “fission.autostart” and “gfx.webrender.all” prefs to “true”.
3. DO NOT edit any other “fission.*” or “gfx.webrender.*” prefs.
4. Restart Firefox Nightly.

If you then hover your mouse pointer over any tab the operating system's process ID (PID) of the process for that page will be added to the end of the pop-ups text.

I'm not going to worry about it now. But it will be great to have that added robust security in Firefox once it goes mainstream.

Microsoft's "Chredge" for Linux

As we know, Microsoft's Chromium-based Edge browser is available on Windows and Mac, iOS and Android. So the only major platform that was missing was Linux. The Dev channel build of Edge for Linux is now available for download and installation. And it's running all of Edge's features including support for smooth scrolling, Google extensions, themes, etc.

Now, that said, since Linux can already run Chrome or Firefox — I always seem to be running Firefox on Linux — I'm unsure why anyone would want to run Edge there. But developers, for example, might need to verify compatibility with their stuff on Edge for Linux. In any event, it's there for anyone who wants it. And I'm sure it'll emerge from the Dev channel in due course.

IE-to-Edge

Inertia in the computer world is really quite something. Despite years of Microsoft pushing their users away from the increasingly antiquated Internet Explorer, IE still commands a 5% — so 1 in 20 — market share. And the need to move away is not just for security — though that should be ample reason. It turns out that as IE has petrified, the rest of the web has move on. Microsoft now maintains a list of 1,156 web domains which no longer work with IE.

https://edge.microsoft.com/neededge/v1

And this list includes Twitter, Facebook, Instagram, Godaddy, Google Drive, Google Earth, Microsoft Teams, ESPN, Yahoo Mail, and a great many more mainstream and obscure sites, all which require a browser made sometime in this century.

To address this trouble, Microsoft has written a browser helper object (we haven't talked about BHO's in years). Microsoft will be installing the browser helper object into any remaining and still surviving instances of IE and, when one of these BHO-enhanced instances of IE attempts to bring up any of the incompatible sites, Edge will be launched instead, along with a not-so-subtle banner recommending that the user switch to making Edge their default web browser.

Security News

WordPress "Loginizer"

In this week's installment of "why you never want to host your own WordPress site," and also why you should really try to only run with the barest minimum of plug-in add-ons...

We now have the "Loginizer" add-on, installed in more than one million WordPress sites, which the WordPress security team took the rare step last week of forcing an update, using a virtually

unknown internal capability of WordPress. It turns out that WordPress sites can be forcibly updated without their owner's permission. And, as a result, sites running the Loginizer plugin were forcibly updated to version 1.6.4.

Earlier versions contained a SQL injection bug that could have allowed hackers to take over WordPress sites. Why would more than one million WordPress instances choose to install "Loginizer." Because it promised to enhance the security of WordPress sites by providing IP address black or white lists for accessing WordPress's login page. Among other login-related features it also provided support for two-factor authentication or simple CAPTCHAs to block login automation.

https://wordpress.org/plugins/loginizer/

Those things seem like features that should be built-in natively so that users are not forced to download and obtain them through what prove to be insecure add-ons. I'm 100% certain that the authors of these plug-in are well meaning. But all of the evidence we've seen informs us that writing web-facing browser add-ons is not simple. It requires extreme awareness of security and it should not be left to random, if well intentioned, authors. I get it that WordPress wants to have an active and vital add-on ecosystem to enhance their offering. But leaving popular features missing, such as at least some of those provided by Loginizer, hugely increases the footprint of exactly this sort of devastating event. Many of those more than one million potential attack targets could have been sidestepped from the start if WordPress natively offered the more obvious, popular and missing benefits of that add-on.

While I was hosting my own WordPress site, I also employed my own IP-based filter for that inherently vulnerable and abuse-prone login page because I was somewhat incredulous that out-of-the-box there was no strong native protection built-in. Since I'm running atop IIS, I did it with a web.config file which is the equivalent of the .htaccess file typically supported by Unix servers.

We should also note that this Internet-wide forced update was not without controversy. WordPress users have been given the impression that they control what's done when on their sites. There's a permission setting for allowing auto updates. So, of course, I immediately turned it **ON** for my installation. But many old school WordPress admins enjoy the illusion that if they have more control, things will go better. They are, of course, wrong.

Ryan Dewhurst, the Founder & CEO of WPScan was interviewed about this. He noted that the flaw "... allows anyone with some basic command-line skills to completely compromise a WordPress website." And Ryan pointed out that the flaw's discoverer had also provided a simple proof-of-concept script in a detailed write-up which was recently published. This was responsibly posted 5 days after the update was pushed to all known sites. But imagine allowing a known and powerful SQL injection attack to linger on more than one million WordPress sites. That bug is one of the worst security issues discovered in WordPress plugins in recent years, which probably explains why the WordPress team decided to forcibly push the patch to all affected sites.

Ryan Dewhurst told ZDNet that this "forced plugin update" feature has been present in the WordPress codebase since v3.7 which was released in 2013. He believes that it has rarely been used. But that's not entirely clear.

He said: “A vulnerability I myself discovered in the popular Yoast SEO WordPress plugin back in 2015 was forcibly updated. Although, the one I discovered was not nearly as dangerous as the one discovered within the Loginizer WordPress plugin.” He added “I'm not aware of any other [cases of forced plugin updates], but it is very likely that there have been others.” And, confirming that WordPress core developer Samuel Wood said the feature was used “many times” but declined to provide additional details about other instances.

Not everyone was happy with WordPress's unilateral move. Not long after the Loginizer 1.6.4 patch started reaching WordPress sites users started complaining on the plugin's forum on the WordPress.org repository.

One disgruntled user posted: “Loginizer has been updated from 1.6.3 to 1.6.4 automatically although I had NOT activated this new WordPress option. How is it possible?”

Another added: “I have the same question too. It has happened on 3 websites I look after of which none of them have been set to auto update.”

And in a follow-up, Loginizer's developer said that the security patch had reached 89% of all sites with the help of that forced-update.

All of this podcast’s listeners know that anything that's connected to the Internet needs to have some highly reliable means of being updated by its source when important problems are inevitably found. As we know, there is a grow body of IoT devices that have been stranded and will never be updated.

So my feeling is that WordPress should commandeer the best ideas from their add-on authors and implement them safely and securely. And also that WordPress should firmly disabuse its platform's administrators of the idea that they can prevent the automatic updating of anything when WordPress feels that it's sufficiently important. That guy who complained that three of his VULNERABLE SITES were updated without his permission, to remove a critical, complete site takeover, ought to pull his head out of you-know-where... And WordPress ought to also make their ability to do this a formally known policy.

Miscellany

I put this final piece under "miscellany" because it's less directly related to security than it is to masochism. The news is that the refusal of a system to perform a Windows 10 feature update may now be bypassed. So, you know, if things have been going well lately and your systems have been running without problems, and you're a bit bored... then forcing Windows 10 to update against its will may be just the thing to return some excitement to your life.

Microsoft has added a new group policy option that allows users to bypass the "safeguard holds" placed on devices due to known conflicts with hardware or software. Because, you know, maybe Microsoft is being overprotective and you'll get lucky. It could happen.

This month's Patch Tuesday updates added a new group policy titled "Disable safeguards for Feature Updates." (What could possibly go wrong?) The policy setting is located under Computer Configuration > Administrative Templates > Windows Components > Windows Update > Windows Update for Business.

The setting's description reads: "Enable this setting when Feature Updates should be deployed to devices without blocking on **any** safeguard holds. Safeguard holds are known compatibility issues that block the upgrade from being deployed to affected devices until the issue is resolved. Enabling this policy can allow an organization to deploy the Feature Update to devices for testing, or to deploy the Feature Update without blocking on safeguard holds."

I'm not suggesting that any sane person would or should do this. But I know that some of our listeners like to walk on the wild side. So here's your ticket.

Closing the Loop

DavidMD @SirSycho

Hi Steve... a little late here but ADFind (typically pronounced A-D-Find, as in Active Directory Find), is an awesome admin tool for those of us in corporate Windows environments. I actually know Joe, we used to work together at HP and he has been a Microsoft MVP for AD for over a decade. He knows his structured directories very well. ADFind can even be used to query OpenLDAP directories. It's unfortunate that his tools have been used by malicious parties but that seems to be more a function of the utility of his tools rather than a reflection on his motives.

I just wanted to speak up and defend Joe. Though I think he'd be the first to agree with you on his website design chops (probably why his tools are all CLI). Love the podcast! Keep up the great work!

SpinRite

The work on SpinRite is progressing very well and we're getting tantalizingly close to the first release candidate of the "ReadSpeed" benchmark. During the past week I significantly rewrote a chunk of the benchmark that will be important for SpinRite. As I mentioned before, I'm implementing as much of the technology that SpinRite will need as I can now, rather than later, so that more of it can be tested — sort of in vivo — as part of the "ReadSpeed" benchmark. The new maximum speed hardware drivers can enumerate and talk to all of a system's drives and controllers. But there will still be some drives that this new code doesn't yet handle, such as USB-connected devices. That will be added immediately following the addition of UEFI booting and operation. But I want to get directly-connected parallel IDE and all serial SATA drives running at lightning speed first. And that means synchronizing the BIOS's view of the system's drives with the new view that this code provides from the PCI bus. And that now appears to be working much better than I had hoped and expected.

I'm working with a couple of the testers to eliminate some one-off side effects of the rewrite, then we'll move to the first release candidate.

Meanwhile, GRC's web forums are bubbling along nicely. We have more than 3700 registered users and generally around 80 or so visitors just there looking and poking around. Although I originally created the forums as a place for GRC project and product support, there was so much interest in just hanging out and chatting that I created an array of topic-centric groups to better organize the community that has been forming. So now we have a dedicated Security Now Podcast forum, and forums for discussing Security, Networking, Operating Systems, Software, Hardware, Coding, Health, Nerd Recreation, Science Fiction, and one titled "None Of The Above."

Once the "ReadSpeed" benchmark is ready, that'll be the place to go to report and discuss your results, and to tell me about any problems that you might encounter. The whole point of this is to allow for the early verification of SpinRite's forthcoming technology. If "ReadSpeed" works on your systems — from booting through obtaining results — then so will SpinRite.

The 25 Most Attacked Vulnerabilities

Last week, the US National Security Agency (our NSA) published a detailed report enumerating the top 25 well-known security vulnerabilities that are currently being consistently scanned, targeted, and exploited by Chinese state-sponsored hacking groups.

https://media.defense.gov/2020/Oct/20/2002519884/-1/-1/0/CSA_CHINESE_EXPLOIT_VULNERABILITIES_UOO179811.PDF

As anyone who follows this podcast will already know or presume, it is today's security reality (which might have been a good name for this podcast) that all 25 of these vulnerabilities are well known and have had patches available from their vendors, typically for months or years. And the development of exploits for many of these is hugely aided by the publication of proofs of concepts and often outright working exploit demos. And, of course, it's not just state-sponsored Chinese hackers who are making hay. We've seen many of these flaws incorporated into the attack kits of ransomware gangs, malware groups, and nation-state actors from other countries including Russia and Iran.

The NSA's 6-page PDF states their theory of the case by writing:

> One of the greatest threats to U.S. National Security Systems, the U.S. Defense Industrial Base, and Department of Defense information networks is Chinese state-sponsored malicious cyber activity. These networks often undergo a full array of tactics and techniques used by Chinese state-sponsored cyber actors to exploit computer networks of interest that hold sensitive intellectual property, economic, political, and military information. Since these techniques include exploitation of publicly known vulnerabilities, it is critical that network defenders prioritize patching and mitigation efforts.
>
> The same process for planning the exploitation of a computer network by any sophisticated cyber actor is used by Chinese state-sponsored hackers. They often first identify a target, gather technical information on the target, identify any vulnerabilities associated with the target, develop or re-use an exploit for those vulnerabilities, and then launch their exploitation operation.
>
> This advisory provides Common Vulnerabilities and Exposures (CVEs) known to be recently leveraged, or scanned-for, by Chinese state-sponsored cyber actors to enable successful hacking operations against a multitude of victim networks. Most of the vulnerabilities listed below can be exploited to gain initial access to victim networks using products that are directly accessible from the Internet and act as gateways to internal networks. The majority of the [vulnerable] products are either [used] for remote access or for external web services, and should be prioritized for immediate patching.

Remember that it was recently necessary for the US Department of Homeland Security to **DEMAND** that all government agencies immediately update their Windows deployments to remove the ZeroLogon vulnerability — with a deadline — and also with the predictable consequence that at least some still did not.

So, in light of all this list, I thought it would be interesting for us to take a casual stroll through a single coherent summary of what the NSA is currently actively seeing online. Without stepping on the lead, I'll note that some of these 25 are surprisingly obscure. They don't tend to make big waves nor headlines, but they are nevertheless powerful network vulnerabilities.

It's interesting that some of these more obscure vulnerabilities are due to errors in "deserialization." We've talked about this before, but not recently, and it's most often seen with Java. The idea is that an object has a schema, a structured layout of data occupying RAM. The structure's definition often grows over time with all sorts of optional add-ons so that any particular instance of it is often only sparsely filled-in. In order to store it efficiently, and especially to transmit it over a communications channel, the object goes through a process known as "serialization" — not as in assigning it a serial number, but as in converting it from parallel to serial, or sequential data.

But the trouble arises at the receiving or "deserializing" end, when software wishes to restore the received or stored data back into its parallel or usable form. The same programmer's who wrote the serializer typically also write the deserializer. Right? So they naturally assume that the data their deserializer is deserializing was properly serialized by their own serializer. The trouble arises when that's not the case. And what is the inherent nature of any deserializer? It's an interpreter. And one of this podcast's fundamental precepts is that interpreters are surprisingly difficult to make bulletproof. They are one of secure software's greatest challenges and banes.

Okay, so what are remote threat actors actively scanning for and exploiting wherever possible?

1) CVE-2019-11510 - On **Pulse Secure VPN servers**, an unauthenticated remote attacker can send a specially crafted URI to perform an arbitrary file reading vulnerability. This may lead to exposure of keys or passwords

2) CVE-2020-5902 - On **F5 BIG-IP proxies and load balancer**, the Traffic Management User Interface (TMUI) —also referred to as the Configuration utility— is vulnerable to a Remote Code Execution (RCE) vulnerability that can allow remote attackers to take over the entire BIG-IP device.

3) CVE-2019-19781 - **Citrix Application Delivery Controller (ADC) and Gateway** systems are vulnerable to a directory traversal bug, which can lead to remote code execution without the attacker having to possess valid credentials for the device. These two issues can be chained to take over Citrix systems.

4+5+6) CVE-2020-8193, CVE-2020-8195, CVE-2020-8196 - **Another set of Citrix ADC and Gateway** bugs. These ones also impact SDWAN WAN-OP systems as well. The three bugs allow unauthenticated access to certain URL endpoints and information disclosure to low-privileged users.

7) CVE-2019-0708 **(aka BlueKeep)** - A remote code execution vulnerability exists within **Remote Desktop Services** on Windows operating systems. (Remember that Microsoft's cute euphemism for this was an "authentication bypass." I'll say!)

8) CVE-2020-15505 - A remote code execution vulnerability in the **MobileIron mobile device management (MDM) software** that allows remote attackers to execute arbitrary code and take over remote company servers.

9) CVE-2020-1350 **(aka SIGRed)** - A remote code execution vulnerability exists in **Windows Domain Name System** servers when they fail to properly handle requests.

10) CVE-2020-1472 **(aka Netlogon)** - An elevation of privilege vulnerability exists when an attacker establishes a vulnerable Netlogon secure channel connection to a domain controller using the Netlogon Remote Protocol (MS-NRPC).

11) CVE-2019-1040 - A tampering vulnerability exists in **Microsoft Windows** when a man-in-the-middle attacker is able to successfully bypass the NTLM MIC (Message Integrity Check) protection.

12) CVE-2018-6789 - Sending a handcrafted message to an **Exim mail transfer agent** may cause a buffer overflow. This can be used to execute code remotely and take over email servers.

13) CVE-2020-0688 - A remote code execution vulnerability exists in **Microsoft Exchange** software when the software fails to properly handle objects in memory.

14) CVE-2018-4939 - Certain **Adobe ColdFusion** versions have an exploitable Deserialization of Untrusted Data vulnerability. Successful exploitation could lead to arbitrary code execution.

15) CVE-2015-4852 - The **WLS Security component in Oracle WebLogic 15 Server** allows remote attackers to execute arbitrary commands via a crafted serialized Java object

16) CVE-2020-2555 - A vulnerability exists in the **Oracle Coherence product of Oracle Fusion Middleware**. This easily exploitable vulnerability allows unauthenticated attacker with network access via T3 to compromise Oracle Coherence systems.

17) CVE-2019-3396 - The **Widget Connector macro in Atlassian Confluence 17 Server** allows remote attackers to achieve path traversal and remote code execution on a Confluence Server or Data Center instance via server-side template injection.

18) CVE-2019-11580 - Attackers who can send requests to an **Atlassian Crowd or Crowd Data Center** instance can exploit this vulnerability to install arbitrary plugins, which permits remote code execution.

19) CVE-2020-10189 - **Zoho Manage Engine Desktop Central** allows remote code execution because of deserialization of untrusted data.

20) CVE-2019-18935 - **Progress Telerik UI for ASP.NET AJAX** contains a .NET deserialization vulnerability. Exploitation can result in remote code execution.

21) CVE-2020-0601 **(aka CurveBall)** - A spoofing vulnerability exists in the way Windows CryptoAPI (Crypt32.dll) validates Elliptic Curve Cryptography (ECC) certificates. An attacker could exploit the vulnerability by using a spoofed code-signing certificate to sign a malicious executable, making it appear that the file was from a trusted, legitimate source.

22) CVE-2019-0803 - An elevation of privilege vulnerability exists in Windows when the **Win32k component** fails to properly handle objects in memory. (Recall that spate of elevation of privilege vulnerabilities that were being released by SandboxEscaper early last summer? Yep... This is one of those.)

23) CVE-**2017**-6327 - The **Symantec Messaging Gateway** can encounter a remote code execution issue.

24) CVE-2020-3118 - A vulnerability in the **Cisco Discovery Protocol implementation for Cisco IOS XR Software** could allow an unauthenticated, adjacent attacker to execute arbitrary code or cause a reload an affected device.

25) CVE-2020-8515 - **DrayTek Vigor** devices allow remote code execution as root (without authentication) via shell metacharacters.

So there's the list. Even a 3 year old vulnerability, like that flaw in a Symantec Messaging Gateway remains on the hit list of well-financed state-level hackers.

Here in the US we have a naturally US-centric viewpoint. So we don't see stories about NSA or CIA hackers doing the same to China, Russia, Iran or North Korea. I really wish that none of this was happening, since it all just seems so destructive and wrong. But from a patriotic standpoint, I cannot help hoping that we're giving as good as we get... because we certainly do appear to be getting a lot.

Once upon a time, the ongoing level of network intrusions was mostly off the radar. It was embarrassing for any company to admit when they had discovered that their network security perimeter had been breached. So whenever possible it would be kept strictly on the down-low and dealt with as quietly as possible.

But now, as we detailed last week with Ryuk, we have Ransomware which proactively makes any such network intrusion embarrassingly public and impossible to hide. And look!!! It's happening EVERYWHERE! So the only sane appraisal would be that non-Ransomware attacks are also **everywhere** ... they're just continuing to go unreported, like once upon a time in the old days.