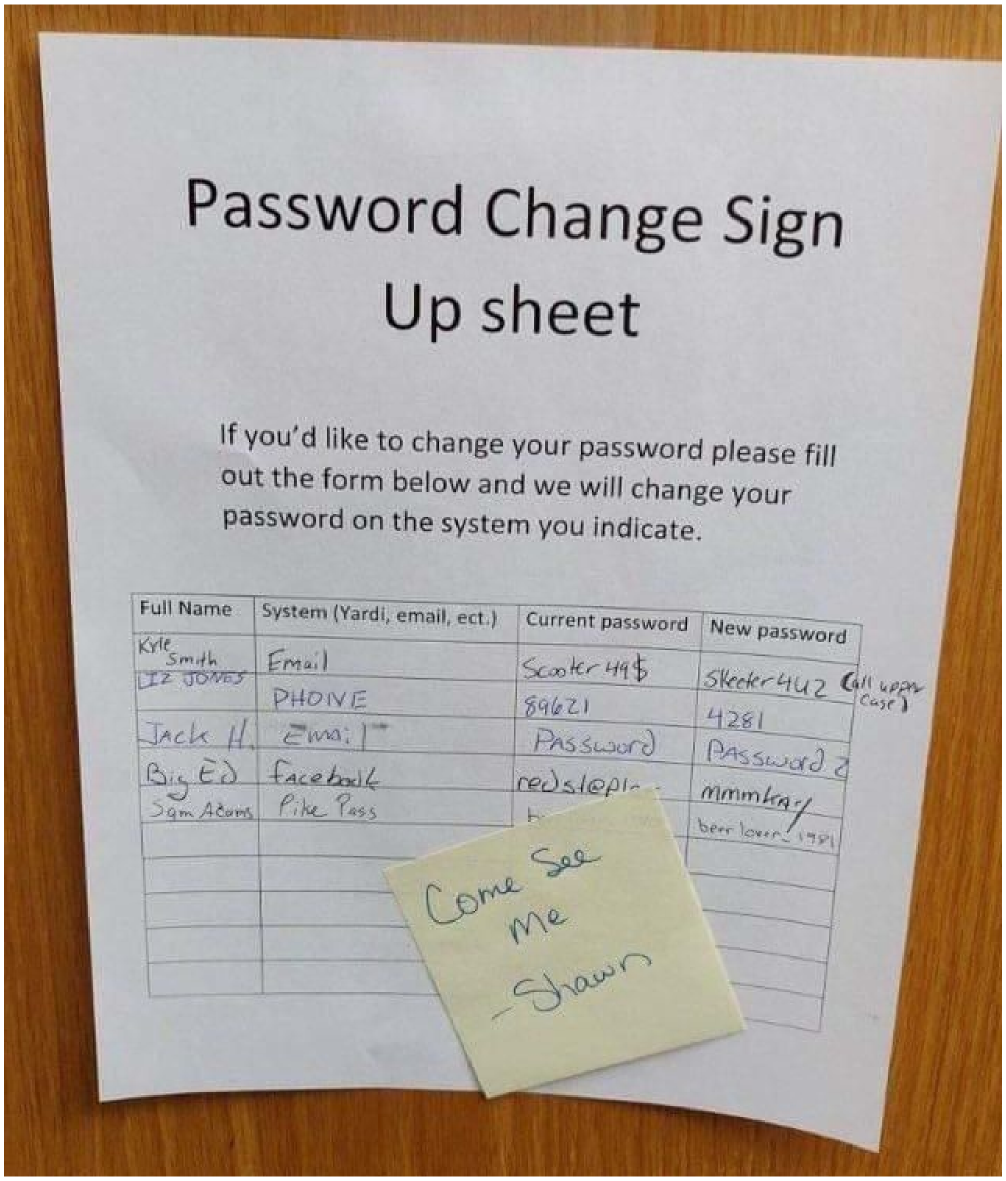
Security Now! #779 - 08-11-20
Geneva

This week on Security Now!

This week we note the completion of the first virtual Black Hat and DefCon conferences. We also examine the latest academic work to emerge from the Graz University which dramatically advances our understanding of the past few years of performance optimizing processor vulnerabilities. We look at the Ransomware attack on Canon, a mishandled vBulletin vulnerability disclosure, the forthcoming support for DoH on Windows 10, and the result of Troy Hunt's year-long quest to find a home for his much loved "Have I Been Pwnd" services. We have a bit of miscellany, some feedback, and an update on my SpinRite work. Then we examine a very interesting new technology being used to evade State-based Internet censorship known as "Geneva."

An oldie but goodie...



Password Change Sign Up sheet

If you'd like to change your password please fill out the form below and we will change your password on the system you indicate.

Full Name	System (Yardi, email, ect.)	Current password	New password
Kyle Smith	Email	Scooter 49$	Skeeter 442 (all upper case)
LIZ JONES	PHONE	89621	4281
Jack H.	Email	PASSword	PASSword 2
Big ED	facebook	redsl@pl-	mmmbeer/
Sam Adams	Pike Pass	[illegible]	beer lover 1981

Come See me
-Shawn

Security News

Patch Tuesday

> Woody Leonhard / @AskWoody
> MS-DEFCON 2: Patch Tuesday's tomorrow. Take a minute to make sure you have Automatic Update paused. You have to get patched sooner or later, but there's no reason to join the ranks of the unpaid beta testers.

We'll cover all of the details—and the now sadly inevitable fallout—once it's known, next week.

Black Hat 2020

Last week, August 1-6, the first virtual Black Hat event.
Many of the presentations are of special interest and/or somewhat obscure. But their titles are quite titillating. We'll be discussing a couple of the more interesting and relevant revelations, and in some cases the Black Hat presentations cover topics that were disclosed months ago which we have already covered in detail. But, I've created a GRC.SC shortcut link that will take our listeners directly to the main Black Hat briefings list of all presentations:

- https://www.blackhat.com/us-20/briefings/schedule/
- https://grc.sc/bh

Click on any presentation and you'll find a summary of the presentation, and links to the presentation slides, often an accompanying detailed White Paper that can be download, and sometimes even an accompanying software tool that can be downloaded,

Generalizing Speculative Execution Vulnerabilities

The Graz University gang are at it again. Their most recent paper is titled "Speculative Dereferencing of Registers: Reviving Foreshadow"
https://arxiv.org/pdf/2008.02307.pdf

In their paper they demonstrate that without a carefully conducted purely academic understanding of the impact of modern processor design optimizations it's not possible to make those designs truly secure. They show that the actual fundamental root cause underlying many of the previously disclosed speculative execution attacks, such as Meltdown and Foreshadow, was misattributed to the effect of prefetching. This misapprehension resulted in hardware vendors rushing out and releasing incomplete mitigations and countermeasures. This paper shows that the microarchitectural attacks were actually caused by speculative dereferencing of user-space registers in the kernel, and that it doesn't only impacts the most recent Intel CPUs — even those carrying the latest hardware mitigations — but also several modern processors from ARM, IBM, and AMD, all which were previously believed to be immune.

In other words, by developing a true understanding of the problem, they have come up with a new approach to attacking the security guarantees provided by all modern processors... and made no one but their thesis advisors happy in the process:

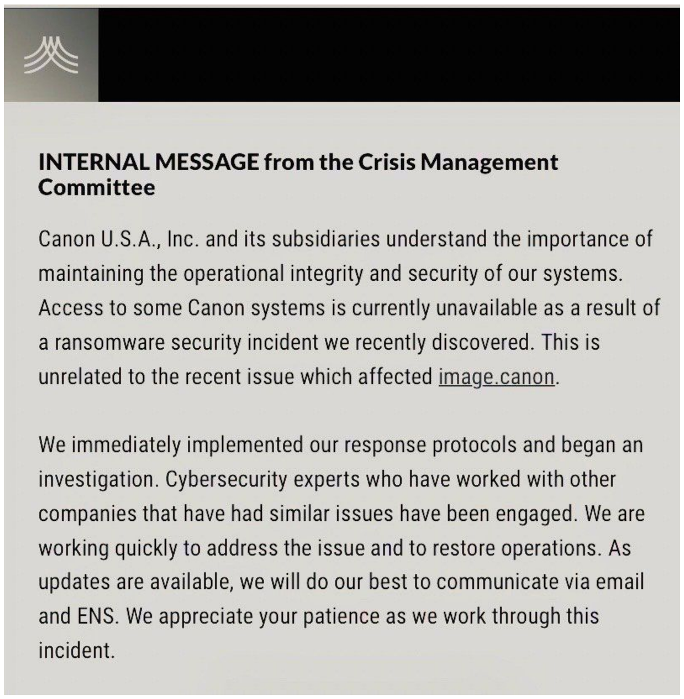
ABSTRACT
Since 2016, multiple microarchitectural attacks have exploited an effect that is attributed to prefetching. These works observe that certain user-space operations can fetch kernel addresses into the cache. Fetching user-inaccessible data into the cache enables KASLR breaks and assists various Meltdown-type attacks, especially Foreshadow. In this paper, we provide a systematic analysis of the root cause of this prefetching effect. While we confirm the empirical results of previous papers, we show that the attribution to a prefetching mechanism is fundamentally incorrect in all previous papers describing or exploiting this effect. In particular, neither the prefetch instruction nor other user-space instructions actually prefetch kernel addresses into the cache, leading to incorrect conclusions and ineffectiveness of proposed defenses. The effect exploited in all of these papers is, in fact, caused by speculative dereferencing of user-space registers in the kernel. Hence, mitigation techniques such as KAISER do not eliminate this leakage as previously believed. Beyond our thorough analysis of these previous works, we also demonstrate new attacks enabled by understanding the root cause, namely an address-translation attack in more restricted contexts, direct leakage of register values in certain scenarios, and the first end-to-end Foreshadow (L1TF) exploit targeting non-L1 data. The latter is effective even with the recommended Foreshadow mitigations enabled and thus revives the Foreshadow attack. We demonstrate that these dereferencing effects exist even on the most recent Intel CPUs with the latest hardware mitigations, and on CPUs previously believed to be unaffected, i.e., ARM, IBM, and AMD CPUs.

What does this translate to in the real world? The researchers established a cache-based covert channel that exfiltrated data from a process running on an Intel Core i7-6500U CPU to their stealth process, achieving a transmission rate of 10 bit/s and relaying a total of 128 bytes from the sender to the receiver process. 128 bytes is the length of four 256-bit cryptographic secrets. They also worked out a means of leaking the register contents from an Intel SGX enclave to extract a 32-bit value stored in a 64-bit register within 15 minutes. And, to give their research a bit more bite, they also demonstrated that some of their attacks can be performed at a distance using JavaScript with WebAssembly.

There's nothing that we as end users need to, or can do, about any of this. In this mess we're spectators. The only full and true solution requires our processors to further back away from their dearly beloved performance enhancements. It may be that someday we'll be given a choice between performance with reduced security or full security with a performance compromise. It would make sense for unshared personal PCs to opt for performance and for cloud-based shared infrastructure beasts to take the enforced security route.

Canon hit by the Maze ransomware

We can add Canon to the growing list of large high-profile companies who have a lot to lose who have recently had their security compromised by a quite public ransomware attack. Bleeping Computer was all over this one and, as they did with Garmin, they solicited and were able to obtain some previously unpublished internal memos in advance of any Canon official disclosure. Canon’s many services dropped offline on Thursday, July 30th and remained down until they began returning to use six days later, one week ago, last Tuesday, August 4th.



INTERNAL MESSAGE from the Crisis Management Committee

Canon U.S.A., Inc. and its subsidiaries understand the importance of maintaining the operational integrity and security of our systems. Access to some Canon systems is currently unavailable as a result of a ransomware security incident we recently discovered. This is unrelated to the recent issue which affected image.canon.

We immediately implemented our response protocols and began an investigation. Cybersecurity experts who have worked with other companies that have had similar issues have been engaged. We are working quickly to address the issue and to restore operations. As updates are available, we will do our best to communicate via email and ENS. We appreciate your patience as we work through this incident.

Internal notice sent to employees

A list of 24 different Canon-related domain properties appear to have been affected, though Canon states that the similarly-timed outage of their image.canon cloud photo and video storage service, which offers 10GB of free image upload and storage, was not related. Nothing is currently known about the way this was resolved, but Cannon's services

A vBulletin Emergency
This story begins just two days ago, on Sunday August 9th, with a posting by a security researcher, Amir Etemadieh, who goes by his Twitter @zenoflex. He maintains a blog at "exploitee.rs" and to give you some feel for his approach, his recent postings have been titled:

- Rooting the FireTV Cube and Pendant with FireFU
- All Your Things Are Belong To Us
- Hacking the Western Digital MyCloud NAS
- Re-Hacking The Samsung Smartcam

So this guy clearly likes poking at things and documenting his discoveries. He's also currently looking for a job. He recently posted a pinned Tweet addressed to Twitter stating on July 6th:

> Hey twitter! I'm actively looking for a job. An ideal role would include full time exploit dev (remote) w/ great benefits. I have crashes, fuzzers, & (good) 0day. If you have an opening that you think I may be a good fit for, contact me zenofex [at] http://zenofex.com

Unfortunately, he followed up that job solicitation by very irresponsibly posting his discovery and full details of a true 0-day exploit against vBulletin without ever giving them a heads-up. I'm not Twitter, he's clearly a talented reverse engineer, and I'm sure he understands what he has just done... and doesn't care. I would not want this person inside my company.

Sunday's particularly explosive security bomb posting was titled: "Exploiting vBulletin: A Tale of a Patch Fail" https://blog.exploitee.rs/2020/exploiting-vbulletin-a-tale-of-patch-fail/

Amir writes:

On September 23, 2019 an undisclosed researcher released a bug which allowed for PHP remote code execution in vBulletin 5.0 through 5.4. This bug (CVE-2019-16759) was labeled as a 'bugdoor' by a popular vulnerability broker because of its simplicity, and was marked with a CVSS 3.x score of 9.8 giving it a critical rating.

Today, we're going to talk about how the patch that was supplied for the vulnerability was inadequate in blocking exploitation, show how to bypass the resulting fix, and release a bash one-liner resulting in remote code execution in the latest vBulletin software.

The vulnerability mentioned above was later formally labeled "CVE-2019-16759" and a patch was issued on September 25, 2019. Although the patch was provided in under 3 days, the patch seemed, at the time, to fix the proof of concept exploit provided by the un-named finder.

Amir then proceeds to take this apart step-by-step. And like Sandbox Escaper, he's clearly a skilled technician. His posting beautifully walks its reader though the way vBulletin's on-the-fly PHP authoring engine operates and he produces a proof-of-concept. And not just one. He first offers a single CURL command line, and provides the bash script he promised earlier. He shows a Python exploit and indicates that he's in the process of pushing a public metasploit module. He even produced and published a deck of PowerPoint slides:

https://download.exploitee.rs/file/vbulletin/Exploiting_vBulletin_5.6.2_A_Tale_of_a_Patch_Fail.pptx

He then concludes by offering what he describes as a short-term fix:

This fix will disable PHP widgets within your forums and may break some functionality but will keep you safe from attacks until a patch is released by vBulletin.

- Go to the vBulletin administrator control panel.
- Click "Settings" in the menu on the left, then "Options" in the dropdown.
- Choose "General Settings" and then click "Edit Settings"
- Look for "Disable PHP, Static HTML, and Ad Module rendering", Set to "Yes"
- Click "Save"

And what happened next? Unfortunately, vBulletin sites are easy to locate with any Google search for the phrase "Powered by vBulletin." VBulletin is one of the more popular forum software systems. It is used by many major brands including Electronic Arts, Zynga, Sony, Pearl Jam, NASA, Steam, and many many more. The attacks started almost immediately.

Jeff Moss, aka "The Dark Tangent" who is perhaps best known as the creator of the Black Hat and Defcon security conferences tweeted:

A new VBulletin Zero Day got dropped yesterday by @Zenofex that revealed the CVE-2019-16759 patch was incomplete - within three hours https://forum.defcon.org was attacked, but we were ready for it. Disable PHP rendering to protect yourself until patched! https://blog.exploitee.rs/2020/exploiting-vbulletin-a-tale-of-patch-fail/

Amir's reply to Jeff was "Thanks for the mention" though I'm sure Jeff would have preferred not to have his defcon.org site attacked.

At around the same time Amir tweeted: "Looks like the @vBulletin forums are currently down while they patch from the RCE vulnerability release. Hopefully that means customers will also see a working patch soon."

Yeah. Gee. Or, instead of being attacked by an irresponsibly-disclosed 0-day, customers would probably have much preferred that he disclose this quietly to vBulletin, who would certainly have taken it very seriously. Last September's publication of the original 0-day, the fix for which this is a workaround, triggered a massive wave of vBulletin hacks which resulted in many companies disclosing security breaches through the months that followed. Since Amir did disclose this vulnerability, it's clear that he didn't want to use it himself. But the disconnect here seems to be that he won't mind at all if he enables others to wreak havoc in his name.

And, sure enough, yesterday we have a thread on the vBulletin support forum:

- This morning I received the scan results on my server with a:
  '/home/*******/public_html/forums/web.php'
  Known exploit = [Fingerprint Match] [PHP Shell Exploit [P1747]]
  Indicating I have some type of issue. It was recommended that I replace the web.php file. That is fine but I don't know what this file does or where another copy of it would be.

- Yesterday, 9:27am
  My site was also compromised this morning at 6:35 AM GMT-7. The file is named 1.php and was placed in the root of the site. Windows Defender identified the exploit as Backdoor:PHP/Shell.Q
  https://www.microsoft.com/en-us/wdsi...tid=2147682386
  I've zipped up the file and attached here as Backdoor.zip.

- Yesterday, 11:15am
  I got hacked at noon (CST) today -- was on 5-6-1 Patch Level 1.
  Have upgraded to 5-6-2. Can provide http access and error logs to Wayne Luke if you guys need help determining the attack vector.

- vBulletin Technical Support / Yesterday, 11:25am
  We know the vector and a patch is being developed. Hopefully, I can say more soon. The only workaround I can say at this time is to make your vBulletin directory read only (chmod 444).

  (That remediation was later found not to work because the access to .htaccess was blocked.)

- Yesterday, 11:44am
  I found the files hax.php at these locations:
  includes/vb5/template/cache/hax.php
  includes/vb5/template/bbcode/hax.php

- Yesterday, 12:20pm
  My 5.6.1 site was hit and I cannot access either url. I have 444'd it.
  My 5.6.2 sites appear unaffected.

- Yesterday, 12:57pm
  Hi, is this the best place to be kept up to date about this issue? Our 5.6.2 forum was hit as well today. Is there a CVE number of track the issue yet?

- Yesterday, 2:06pm
  Patches are available.
  https://forum.vbulletin.com/forum/vbulletin-announcements/vbulletin-announcements_aa/4445227-vbulletin-5-6-0-5-6-1-5-6-2-security-patch

Needless to say, if you or anyone you know is using version 5 of vBulletin, it needs to be patched last Sunday. And to the common refrain that it's crucial to keep lines of communication open to the vendors of the software you're using, I'll now add that this reality suggests that some level of compromise needs to be factored into any security-conscious organization.

It's similar to the fact that intrusion detection has now become an accepted part of an overall security posture. Detecting an intrusion inherently acknowledges that there's an intruder to detect. And that's never good. But if there is going to be an intruder to detect, it's far better that it be detected than not.

We know that the proper way to think about security is that it is inherently porous. The sobering truth is that this means that systems must be made less brittle. By that I mean that the event of a security breach should bend but not break an organization's security. A practical example of this might be to recognize that PHP-based forum software has grown quite complex. And we all know that means. With complexity comes risk. So, to mitigate that risk, software like PHP-based forums should be run on its own hardware behind a firewall that even gaining root access on that machine cannot bypass. The typical forum only needs to have access to an organization's DNS and eMail servers. I would argue that it should have its own local SQL database and not be reaching outside to the organizations shared SQL server. Containment in the event of a breach is the point. And since DNS and eMail services exist at fixed addresses and only need a couple of ports, a forum-hosting machine's network connection should be placed behind a "deny any" external firewall that only makes exceptions to that for a few ports. After all, this is the reason we call a firewall... a firewall. If anything catches fire and explodes on that machine, which in this case might be inevitable, the damage will be contained.

DoH for Win10
We don't talk much about the "Fast Ring", the "Slow Ring" or the "Release Preview Ring" because even being on the "How do I get off this rollercoaster Ring" provides sufficient excitement these days. But those rings fascinate Paul and especially MaryJo, who talk about

them all the time over on Windows Weekly. While we're on the subject, I'll note that the paint is still wet on the renaming of those “Rings” to now refer to them as “Channels.” In order of decreasing excitement and increasing sanity and stability we have the “Dev Channel”, the “Beta Channel” and the “Release Preview Channel.”

As we know, back in November of 2018 Microsoft surprised us by announcing that Win10 would natively be getting encrypted DNS with DoH — DNS over HTTPS. So not just random browsers running on the OS, but everything in the OS. This will actually happen when the “Dev Channel” gets the Win10 Insider Preview Build 20185. However, this early access news doesn't move me at all, since I'm definitely happy right where I am, getting actual work done over in the “Won't you please just stop fussing with it” channel.

But, once this does migrate to one of our “fasten your seatbelt” updates for the rest of us, we'll definitely be talking about its configuration and details in some depth because I'm sure our listeners will be interested.

Hasn't Been Pwned

Without success, Troy Hunt has just concluded, without success, a year long search for someone to purchase his very popular and useful “Have I Been Pwned” site and facility. Have I Been Pwned is really a much bigger deal than it might seem from the outside. For example, in his posting in March of this year, following 11 months of work to find a partner, he noted…

A Lot Happens in 11 Months
I onboarded 5 new governments onto HIBP: Austria, Ireland, Norway, Switzerland and Denmark (and a 6th one about to be announced any day now). I loaded 77 new data breaches comprising 1.7B records into HIBP and signed up almost 400k more individual subscribers to the service. I built and launched the authenticated API and payment process (I really should have done this earlier, I'm so happy with it!)

So, yeah... there's a lot there.

As I said, he posted the story of his 11-month quest back in March. It is really and truly a fantastic read. He describes the insanity of endless meetings and notes the processes' bizarre similarity to much-loved episodes of “Silicon Valley.” If we didn't have way too much else to get to, I would read it into this podcast. Since I really believe it's worthwhile I've given it a grc.sc shortcut: https://grc.sc/troy You really should check it out:

https://www.troyhunt.com/project-svalbard-have-i-been-pwned-and-its-ongoing-independence/

So that was as of March of this year. I can't even really summarize it, because there's so much interesting content. But Troy began the wrap up of that posting by writing:

So, What's Next for HIBP and for Me?
To be honest, I need some time to recover. What I've explained in this post will never adequately illustrate just how stressful this process was. I need some time where I'm not

waking up dreading how much work will have landed in my inbox overnight. I need some time to write more code and more blog posts, two things that remain my passion but had to take a back seat during this process. I'll still keep running HIBP as I always have, but I need the head-space to get my energy levels back up and plan the next phase. I've (almost entirely) cleared my calendar for the next few months to give me that much-needed time out and with coronavirus causing a heap of conferences to be cancelled and travel plans to be disrupted, it's probably not a bad time to stay home anyway.

And this brings us to today and Troy's posting last Friday:

https://www.troyhunt.com/im-open-sourcing-the-have-i-been-pwned-code-base/

"Let me just cut straight to it: I'm going to open source the Have I Been Pwned code base. The decision has been a while coming and it took a failed M&A process to get here, but the code will be turned over to the public for the betterment of the project and frankly, for the betterment of everyone who uses it. Let me explain why and how."

And I'm stuck again, wishing that I could share Troy's detailed description of his plans for HIBP. But at least the gist of it is clear: HIBP will become a major and significant open source project on Github so that Troy is no longer carrying the entire burden of moving it forward. It's clear that there's huge enthusiasm for the features offered by the system. And Troy's approach has always been to be entirely open and transparent anyway. So this is the natural next step. We'll certainly be keeping an eye on it!

Miscellany

"BloatBox" for debloating Win10 misses the mark.

Security Now shortcut of the week:

https://grc.sc/779

https://www.bleepingcomputer.com/news/microsoft/useful-registry-hacks-to-optimize-your-windows-10-experience/

1. Disabling Bing Search in Windows Search
   Windows Search comes with the Bing search engine integration, and it allows you to find the content on Bing when a local search fails to find anything. If you don't like Bing due to privacy or performance issues or if it's causing problems, you can disable it via the Registry.

2. Disabling the Windows 10 lockscreen
   The lock screen feature of Windows 10 could be unnecessary when you don't have a touch-enabled display. While there are several settings available in Windows 10 that you can modify to improve the lock screen experience, you can also remove it by editing your Registry.

3. Add 'Open command window here' to the File Explorer context menu
   By tweaking your Registry, you can also restore the context menu option to open a Command Prompt window and have it open automatically in a specific folder, as shown below.

4. Display seconds on the taskbar clock
   By default, the taskbar of Windows 10 shows only the hour and minutes and does not provide a built-in option to also show the seconds. Fortunately, you can modify your registry and enable support for seconds on Windows 10's taskbar clock, as shown below.

5. Enable Verbose mode in Windows 10
   With a tweak to the Registry, you can boot your Windows 10 PC into "Verbose mode" and get more detailed information and extremely helpful when troubleshooting problems. This mode will display the specific step the operating system is on while booting up and shutting down Windows 10. This mode allows you to troubleshoot startup and shutdown issues to see the specific steps are being performed when a problem occurs.

Closing The Loop

StarKiss / @StarKissedOne
@SGgrc Thanks for mentioning QNAP in the last SecurityNow. Found that my firmware was >4 revisions out of date. Got absolutely no notifications! Love the podcast!

SpinRite

- 65,536 sectors vs 32,768 sectors.
- Link layer faults being covered up.
- Really interesting "slow spots" — are they early indications of regions that should be selectively rewritten?

For a brief few hours, Sunday, there were no known problems with the AHCI driver code. Earlier in the day, Sunday, I had resolved the problem with a really troubled Kingston SSD on an Acer Aspire laptop by making the AHCI driver insanely tolerant of SATA-link troubles. This will mean that SpinRite will be able to notify its users when there’s a hardware-connection problem between their system and any drive. And by purchasing a late 2013 macBook Pro, and replacing its SSD with a specific model, I was also able to finally recreate and resolve a puzzle surrounding its power management. That’s solidly fixed.

But then, yesterday, a new problem with a Samsung 860 EVO on an AMD chipset appeared. The problem appears to be the way I’m resetting the SATA port. Different chipsets want it to be done differently. So I may need to make my port resetting logic more heuristic than it currently is.

This is the way the sausage is made. What this painstaking iterative process will eventually produce is what everyone wants: A single AHCI driver for SpinRite that’s smart enough to work on every machine and drive it will ever encounter.

Geneva

China's Great Firewall Tightens
A few weeks ago, China added a rule to block one of TLS v1.3's new privacy-enhancing features: ESNI = Encrypted Server Name Indication.

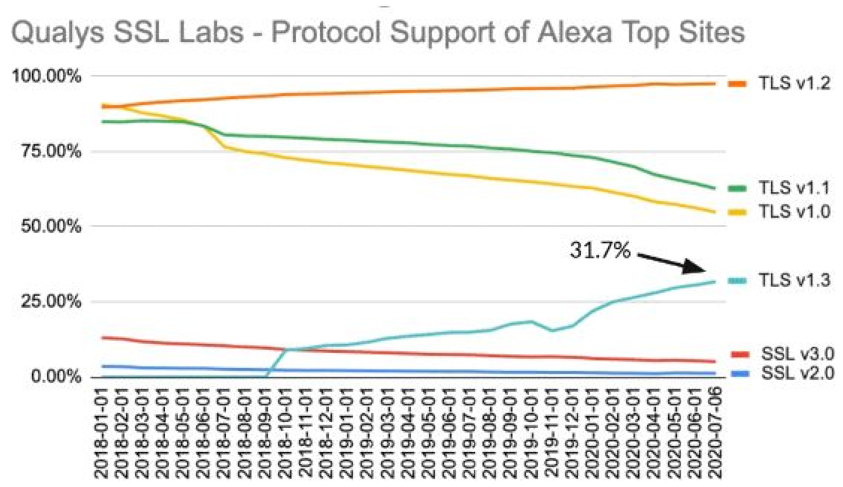
Once upon a time, a single IP address could only be bound to a single web server certificate. When any client connected, the web server listening at that IP's port 443 would respond with the certificate for the domain that was associated with that IP.

This prevented the practice that is now very common and crucial for IPv4, known as multiple hosting. Multiple hosting allows multiple separate domains to all point to the same single IP. But for that to work, the client (typically a web browser) must indicate from which among multiple domains at that IP it wishes to be served. So an extension was added to SSL to allow the client's SSL “Hello” handshake packet to make that specification clear. The web server answering at that IP would inspect the client's incoming SSL Hello packet. If it was in possession of a valid web certificate matching the domain the client was requesting, it would select that certificate for the subsequent SSL negotiation, and would also remember which website to serve to the client.

From a privacy standpoint, the trouble with this is that the so-called "SNI" — Server Name Indication — necessarily had to be provided to the server right up front, before the SSL connection's privacy encryption could be established. So that allowed any passive eavesdropper to easily determine to which domain this nascent SSL connection was being directed.

This overt lack of privacy drove those evolving the TLS standard to work out a means, in TLS v1.3, of adding encryption to the SNI extension, creating ESNI - Encrypted Server Name Indication. This is good for user privacy, but it’s a bane for anyone having an anti-privacy agenda, such as, presumably, the Chinese State, which has a manifest interest in controlling and censoring what its citizens are allowed to and able to access.

Since adoption of new security standards is notoriously slow, this wasn’t initially a problem. But as TLS 1.3 usage has continued to grow, the growing usage of ESNI had been giving Chinese sensors some headaches: (Beginning of 2018 to Now…)



As I noted at the top, China reacted with a bit of a heavy hand by simply blocking any TLS 1.3 connections using ESNI. Other HTTPS traffic is still allowed through the Great Firewall, if it uses an older version of TLS or non-encrypted SNI. It's unclear what they expect to happen. TLS 1.3 and ESNI are here to stay. But perhaps not in China.

I found an entry in the IETF list archive dated July 30th titled: "[TLS] Possible blocking of Encrypted SNI extension in China"

```
> The Great Firewall of China may have identified and blocked Cloudflare's
> ESNI implementation.
>
> I have found that when using a TLS client hello with ESNI extension to
> connect to servers behind Cloudflare's CDN, the connection will be cut
> off after the whole TLS handshake is done. And then, that IP address
> will be blocked at the TCP level for several minutes.
```

Some additional probing of the phenomenon revealed that:

- The detector is not merely matching on the lack of plaintext SNI; it is specifically looking for the ESNI extension identified by the identifier 0xFFCE.
- Also, the "encrypted_server_name" extension has to be syntactically correct. The detector is not just looking for the byte pattern 0xFFCE. (That's good, since no one wants false-positive triggering of the block.)
- Once an ESNI-containing ClientHello is detected, the firewall drops packets in the client→server direction for 120 or 180 seconds.
- The detector is running on all TCP ports, not just 443.
- It's already known that most of the functions of the Great Firewall work bidirectionally, and this new ESNI detection and blocking does, too. Sending an ESNI-containing ClientHello from outside China to a server inside also results in temporary blocking. This is convenient since it allows for external experimental probing with inbound connection packets.

And this brings us to GENEVA, the acronym for "GENetic EVAsion"

A web browser's access to the domain https://censorship.ai/ redirects its visitor to https://geneva.cs.umd.edu/ UMD.EDU is the University of Maryland.

Geneva is an experimental genetic algorithm that attempts to discover how to evade censorship by manipulating the packet stream on one end of the connection to confuse the censor. Geneva consists of two components: its genetic algorithm and its strategy engine. The strategy engine runs a given censorship evasion strategy using active network traffic; the genetic algorithm is the learning component that evolves new strategies (using the engine) against a given censor.

Being a genetic algorithm inspired by biology, and just as biological systems compose simple building blocks (the A, T, C, and G of DNA), Geneva generates new algorithms by compositing just a few simple ways of manipulating packets. Specifically, it can duplicate, tamper, drop, or fragment packets. Geneva composes these individual actions into "action trees."

Action trees have a trigger; this controls which packets the action trees act upon and together, these action trees form censorship evasion strategies.

Geneva initially creates many random individual strategies and runs each of them against real censors. Based on how successful they are (and other factors), it assigns a numerical fitness to each individual strategy. The most fit survive from one generation to the next, and Geneva mutates and mates strategies to create new ones.

The key to Geneva's success is its fitness function, which encourages the genetic algorithm to explore the space of strategies that does not damage the underlying TCP connection. Over many successive generations, if Geneva discovers a strategy that defeats censorship, the fitness function encourages the genetic algorithm to refine and simplify the strategy.

One way to think about this is to compare it with fuzzing. In fuzzing we generate entirely random crap -- within the limits of the target API's parameters -- and see what happens.

Geneva is a bit like that, but with memory and much more goal-direction choosing successive strategies. And unlike previous work of this sort, censorship evasion strategies are not manually created. Geneva discovers them automatically. Once Geneva has discovered a means of evading censorship, only then is that means analyzed manually to learn more about how the censor operates.

In this case the work has led to interesting insights, like how China's great firewall synchronizes its state or how it processes packets. Sometimes the analysis concludes that Geneva has discovered a bug in the censor. An example is the Segmentation species Geneva discovered in China against the Great Firewall by segmenting the forbidden request twice at specific indices. Though the Great Firewall can normally reassemble segments, by segmenting twice at these specific bounds, the somewhat-less-than-Great Firewall can be tricked into ignoring the request to evade censorship.

As it turns out, this actually works. Geneva has been deployed against real-world censors in China, India, Iran, and Kazahkstan. It has independently discovered dozens of strategies to defeat censorship, and found previously unknown bugs in censors. And what's more, all of these strategies, and Geneva's strategy engine are open source on Github, and four days ago the ESNI plug-in was added to the repository: https://github.com/kkevsterrr/geneva

In the case of China's ESNI-triggered blocking, they trained Geneva over the span of 48 hours, both client- and server-side. They discovered 6 strategies to defeat the ESNI censorship: 4 that work from the server, and 6 that work from the client. And they are wacky stuff that wouldn't ever occur to someone.

For example, Strategy 1 is the "Triple SYN"

It operates by initiating the TCP 3-way handshake with **three** SYN packets, such that the sequence number of the third SYN is corrupted. Who would ever think of trying that? This strategy performs a "desynchronization attack" against the Great Firewall. The GFW synchronizes on the corrupt sequence number, so it misses the ESNI request. It can also be applied on the server-side. This means that citizens inside China could successfully connect to secure and private services outside China by adding this wacky triple-SYN hack to the server. And it might only do this for client IPs originating from China.

An example of another strategy is "Four Byte Segmentation"

The client sends the ESNI request across two TCP segments, such that the first TCP segment is less than or equal to 4 bytes long. This is not the first time Geneva has discovered segmentation strategies, but it's surprising that this strategy works in China. The Great Firewall has been famous for its ability to reassemble TCP segments for almost a decade. The TLS header is 5 bytes long, so by segmenting specifically the TLS header across multiple packets, we hypothesize this breaks the GFW's ability to protocol fingerprint ESNI packet as TLS. This has interesting implications for how the GFW fingerprints connections: it suggests the component of the GFW that performs connection fingerprinting cannot reassemble TCP segments for all protocols. This theory is supported by other segmentation-based strategies identified by Geneva in the past.

And, again, this strategy can also be triggered server-side because by reducing the TCP window size during the 3-way handshake, a server can force the client to segment their request.

And it goes on like that. I have a link documenting the entire suite of censorship bypass strategies in the show notes for anyone who's interested:
https://geneva.cs.umd.edu/posts/china-censors-esni/esni/

And, yes, it's true that this is a cat and mouse game like virus and anti-virus. China could and will eventually respond with fixes for the problems that Geneva has found... especially since the entire effort is open and public. Maybe then it will find other ways around the filtering.

But I think the larger implication here is for the application of this sort of AI'ish-directed discovery to software vulnerabilities. I drew the analogy earlier to fuzzing. We might call random fuzzing "dumb fuzzing" which begs the question... could we make fuzzing much smarter and more effective by adding a layer of goal-directed AI to the process. Which, I suppose, would bring us...

Super Fuzz!