Security Now! #814 - 04-13-21

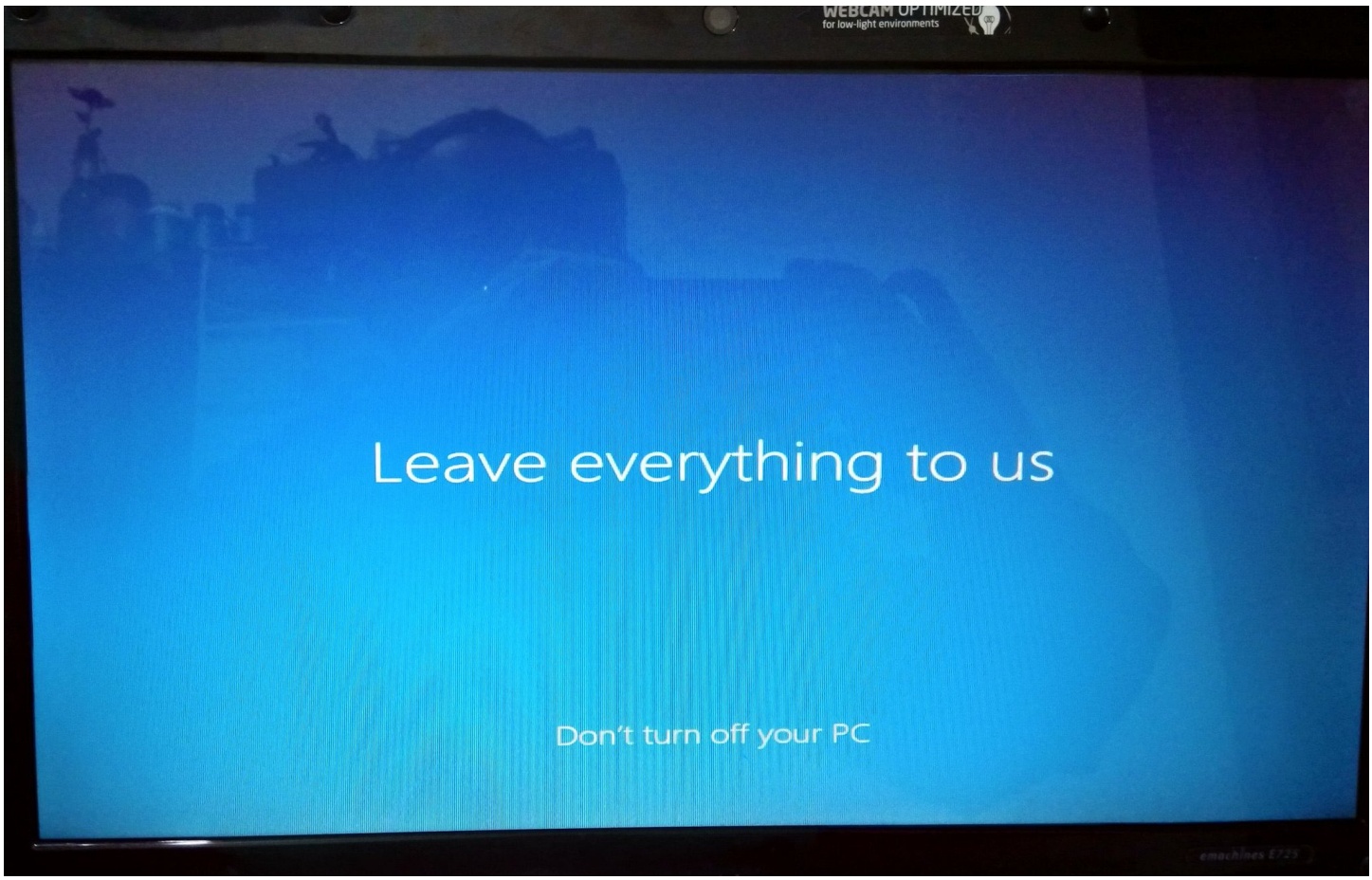
PwnIt And OwnIt

This week on Security Now!

This week we start with some needed revisiting of previous major topics. We look at an additional remote port that Chome will soon be blocking and the need to change server ports if you're using it. We look again at Google's forthcoming FLoC non-tracking technology and a new test page put up by the EFF. We revisit the PHP GIT server hack now that it's been fully understood. And we look at Cisco's eyebrow raising decision not to update some end-of-life routers having newly revealed critical vulnerabilities, and we also examine another instance of the industry's failure to patch — for years. Then, we conclude with a blow-by-blow, or hack-by-hack, walkthrough of last week's quite revealing and somewhat chilling Pwn2POwn competition.

The following is purely rhetorical, but...

What could possibly go wrong?



Browser News

The Slips keep Streaming

As we know, the practice known as "NAT Slipstreaming", which we've talked about twice recently, uses — or rather abuses — a user's local router's Application Layer Gateway (ALG) features. Through NAT Slipstreaming, code running on a browser inside the LAN is able to arrange for unsolicited traffic to enter the LAN, thus effectively bypassing the natural firewall features of any NAT router.

For this form of abuse to work, the browser needs to be able to emit traffic bound for specific remote ports. So browsers have been fighting back by preventing outbound connections to the abuse-prone ports which NAT routers monitor for Application Layer traffic. Google's Chrome browser blocks the use of the FTP, HTTP, and HTTPS protocols to a growing list of ports. Currently we have port 69, 137, 161, 554, 1719, 1720, 1723, 5060, 5061, and 6566.

We're talking about this once again because last Thursday Google stated that they intend to add TCP port 10080 to their growing list. This remote port has been on Firefox's block list since last November, 2020.

https://searchfox.org/mozilla-central/source/netwerk/base/nsIOService.cpp#98

Port 10080 is known to be used by the Amanda backup software and VMWare's vCenter. But in neither of those two cases is there any need for a web browser to use it. The biggest concern is that, much like port 8080 which is the port of choice for userland web servers, port 10080 has been used as a web server alternative. In fact, when I was researching this issue last night, I encountered this posting on Reddit:

> **Why Firefox Developer Edition blocks HTTP on port 10080 after update?**
> "My current school project uses port 10080 to run the web application. After the update to Firefox 85b2 (Dev) I need to set network.security.ports.banned.override in order to access my web. What is the reason behind the ban of this port? I am curious because it is not a standard port, Firefox allows it before, and it is not used by any widely known applications."

We've talked before about how tricky it can be to take back anything once it's been given. The Internet, the Web, and many descendent technologies were originally designed by technologists who placed very few — and in retrospect, too few — limitations on the use of their new toys. Look at how cautious Google was with Chrome's careful, gradual tip-toeing removal of FTP support. They very much wanted to remove it, but they were also worried about taking something away that users might be depending upon.

Over in the Bugzilla forum, I found:

> "I have a similar problem with this change. I have thousands of CPE (customer premise equipment) with their management port on 10080. Of course not accessible from the internet. But I need to manage them. Is there any option in about:config that will allow me to use port 10080 again?"

This was referring to the change in Firefox late last year.

And the reply:

> Sorry for the inconvenience this is causing.
> - Go to about:config
> - Create a new pref of type String with the name network.security.ports.banned.override
> - Add the required ports as the preference value (you can also add multiple as a comma-separated list or as ranges).
>
> This change does not require a restart.

On Chrome's decision, Chrome developer Adam Rice noted of port 10080 that: "It is an attractive port for HTTP because it ends in "80" and does not require root privileges to bind on Unix systems." To allow developers to continue using this port, Rice said that they'll be adding an enterprise policy that developers can use to override the block.

Once Chrome's change is in place, users will receive an "ERR_UNSAFE_PORT" message whenever Chrome attempts to access remote port 10080. The takeaway for our listeners is that it would probably be a good idea to migrate any services you might be hosting on port 10080, which need to be accessed from web browsers, to some unblocked port.

Are You FLoC'ed?

This podcast's title three weeks ago was "What the FLoC?" And now the EFF has put up a test site which can be used to determine one's "FLoC-age" with the title "Am I FLoCed?"

https://amifloced.org/

We already know from our coverage three weeks ago that the best way to characterize the EFF's position is that ANY form of bias in the treatment of users of the Internet — no matter the means — is a fundamentally bad idea because it discriminates among Internet users who, in the EFF's view, should all be treated identically.

As we know, Google's FLoC — Federated Learning of Cohorts — creates a temporary & transient tag formed from hashing the websites Chrome has visited during the previous week. So, even though it is expressly not tracking, and even though Google plans to terminate all 3rd-party cookie support in Chrome in their transition to FLoC, the EFF will be satisfied with nothing less than an entirely non-customized experience on the Internet.

So, they have created the "Am I FLoCed" a page to test and display whether you and your instance of Chrome may have been randomly chosen to participate in Google's initial FLoC research — which Google refers to as an "origin trial." The EFF's page opens with the headline: "Google is testing FLoC on Chrome users worldwide. Find out if you're one of them."

> Google is running a Chrome "origin trial" to test out an experimental new tracking feature called Federated Learning of Cohorts (aka "FLoC"). According to Google, the trial currently affects 0.5% of users in selected regions, including Australia, Brazil, Canada, India, Indonesia, Japan, Mexico, New Zealand, the Philippines, and the United States. This page will try to detect whether you've been made a guinea pig in Google's ad-tech experiment.

Okay. 0.5% is 1 in every 200 Chrome users and assuming a statistically neutral assignment, we will surely have many listeners among us. I checked by two Chrome instances at my two locations and came up negative at each.

What is FLoC?
Third-party cookies are the technology that powers much of the surveillance-advertising business today. But cookies are on their way out, and Google is trying to design a way for advertisers to keep targeting users based on their web browsing once cookies are gone. It's come up with FLoC.

FLoC runs in your browser. It uses your browsing history from the past week to assign you to a group with other "similar" people around the world. Each group receives a label, called a FLoC ID, which is supposed to capture meaningful information about your habits and interests. FLoC then displays this label to everyone you interact with on the web. This makes it easier to identify you with browser fingerprinting, and it gives trackers a head start on profiling you. You can read EFF's analysis and criticisms of FLoC here.

The Chrome origin trial for FLoC has been deployed to millions of random Chrome users without warning, much less consent. While FLoC is eventually intended to replace tracking cookies, during the trial, it will give trackers access to even more information about subjects. The origin trial is likely to continue into July 2021, and may eventually affect as many as 5% of Chrome users worldwide. See our blog post about the trial for more information.

How can I opt out?
For now, the only way for users to opt out of the FLoC trial in Chrome is by disabling third-party cookies. This may reset your preferences on some sites and break features like single sign-on. You can also use a different browser. Other browsers, including independent platforms like Firefox as well as Chromium-based browsers like Microsoft Edge and Brave, do not currently have FLoC enabled.

If you are a website owner, your site will automatically be included in FLoC calculations if it accesses the FLoC API or if Chrome detects that it serves ads. You can opt out of this calculation by sending the following HTTP response header:

Permissions-Policy: interest-cohort=()

What does my FLoC ID mean?
If you have been assigned a FLoC ID, it means that your browser has processed your browsing history and assigned you to a group of “a few thousand” similar users. The FLoC ID is the label for your behavioral group. This numeric label is not meaningful on its own. However, large advertisers (like Google) and websites (like… Google) will be able to analyze traffic from millions of users to figure out what the members of a particular FLoC have in common. Those actors may use your FLoC ID to infer your interests, demographics, or past behavior.

To get more technical: your browser uses an algorithm called SimHash to calculate your FLoC ID. The system currently uses the list of domains you’ve visited in the past 7 days as input, and recalculates the FLoC ID once a week. The current version of the trial places each user

into one of over 33,000 behavioral groups. You can view the code for the FLoC component here. Google has said that it intends to experiment with different grouping algorithms, and different parameters, throughout the trial.

Why does this matter?
FLoC exists because Google acknowledges the privacy harms of third-party cookies, but insists on continuing to let advertisers target you based on how you browse the web. We are happy Google will finally restrict third-party cookies in Chrome, but the last thing it should do is introduce new tracking technology. FLoC has privacy problems of its own, and it will likely continue to enable discrimination and other harms of targeted ads.

EFF believes browser developers should focus on providing a private, user-friendly experience without catering to the interests of behavioral advertisers. We should imagine a better future without the harms of targeted ads—and without Google's FLoC.

Leo and I indicated three weeks ago that this seems like a big improvement over the mess of 3rd-party tracking cookies, though I'm sure feelings and opinions will vary across the spectrum. But there is something factual that I really wish the EFF would drop. Being wrong about this weakens their case. They insist upon incorrectly referring to this, like cookies, using the term "tracking." But this is explicitly NOT tracking. In no way is the user tracked. If the EFF wishes to use a term with derogatory connotations, as I'm sure they do, then they would be entirely accurate to say "profiling." Because that's what it arranges to do. But it does so pretty innocuously.

A highly desirable — in fact required — feature of any traditional cryptographic hash is that tiny differences — even a difference of one bit — in hash inputs yields dramatically different hashed results. But the SimHash algorithm that Google plans to use deliberately produces similar hash results (thus "SimHash") for similar inputs. It's still a hash. But two inputs that may have similarities will result in similar hashed results. So we can think of this as a sort of hashed fuzzy profiling. And in the same way that a hash of a password tells you only if the same password is used again, but nothing about the password itself, the SimHash FLoC tag, being a hash of past sites visited, does not reveal anything about *which* sites were visited. That information doesn't survive the hashing. I suppose I'm a bit of a tech junkie, but I think it's cool.

In any event, I'd love to have our Chrome-using listeners go to https://amifloced.org/ and see whether they are unwittingly participating in Google's FLoC experiment.

Security News

The PHP GIT Hack, revisited
While we're doing follow-ups on past topics, the week after "What the FLoC?" was "GIT me some PHP!" about what I felt was the oh-so-deliberately-obvious hack of the PHP project's private GIT server. The middle of last week we received an update on their investigation of what happened.

https://news-web.php.net/php.internals/113981

Nikita Popov posted “Update on git.php.net incident.” His posting is long and detailed and its link is here in the show notes for anyone who's curious. But the short version is that some legacy crap bit them in the butt. They had a seldom used secondary and less secure means for pushing commits into their private GIT repository. Nikita, who is very much on top of this wasn't even aware of this secondary back-channel at the time.

> When the first malicious commit was made under Rasmus' name, my initial reaction was to revert the change and revoke commit access for Rasmus' account, on the assumption that this was an individual account compromise. In hindsight, this action didn't really make sense, because there was (at the time) no reason to believe that the push occurred through Rasmus' account in particular. Any account with access to the php-src repository could have performed the push under a false name.
>
> When the second malicious commit was made under my own name, I reviewed the logs of our gitolite installation, in order to determine which account was actually used to perform the push. However, while all adjacent commits were accounted for, no git-receive-pack entries for the two malicious commits were present, which means that these two commits bypassed the gitolite infrastructure entirely. This was interpreted as likely evidence of a server compromise.
>
> Something I was not aware of at the time is that git.php.net (intentionally) supported pushing changes not only via SSH (using the gitolite infrastructure and public key cryptography), but also via HTTPS. The latter did not use gitolite, and instead used git-http-backend behind Apache2 Digest authentication against the master.php.net user database. I'm not sure why password-based authentication was supported in the first place, as it is much less secure than pubkey authentication.

Nikita then shows us a chunk of Apache HTTP server log showing the two commits, and notes:

> It is notable that the attacker only makes a few guesses at usernames, and successfully authenticates once the correct username has been found. While we don't have any specific evidence for this, a possible explanation is that the user database of master.php.net has been leaked, although it is unclear why the attacker would need to guess usernames in that case.
>
> The master.php.net system, which is used for authentication and various management tasks, was running very old code on a very old operating system / PHP version, so some kind of vulnerability would not be terribly surprising. We have made a number of changes to increase the security of this system:
>
> - master.php.net was migrated to a new system (running PHP 8) and renamed to main.php.net at the same time. Among other things, the new system supports TLS 1.2, which means you should no longer see TLS version warnings when accessing this site.
> - The implementation has been moved towards using parameterized queries, to be more confident that SQL injections cannot occur.
> - Passwords are now stored using bcrypt.
> - Existing passwords were reset (use main.php.net/forgot.php to generate a new one).

So... right in the middle of the home of PHP we have a very old server on a very old platform running a very old PHP that no one has looked at, or was really even aware of, for quite a long time. You all know me. "Old" doesn't automatically mean bad. Plenty of old things were written well and have stood the test of time. But big complex operating systems, feature laden web servers and PHP are not generally among those things.

So, now we have the point of entry. I still think of the attacker as more of a prankster, due to the "crying out loud" code he placed onto the server which demanded to be noticed. He somehow arranged to authenticate as two of the highest profile developers — again, making sure that his changes would be caught. And they never really did determine exactly how. And they didn't really care. They just got rid of it all, replacing it with up-to-date solutions to do the same thing.

And they did also move the PHP repository over to GitHub so that they can focus upon PHP development and not worry about the security of the access controls of the repository.

CISCO abandons old routers having problems.
A constant in our industry is the dilemma of deliberately terminating important critical patch support for previously supported systems that have reached the end of their support lifecycle. We often talk about this with Microsoft and Windows. It's especially irksome when some people are receiving paid-for updates while others are not. So it's not as if those updates do not exist. But even mighty Microsoft occasionally (we might say often) bends to the severity of their own mistakes to offer out-out-lifecycle patches when the cost of not doing so would be prohibitive. They just did this at the start of March by reaching way back to patch Exchange Sever 2010 for the ProxyLogon flaws even though it was well past its kill-by date.

But we've seen that in similar situations Cisco often makes a different call. And it's not as if Cisco's commercial products are not littered with patchable vulnerabilities. Indeed they are. Not to mention that spate of secret accounts and passwords that suddenly appeared throughout their products as people started poking around in their networking firmware.

Today, Cisco has informed the world that it has no plans to fix critical security vulnerabilities affecting some of its SOHO Small Business routers. What does it tell its past customers to do? Replace them with new Cisco devices. (Or perhaps it's time to consider changing brands?) The bug they share is tracked as CVE-2021-1459, and it carries a difficult-to-achieve CVSS score of 9.8 out of 10. (Just about the only way to get a higher score is if it's able to attack you when it's unplugged!) Four routers are affected, the RV110W VPN firewall and three Small Business routers, the RV130, RV130W, and RV215W routers.

In each of the four cases, the known flaw allows an unauthenticated, remote attacker to execute arbitrary code on the affected device. The flaw, which stems from improper validation of user-supplied input in the web-based management interface, could be exploited to send specially-crafted HTTP requests to the device to achieve remote code execution. Cisco's advisory said: "A successful exploit could allow the attacker to execute arbitrary code as the root user on the underlying operating system of the affected device." Cisco said: "The Cisco Small Business RV110W, RV130, RV130W, and RV215W Routers have entered the end-of-life process. Customers are encouraged to migrate to the Cisco Small Business RV132W, RV160, or RV160W Routers."

I looked them up. The routers are their lowest-end little plastic box consumer routers. The "W" suffix means "Wireless." The newer replacements, the RV160 and RV160W are $114 and $145 respectively on Amazon. So in any event, not a big investment. It's not as if they're leaving their major enterprise customers out to dry. And sadly, even if they were to update the firmware of those old and no-longer-being-made-or-supported routers, how many of them would even ever get the updated code?

The trouble is, there will be tons of those older routers out on the Net, thanklessly doing their job day in and day out. And presumably people chose the "Cisco" brand because they'd heard of Cisco and wanted the assurance of a superior product. Let's hope that the default configuration was to disable remote web access and that it was never turned on. This is another of those "we-see-them-too-often" web management interface issues. Hopefully, no one is exposing their router's web management interface to the Internet. And I'm sure that all of our listeners know that unless remote management is really needed and being actively used, web management — all remote management — should always be kept off of the public Internet.

One thing to note is that all of the tech press which covered Cisco's announcement of the trouble led with "Cisco says it will not patch three small business router models and one VPN firewall device with critical vulnerabilities." That was repeated over and over. So that appears to matter.

The one right way to handle the need for remote access is to turn off remote access everywhere, then use one high-quality SSH server that requires a password, a certificate, and a time based additional factor for authentication. And while you're at it, run that server on some random port. So you use SSH to securely gain access to the internal network, then perform any required administrative work from the inside over LAN interfaces. It's even possible to run a standard Windows Remote Desktop connection over SSH. It works great.

Failure to Patch

Way back in early 2019, FortiNet, who produces the FortiGate SSL VPN, received notification through responsible disclosure of a critical remotely exploitable vulnerability in several current releases of their FortiOS which forms the basis for several products. Security researchers Meh Chang and Orange Tsai from the DEVCORE Security Research Team discovered the trouble.

FortiNet found and immediately fixed the trouble which afflicted three branches of their FortiOS, 5.4, 5.6 and 6.0. And in May of 2019 they produced update patches and their FortiGuard Labs published a clear vulnerability disclosure explaining that: *"A path traversal vulnerability in the FortiOS SSL VPN web portal may allow an unauthenticated attacker to download FortiOS system files through specially crafted HTTP resource requests."*

Three months later, on August 28th, FortiNet posted a blog titled *"FortiOS and SSL Vulnerabilities"* which explained that: *"At the recent Black Hat 2019 conference held in Las Vegas this past August 3-8, security researchers discussed their discovery of security vulnerabilities that impacted several security vendors, including Fortinet. All of the vulnerabilities impacting Fortinet were fixed in April and May of 2019."*

But in this disclosure we get another little interesting tidbit: *"In addition, it was also disclosed (and fixed) in May 2019 that FortiOS included a "magic" string value that had been previously created at the request of a customer to enable users to implement a password change process*

when said password was expiring. That function had been inadvertently bundled into the general FortiOS release, and an Improper Authorization vulnerability resulted in that value being usable on its own to remotely change the password of an SSL VPN web portal user without credentials." A few days before that, this had hit the tech press. For example, on August 24th, Dan Goodin, writing for ArsTechnica titled his story: *"Hackers are actively trying to steal passwords from two widely used VPNs"* and Dan opened with: *"Hackers are actively unleashing attacks that attempt to steal encryption keys, passwords, and other sensitive data from servers that have failed to apply critical fixes for two widely used virtual private network (VPN) products, researchers said.*

The vulnerabilities can be exploited by sending unpatched servers Web requests that contain a special sequence of characters, researchers at the Black Hat security conference in Las Vegas said earlier this month. The pre-authorization file-reading vulnerabilities resided in the Fortigate SSL VPN, installed on about 480,000 servers, and the competing Pulse Secure SSL VPN, installed on about 50,000 machines, researchers from Devcore Security Consulting reported."

480,000 FortiGate SSL VPN instances. Yikes. At the time, the Internet was being sprayed with probes seeking to find and exploit these now well-known weaknesses.

Then... nearly a year passes. On July 16th, 2020, FortiNet blogs with the title: *"ATP 29 Targeting SSL VPN Flaws"*

"United Kingdom's National Cyber Security Centre (NCSC) and Canada's Communications Security Establishment (CSE) have published research into the activity of 'APT29', also known as 'the Dukes' or 'Cozy Bear' who have been targeting various organisations involved in COVID-19 vaccine development in Canada, the United States and the United Kingdom, highly likely with the intention of stealing information and intellectual property relating to the development and testing of COVID-19 vaccines.

The initial attack vectors for this group have been unpatched vulnerabilities in SSL-VPN solutions from Fortinet. One of the vectors used included a vulnerability resolved by Fortinet in May 2019, allowed an unauthenticated attacker to download FortiOS system files through specially crafted HTTP resource requests as disclosed in FG-IR-18-384 / CVE-2018-13379. At the time of the disclosure Fortinet made available patches for all supported releases (5.4, 5.6, 6.0, 6.2).

Customers were notified at the time via the public PSIRT Advisory system of the need to upgrade immediately and highlighted the same in the release notes. For those unable to upgrade, mitigations were provided. For additional transparency, this was again highlighted in a blog in August 2019 after the vulnerabilities were disclosed by the researchers at Black Hat 2019."

And today, this nearly two year old issue is back in the news because Kaspersky Labs has just released their report analyzing a few high profile ransomware attacks, employing a relatively new ransomware strain called "Cring" which has been used to shut down some major European industrial enterprises. Kaspersky explained:

> "The attackers exploited the CVE-2018-13379 vulnerability to gain access to the enterprise's network. The vulnerability was used to extract the session file of the VPN Gateway. The

session file contains valuable information, such as the username and plaintext password.

Unpatched FortiGate devices are vulnerable to a directory traversal attack, which allows an attacker to access system files on the FortiGate SSL VPN appliance. Specifically, an unauthenticated attacker can connect to the appliance through the internet and remotely access the file "sslvpn_websession", which contains the username and password stored in cleartext. The vulnerability affects devices that run FortiOS versions 6.0.0 to 6.0.4, 5.6.3 to 5.6.7, and 5.4.6 to 5.4.12.

Several days before the start of the main attack phase, the attackers performed test connections to the VPN Gateway, apparently in order to check that the vulnerable version of the software was used on the device.

The attackers may have identified the vulnerable device themselves by scanning IP addresses. Alternatively, they may have bought a ready-made list containing IP addresses of vulnerable FortiGate VPN Gateway devices. In autumn 2020, an offer to buy a database of such devices appeared on a dark web forum.

After gaining access to the first system on the enterprise network, the attackers downloaded the Mimikatz utility to that system. The utility was used to steal the account credentials of Windows users who had previously logged in to the compromised system.

With the help of Mimikatz, the attackers were able to compromise the domain administrator account, after which they started distributing malware to other systems on the organization's network. They used the Cobalt Strike framework for that purpose. The Cobalt Strike module was loaded on attacked systems using PowerShell."

https://ics-cert.kaspersky.com/reports/2021/04/07/vulnerability-in-fortigate-vpn-servers-is-exploited-in-cring-ransomware-attacks/

Once upon a time, as we old timers vividly recall, our systems, whether Windows, Mac, Unix or Linux crashed with some regularity. They don't any longer. I'm really curious to see whether our use of networking and networking technologies follows the same path. Are we ever going to get it right? Can we? One of the advantages our operating systems have, is that everything is concentrated in one place. They're self contained monoliths of technology. And even so, at the margins, our operating systems are still not perfect.

But on the Internet, everyone rolls their own. And everyone wants to. So, as a consequence, we keep seeing the same mistakes being made over and over again. And we even see regressions where problems once fixed reappear. And lessons once learned, like "don't use secret strings in firmware" are forgotten or unlearned. Someone new comes along and thinks "I'm going to solve that problem this way", even when many others before have learned the hard way, not to.

On the Internet there's no central control. Which is often touted as being a good thing. It spurs and spawns innovation. I'm sure that's true. But innovation brings mistakes. The old adage "if you're not making mistakes you're not trying hard enough" is unfortunately all too true on the Internet. But, collectively, we really cannot afford to keep making these mistakes. They're becoming more and more expensive. And there are undoubtedly many that have not yet been found.

In **THIS** instance, it's certainly the case that many of the FortiGate SSL VPN gateways were quickly patched by IT professionals who were on their game. They received and read the news of an update and quickly patched their enterprise's servers — even if it may have been briefly inconvenient to some users — to keep them safe. But we know that not everyone received the news. Or if they did, they receive so much of it on a daily basis that they put it into the "I'll deal with this later" pile and never did.

One thing we've seen is that there are technologies that seem to be particularly troublesome. Aside from embedding secret strings in firmware, there are web interfaces. They're attractive because they're so user-friendly. But they're also inherently attacker-friendly because their user-friendliness comes by way of complexity. And complexity is at eternal war with security.

It appears that the best we can do is design our solutions to minimize our attack surfaces. Like by using one high-security SSH server to provide ALL remote access by moving all other remote access to the inside. And of course, as I've often mentioned, being careful to keep all lines of communication open to the vendors of our products and to carefully consider the implications of every security update notice.

PwnIt And OwnIt

Last week, Tuesday through Thursday, $1,210,000 dollars flowed from deep pocketed sponsors to very clever security researchers during the Spring 2021 Pwn2Own contest. The blow by blow details are as interesting as ever, and we'll get to those in a second. But, overall, 23 teams of researchers targeted web browsers, virtualization offerings, servers, and enterprise communications. The total prize pool, most which was distributed, was $1.5 million dollars in cash and a Tesla Model 3. None of the teams chose to tackle the Tesla this year, but, as we'll see in a minute, they pretty well minced up Windows 10, Microsoft's Teams, Exchange Server, Ubuntu, Chrome, Edge, Safari, Parallels Desktop.

In addition to cash, which remains kings, point scores are also awarded for successful hacks and the top three teams all tied at 20 points each. Here's what happened:

Tuesday, April 6

1000 - Jack Dates from RET2 Systems targeted Apple Safari in the Web Browser category

SUCCESS - Jack used an integer overflow in Safari and an OOB Write to get kernel-level code execution. In doing so, he wins $100,000 and 10 Master of Pwn points.

1130 - DEVCORE targets Microsoft Exchange in the Server category

Recall that it was DEVCORE who originally discovered and reported those authentication bypass bugs in Exchange Server that led to this year's devastating ProxyLogon attacks when Microsoft appears to have badly underestimated their impact and took their time releasing patches.

SUCCESS - Think all of the problems with Exchange Server have been resolved? Think again! The DEVCORE team combined an authentication bypass and a local privilege escalation to completely take over Exchange server to earn $200,000 and 20 Master of Pwn points.

And this time it's not difficult to imagine that Microsoft will be patching Exchange Server with some alacrity. But since today is April's Patch Tuesday, those fixes may not have been ready in time. Will we see an out-of cycle update?

1300 - The researcher who goes by OV targeted Microsoft Teams in the Enterprise Communications category

SUCCESS - OV combined a pair of bugs to demonstrate code execution on Microsoft Teams. In doing so, he earned himself $200,000 and 20 points towards Master of Pwn

1430 - Team Viettel took aim at Windows 10 in the Local Escalation of Privilege category

SUCCESS - And they, too, succeeded with an integer overflow in Windows 10 to escalate their privilege from regular user to SYSTEM privileges. This earns them $40,000 and 4 points towards Master of Pwn.

1530 - The STAR Labs team of Billy, Calvin and Ramdhan targeted Parallels Desktop in the Virtualization category... and this was the first failure of the first day.

FAILURE - The STAR Labs team could not get their exploit to work within the time allotted.

1630 - Ryota Shiga of Flatt Security Inc targeted a fully patched and up-to-date Ubuntu Desktop hoping to achieve an Escalation of local Privilege

SUCCESS - Ryota used an OOB access bug to elevate himself from a standard user to root on Ubuntu Desktop and earned himself a tidy $30,000 and 3 Master of Pwn points in his Pwn2Own debut.

1730 - Undaunted by their inability to penetrate the Parallel's Desktop previously, the STAR Labs team of Billy, Calvin and Ramdhan went after Oracle's VirtualBox in the Virtualization category.

FAILURE - And, unfortunately, they were again unable to accomplish their penetration within the allotted time.

Wednesday, April 7

0900 - Jack Dates from RET2 Systems kicked off the day again, as he had Tuesday, this time targeting the Parallels Desktop...

SUCCESS - Jack nailed it by combining three bugs - an uninitialized memory leak, a stack overflow, and an integer overflow to escape from the Parallels Desktop and execute code directly on the underlying OS. Jack added $40,000 to the $100,000 from the day before and racked up an additional 4 Master of Pwn points.

1000 - Bruno Keith (@bkth_) & Niklas Baumstark (@_niklasb) of Dataflow Security (@dfsec_it) targeting Google Chrome and Microsoft Edge (Chromium) in the Web Browser category

SUCCESS - They successfully employed a Type Mismatch bug to exploit the rendering engines in both Chrome and Microsoft Edge — Same exploit for *both* browsers — because, of course, for better or for worse, we're heading toward a web browser monoculture. They earned $100,000 total and 10 Master of Pwn points.

1130 - Team Viettel targeting Microsoft Exchange Server and scored a partial success only due to previous disclosure.

PARTIAL - They successfully demonstrated their code execution on the Exchange server, but some of the bugs they used in their exploit chain had been previously reported in the contest. This counts as a partial win but did award them 7.5 Master of Pwn points.

1300 - Daan Keuper and Thijs Alkemade from Computest targeted Zoom Messenger in the Enterprise Communications category... and this one made some news.

SUCCESS - They successfully used a three bug chain to exploit Zoom messenger and get code execution on the target system - whoopsie! - and this was without the target clicking anything. That got them $200,000, 20 Master of Pwn points, and I'm sure a follow-up conversation with Zoom!

1430 - Tao Yan (@Ga1ois) of Palo Alto Networks went after Windows 10 in the Local Escalation of Privilege category

SUCCESS - By using a Race Condition bug, Tao was able to successfully escalate his access to full SYSTEM privilege on the fully patched Windows 10 machine. He earned $40,000 and 4 points towards Master of Pwn.

1530 - Sunjoo Park (aka grigoritchy) also targeted Parallels Desktop.

SUCCESS - And, sure enough, by using a logic bug in Parallels, he was able to execute code on the underlying operating system. That earned him $40,000 and 4 points towards Master of Pwn.

1630 - Manfred Paul also targeted Ubuntu Desktop for Local Escalation of Privilege.

SUCCESS - By using an OOB Access bug he succeeded in escalating to Root Ubuntu Desktop. So he's now $30,000 richer and scored himself 3 points towards Master of Pwn.

1730 - The researcher known as z3r09 targeted Windows 10 for with a Local Escalation of Privilege

SUCCESS - By successfully using an integer overflow, z3r09 escalated his permissions to NT Authority\SYSTEM which simultaneously escalated his bank account by $40,000 and his Master of Pwnage level by 4.

Thursday, April 8

0900 - Benjamin McBride from L3Harris Trenchant also targeted Parallels Desktop.

SUCCESS - And Ben employed a memory corruption bug to successfully execute code on the host OS from within Parallels Desktop earning himself $40,000 and 4 Master of Pwn points.

1000 - Steven Seeley of Source Incite thought that Microsoft Exchange probably still presented some low hanging fruit.

PARTIAL - Although Steven did successfully use two unique bugs in his demonstration, he was only credited with a partial win because his attack required a Man-in-the-Middle aspect. But the great research earned him 7.5 Master of Pwn points.

1130 - Billy, of the he STAR Labs team also targeted Ubuntu Desktop.

PARTIAL - Although Billy was able to successfully escalate his privileges to Root, the bug he used was already known to the vendor and on their patch list. So the demo gets Billy 2 additional Master of Pwn points.

1230 - Fabien Perigaud of Synacktiv targeted Win10 for a Local Escalation of Privilege

PARTIAL - And despite Fabien's reported excellent use of ASCII art during his demonstration, it turns out Microsoft was aware of the bug he used. So he still earns 2 Master of Pwn points for the partial win.

1330 - Alisa Esage went after a Parallels Desktop penetration.

PARTIAL - Despite her great demonstration (replete with ASCII art), the bug used by Alisa had been reported to ZDI prior to the contest which reduced it to a partial win. The judges commented that it was great work, and were thrilled she broke ground as the 1st woman to participate as an independent researcher in Pwn2Own history. She took home a pair of Master of Pwn points.

1430 - Vincent Dehors of Synacktiv targeted Ubuntu Desktop for Local Escalation of Privilege.

SUCCESS - And despite Vincent's admission that this was the first exploit he had written for Linux, he had no issues escalating to root through a double free bug, thus earning himself $30,000 and 3 Master of Pwn points.

1530 - Da Lao took aim at Parallels Desktop.

SUCCESS - And using an OOB Write, he, too, to successfully completed a guest-to-host escape in Parallels and earned himself $40,000 and 4 points towards Master of Pwn.

1630 - And last but not least, Marcin Wiazowski targeted Win10 for a Local Escalation of Privilege.

SUCCESS - And Marcin nailed it using a Use After Free (UAF) flaw to escalate his privilege to SYSTEM on Win10... taking home $40,000 and 4 Master of Pwn points.

So... a bunch of high-profile companies have some high-profile patching to do.

And every Pwn2Own, where we watch talented research hackers appear to so easily find and exploit previously unknown flaws — those are all true 0-day exploits when they are demonstrated — always gives me a bit of a chill. It really begs the question, "just how much other unknown stuff lies out there, either undiscovered or previously discovered and being put to quiet use?"

The newly revealed Zoom vulnerabilities, demonstrated by the team from Computest Security, are particularly noteworthy because they require no interaction of the victim other than being a participant on a Zoom call. What's more, it affects both Windows and Mac versions of the app, though it's not clear if Android and iOS versions are also vulnerable.

Fortunately, details of the flaws have not been disclosed, but in a statement sharing the findings, Computest Security, the company said the researchers "were then able to almost completely take over the system and perform actions such as turning on the camera, turning on the microphone, reading emails, checking the screen and downloading the browser history."

For its part, Zoom said it has already pushed a server-side change to patch the bugs, and noted that it's also working on incorporating extra protections to resolve some security shortcomings.

A Zoom spokesperson said: "On April 9th, we released a server-side update that defends against the attack demonstrated at Pwn2Own on Zoom Chat. This update does not require any action by our users. We are continuing to work on additional mitigations to fully address the underlying issues." So... that sounds like they pushed an immediate short-term but incomplete fix to block the explicit attack that was demonstrated, and that it revealed the need for some deeper re-engineering.

Zoom said it's not aware of any evidence of active exploitation by these issues, while pointing out the flaws don't impact in-session chat in Zoom Meetings. They said that the "attack can only be executed by an external contact that the target has previously accepted or be a part of the target's same organizational account."

In any event, Pwn2Own is clearly providing a valuable service... and I hope that, for what it's worth, those in charge of security everywhere receive the chill they deserve when they watch what talented security hackers are able to do with some of the industry's most secure offerings.