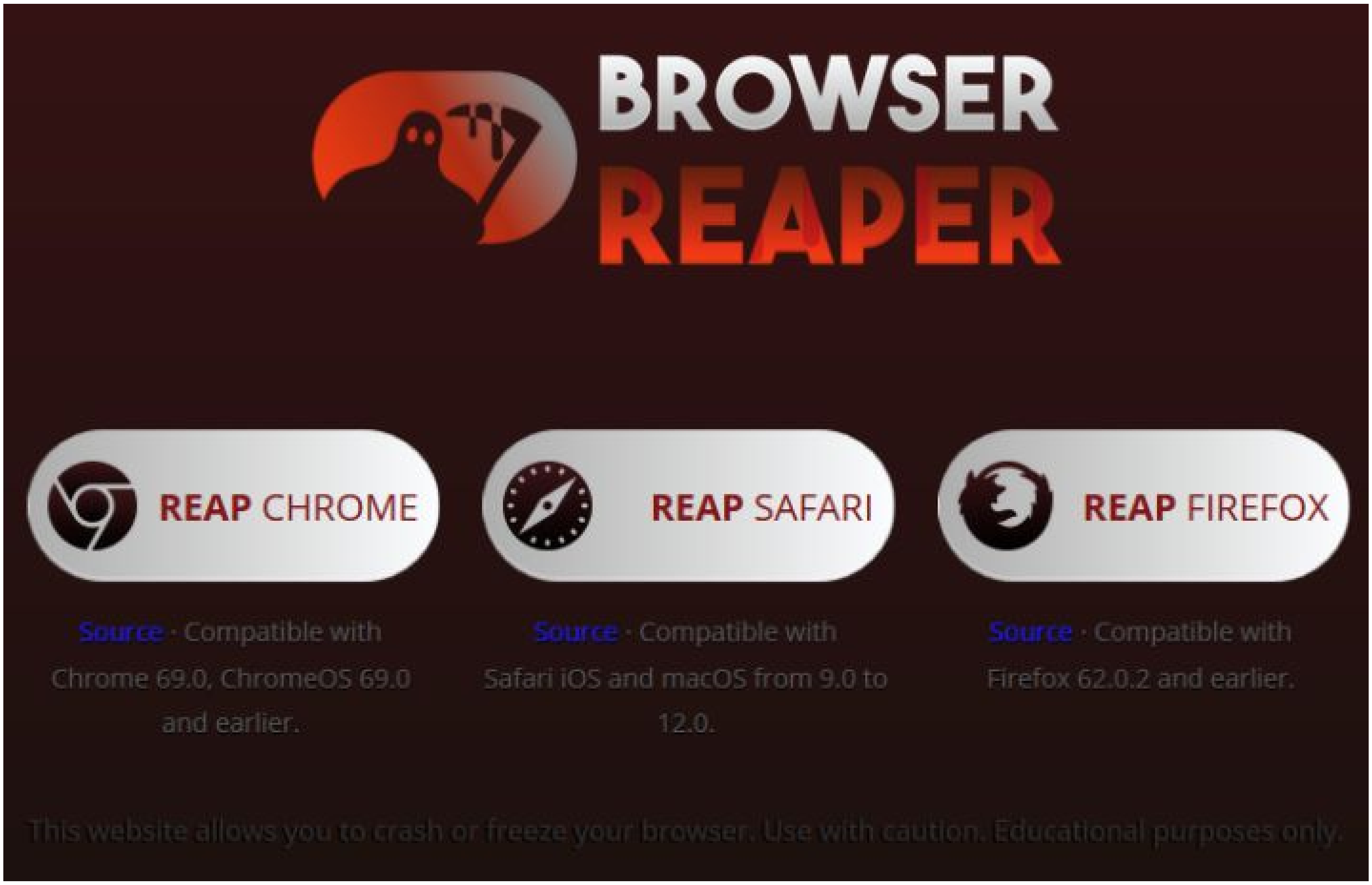
Security Now! #682 - 09-25-18

SNI Encryption

This week on Security Now!

This week we look at additional changes coming from Google's Chromium team, another powerful instance of newer cross-platform malware, the publication of a 0-day exploit after Microsoft missed its deadline, the return of Sabri Haddouche with browser crash attacks, the reasoning behind Matthew Green's decision to abandon Chrome after a change in release 69... and an "UnGoogled" Chromium alternative that Matthew might approve of, Western Digital's pathetic response to a very serious vulnerability, a cool device exploit collection website, a question about the future of the Internet, a sobering example of the aftermarket in unwiped hard drives, the Mirai Botnet creators are now working with and helping the FBI, another fine levied against Equifax, and a look at Cloudflare's quick move to encrypt a remaining piece of web metadata.

First we crash… then we burrow inside...



Source · Compatible with Chrome 69.0, ChromeOS 69.0 and earlier.

Source · Compatible with Safari iOS and macOS from 9.0 to 12.0.

Source · Compatible with Firefox 62.0.2 and earlier.

This website allows you to crash or freeze your browser. Use with caution. Educational purposes only.

https://www.reaperbugs.com/index

Security News

Google continues messing with Chrome's UI.
First the 'www' and 'm' went away. Then they came back. Now we're told that 'www' is going away again while 'm' is remaining... For now.

Next, 'file://' for local files is going away and a special new visual flag, which Google terms a 'chip' is going to indicate that a non-HTTP file is being viewed. So with Chrome 70, we'll see the indication of “File” to the area to the left of the URL and just a clean drive/directory/pathname without the file:// scheme that all browsers have, until this been consistently displaying.

Google is also messing with ‘www’ on their mainstream search results.
https://www.bleepingcomputer.com/news/google/google-testing-removal-of-www-subdomain-from-search-results/
Lawrence Abrams for Bleeping Computer:

<quote>Google really wants to get rid of the WWW subdomain. First we had Google removing WWW in the Chrome 69 address bar and now there is some test underway to remove it from search results as well. I was first alerted to this when one of BleepingComputer's reporters noticed that the BleepingComputer domain was showing up in Google search results as https://bleepingcomputer.com.

When I checked from my end, though, it was showing it listed as normal with https://www.bleepingcomputer.com. While researching this behavior, I found many domains where Google was removing the WWW subdomain in the search results. Once I performed a refresh of the page, though, the normal www subdomain would be listed again. In some cases, I could refresh over and over and the results would switch back and forth between www and non-www. Ultimately, I could not get BleepingComputer.com to show the non-www version, so I found another site that was also performing this behavior. When I searched for "paloaltonetworks", it showed the domain listing without the www subdomain. If you clicked on the search result, the site would perform a 301 redirect to www.paloaltonetworks.com, which is the site's desired behavior. On a refresh of the search result page, the normal www version of the URL appeared again in the search results. This time, though, the site links have been changed to a smaller display under the domain description.

So what does this mean? Unfortunately, I have no idea and have not received a response back from Google at the time of this publication. Google, though, does have something against the WWW subdomain lately and feels that the www is unimportant and should be considered a trivial subdomain. This is because almost all web sites who have a site configured on a domain also have the same same configured on the www variant. Therefore, to Google, www is just not important.

And speaking of Palo Alto Networks, they have identified a new multifarious malware strain...

XBash Malware Deletes Databases on Linux, Mines for Coins on Windows
https://thehackernews.com/2018/09/ransomware-coinmining-botnet.html
https://www.bleepingcomputer.com/news/security/xbash-malware-deletes-databases-on-linux-mines-for-coins-on-windows/

XBash is written in Python, it's a self-propagating worm which targets both Windows and Linux systems. It's a Botnet. It's a cryptocurrency miner and a ransomware spoof that deletes a victim's databases and demands payment while being unable to restore the destroyed data.

It was found in the wild by Palo Alto Networks and has been tracked back to Chinese-speaking advanced persistent threat actors known for previous cyber attacks utilizing similar attack mechanisms.

Palo Alto Networks indicated that XBash also contains nascent functionality that could allow the malware to spread quickly within an organization's network.

XBash hunts for vulnerable or unprotected web services and deletes the data from many popular databases including MySQL, Maria, Couch, and Mongo running on Linux servers. XBash scans for services on a target IP, on both TCP and UDP ports such as HTTP, VNC, common database ports, Telnet, FTP, RDP, ElasticSearch, Rlogin and others. Upon finding a listening service the malware uses weak a username and password dictionary attack to brute force itself into the vulnerable service, and once in, deletes all the databases then displays a ransom note.

Palo Alto Networks notes that the malware itself does not contain any functionality that would allow the recovery of the deleted databases once a ransom amount has been paid by the victims.

XBash is known to have infected at least 48 victims, who have already paid ransom totally about $6,000 for the cybercriminals behind the threat. However, the researchers have seen no evidence that the ransom payments resulted in the recovery of any data for the victims.

Whereas the malware has capabilities to enlist targeted Linux-based systems into a botnet, at least for the time being, on Windows machines it only conducts cryptocurrency mining and self-propagation. For self-propagation, it exploits three known vulnerabilities in Hadoop, Redis, and ActiveMQ:

- Hadoop YARN ResourceManager unauthenticated command execution bug disclosed in October 2016 and has no CVE number assigned.
- Redis arbitrary file writes, and remote command execution vulnerability disclosed in October 2015 with no CVE number assigned.
- ActiveMQ arbitrary file write vulnerability (CVE-2016-3088), disclosed in earlier 2016.

If the entry point is a vulnerable Redis service, Xbash will send malicious JavaScript or VBScript payload for downloading and executing a coinminer for Windows instead of its botnet and ransomware module.

Developed in Python, the PyInstaller system converts the malware into standalone executable binaries for multiple platforms, including Windows, Apple macOS, and Linux.

This enables XBash to be widely cross-platform malware though, at the time of their report, Palo Alto Networks researchers had only found samples for Linux and had not encountered any Windows or macOS versions of Xbash.

So what's our takeaway from this?: NO services should EVER be publicly exposed unless they are being actively used, and, if they are -- since they are almost certainly going to be connected to by other non-human client programs, their usernames and passwords should be pure high-entropy gibberish such as they which can be obtained from grc.com's passwords page.

Headlines:
"0-Day Windows JET Database Vulnerability disclosed by Zero Day Initiative"
"Researcher Discloses New Zero-Day Affecting All Versions of Windows"
https://thehackernews.com/2018/09/windows-zero-day-vulnerability.html
https://www.bleepingcomputer.com/news/security/0day-windows-jet-database-vulnerability-disclosed-by-zero-day-initiative/

Trend Micro: "(0Day) Microsoft Windows Jet Database Engine Out-Of-Bounds Write Remote Code Execution Vulnerability"
https://www.zerodayinitiative.com/advisories/ZDI-18-1075/

Back on the 8th of May this year, Trend Micro's Zero-Day Initiative responsibly disclosed an Out-Of-Bounds Write in Microsoft's Jet Engine database which, if exploited, would allow remote attackers to execute arbitrary code on all versions of Microsoft Windows.

However, user interaction =IS= required to exploit this vulnerability in that the target must visit a malicious page or open a malicious file. The flaw exists within the management of indexes in the Jet database engine such that data in a specially crafted data database file can trigger a write past the end of an allocated buffer. (Whoopsie! That's never good.) Consequently, an attacker can leverage this vulnerability to execute code under the context of the current process.

- 05/08/18 - ZDI reported the vulnerability to the vendor and the vendor acknowledged the report
- 05/14/18 - Microsoft replied that they successfully reproduced the issue ZDI reported
  .... *nearly four months slip past….*
- 09/09/18 - Microsoft reported "an issue" with the fix (whatever the heck that means) and that the fix might not make the September release
- 09/10/18 - ZDI cautioned potential 0-day
- 09/11/18 - Microsoft confirmed the fix did not make the build
- 09/12/18 - ZDI confirmed to the vendor the intention to 0-day on 09/20/18

And so now Trend Micro has published a full exploit proof of concept on Github:
https://github.com/thezdi/PoC/tree/master/ZDI-18-1075

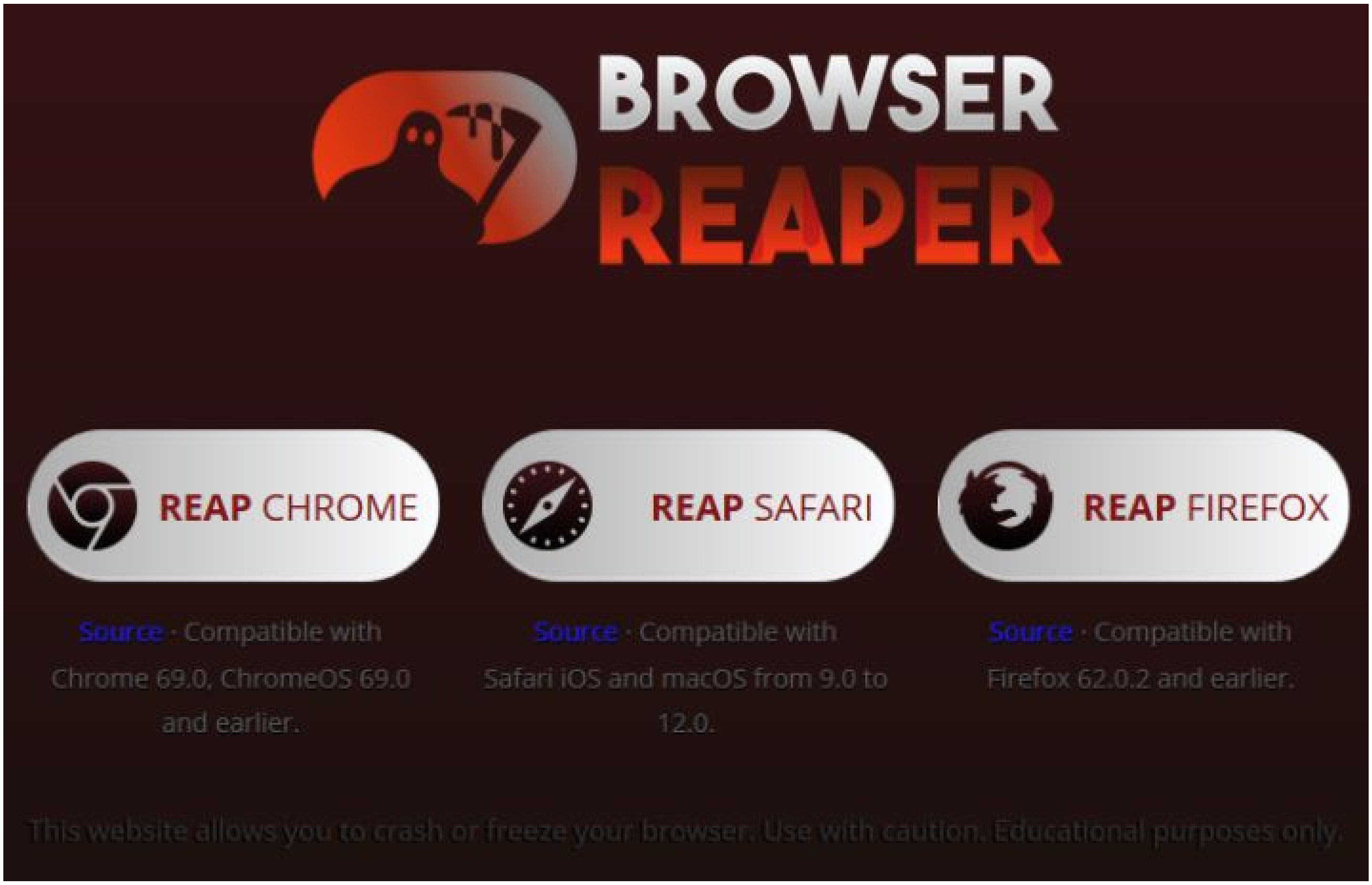
We can presumably expect to see a patch for this next month in October. In the mean time, Trend Micro posted that "Given the nature of the vulnerability the only mitigation strategy is to restrict interaction with the application to trusted files."

Sabri Haddouche is back… and crashing FireFox this time *(But… do we care?)*
https://www.reaperbugs.com/index

ZDNet Notes: *"Firefox bug crashes your browser and sometimes your PC. Bug affects Firefox on Mac, Linux, and Windows, but not Android."*

Last week we covered Sabri's interesting CSS exploit for iOS which took down by iOS v11.x and also the just-updated iOS v12.0. At the time Sabri explained that he had been searching for DoS bugs in popular web browsers when he happened to nuke both Safari's webkit-based browser and its underlying OS.

Now, at reaperbugs.com we have takedowns for Chrome, Safari and Firefox:



Source · Compatible with Chrome 69.0, ChromeOS 69.0 and earlier.

Source · Compatible with Safari iOS and macOS from 9.0 to 12.0.

Source · Compatible with Firefox 62.0.2 and earlier.

This website allows you to crash or freeze your browser. Use with caution. Educational purposes only.

In the case of the Firefox DoS, Sabri's code floodss the Inter-Process Communications channel which the main Firefox browser process uses to communicate with its child processes. This results in a browser freeze and ultimately a crash.

Examining the code Sabri posted at "Browser Reaper" it's clear that he's creating an insanely long filename and asking for it to be downloaded in a tight loop 1000 times per second. This floods the IPC communications and brings the browser to its knees.

Okay… so what's our takeaway about this one? (hint: <Yawn>)

Matthew Green: **“Why I’m done with Chrome”** *(September 23, 2018 -- 2,236 Words)*
Cryptographer and professor at Johns Hopkins University.

Matt starts off with: *“This blog is mainly reserved for cryptography, and I try to avoid filling it with random “someone is wrong on the Internet” posts. After all, that’s what Twitter is for! But from time to time something bothers me enough that I have to make an exception. Today I wanted to write specifically about Google Chrome, how much I’ve loved it in the past, and why — due to Chrome’s new user-unfriendly forced login policy — I won’t be using it going forward.”*

So… wait… what?

After the release of Chrome 69 users discovered that anytime they logged into their Google account, or any Google service, they would also be automatically logged into Chrome whether they wanted that or not.

The underlying feature is something the Chromium folks call “Identity consistency” between the web browser and the cookie jar. It is described: “When enabled (which it is by default), the browser manages signing in and out of Google accounts under Mac, Windows, Linux, ChromeOS and Android.

Matthew Green and others feel that this is a big deal since it associates a browser with a Google account, which, he argues, should never happen unless the user explicitly chooses to login to Chrome. Even if browsing data is not uploaded and sync is not enabled, there is data that could be gathered simply by the authentication process alone. And it’s true that, pursuant to Google’s own privacy policy, logging into the browser incurs a different set of privacy expectation:

> When you sign in to the Chrome browser or a Chromebook with your Google Account, your personal browsing data is saved on Google's servers and synced with your account. This type of information can include:
>
> - Browsing history
> - Bookmarks
> - Tabs
> - Passwords and Autofill information
> - Other browser settings, like installed extensions
>
> These settings are automatically loaded for you anytime you sign in to Chrome on other computers and devices. To customize the specific information that you synchronize, use the "Settings" menu.

The fact that Google has decided sign users into their browser without their permission causes Matthew to worry that Google may decide to start synchronizing user data whenever they choose. He wrote: “If you didn’t respect my lack of consent on the biggest user-facing privacy option in Chrome (and didn’t even notify me that you had stopped respecting it!) why should I trust any other consent option you give me? What stops you from changing your mind on that option in a few months, when we’ve all stopped paying attention?”

Matthew Green: *"A brief history of Chrome...*

When Google launched Chrome ten years ago, it seemed like one of those rare cases where everyone wins. In 2008, the browser market was dominated by Microsoft, a company with an ugly history of using browser dominance to crush their competitors. Worse, Microsoft was making noises about getting into the search business. This posed an existential threat to Google's internet properties.

In this setting, Chrome was a beautiful solution. Even if the browser never produced a scrap of revenue for Google, it served its purpose just by keeping the Internet open to Google's other products. As a benefit, the Internet community would receive a terrific open source browser with the best development team money could buy. This might be kind of sad for Mozilla (who have paid a high price due to Chrome) but overall it would be a good thing for Internet standards.

For many years this is exactly how things played out. Sure, Google offered an optional "sign in" feature for Chrome, which presumably vacuumed up your browsing data and shipped it off to Google, but that was an option. An option you could easily ignore. If you didn't take advantage of this option, Google's privacy policy was clear: your data would stay on your computer where it belonged.

A few weeks ago Google shipped an update to Chrome that fundamentally changes the sign-in experience. From now on, every time you log into a Google property (for example, Gmail), Chrome will automatically sign the browser into your Google account for you. It'll do this without asking, or even explicitly notifying you. (However, and this is important: Google developers claim this will <u>not</u> actually start synchronizing your data to Google — yet. See further below.) Your sole warning — in the event that you're looking for it — is that your Google profile picture will appear in the upper-right hand corner of the browser window. I noticed mine the other day.

The change hasn't gone entirely unnoticed: it received some vigorous discussion on sites like Hacker News. But the mainstream tech press seems to have ignored it completely. This is unfortunate — and I hope it changes — because this update has huge implications for Google and the future of Chrome.

In the rest of this post, I'm going to talk about why this matters. From my perspective, this comes down to basically four points:

1. *Nobody on the Chrome development team can provide a clear rationale for why this change was necessary, and the explanations they've given don't make any sense.*
2. *This change has enormous implications for user privacy and trust, and Google seems unable to grapple with this.*
3. *The change makes a hash out of Google's own privacy policies for Chrome.*
4. *Google needs to stop treating customer trust like it's a renewable resource, because they're screwing up badly.*

I warn you that this will get a bit ranty. Please read on anyway.

(This is where we step off…)

For those pedantic among us, Google's "Identity Consistency" CAN be disabled:

In Chrome (obviously), go to: **chrome://flags** Into "Search Flags" enter "account-con" which will whittle the flags down to the one you want: Identity consistency between browser and cookie jar. Disable it and restart your browser. It's logon will no longer be automatic.

Introducing: Ungoogled-Chromium / Bringing back the "Don't" in "Don't be evil"
https://github.com/Eloston/ungoogled-chromium

Ungoogled-chromium is Google Chromium, sans integration with Google. It also features some changes to enhance privacy, control, and transparency.

Motivation and Description

A number of features or background services communicate with Google servers despite the absence of an associated Google account or compiled-in Google API keys. Furthermore, the normal build process for Chromium involves running Google's own high-level commands that invoke many scripts and utilities, some of which download and use pre-built binaries provided by Google. Even the final build output includes some pre-built binaries. Fortunately, the source code is available for everything.

ungoogled-chromium is a set of configuration flags, patches, and custom scripts. These components altogether strive to accomplish the following:

- Disable or remove offending services and features that communicate with Google or weaken privacy
- Strip binaries from the source tree, and use those provided by the system or build them from source
- Disable features that inhibit control and transparency, and add or modify features that promote them (these changes are minor and do not have significant impacts on the general user experience)

ungoogled-chromium should not be considered a fork of Chromium. The main reason for this is that a fork is associated with more significant deviations from the Chromium, such as branding, configuration formats, file locations, and other interface changes. ungoogled-chromium will not modify the Chromium browser outside of the project's goals. Since these goals and requirements are not precise, unclear situations are discussed and decided on a case-by-case basis.

Prebuilt Binaries: https://ungoogled-software.github.io/ungoogled-chromium-binaries/

	Release	Development
Debian 9.0 (stretch) amd64	68.0.3440.106-1	69.0.3497.100-1
Portable Linux 64-bit	67.0.3396.87-2	69.0.3497.100-1
macOS 68.0.3440.106-1		
Ubuntu 18.04 (bionic) amd64	67.0.3396.87-2	69.0.3497.100-1
Windows 64-bit	67.0.3396.87-3	

Western Digital's My Cloud NAS Devices Turn Out to Be Easily Hacked

https://thehackernews.com/2018/09/wd-my-cloud-nas-hacking.html

[The hacker News] Western Digital's My Cloud (WD My Cloud) is one of the most popular network-attached storage (NAS) devices used by businesses and individuals to host their files, as well as backup and sync them with various cloud and web-based services. The WD My Cloud devices let users not only share files in a home network but its private cloud feature also allows them to access their data from anywhere around the world at any time.

However, security researchers at Securify discovered an authentication bypass vulnerability on the WD My Cloud NAS boxes that could allow unauthenticated attackers with network access to escalate their privileges to admin-level without needing to provide a password. This allow attackers to run commands that would typically require administrative privileges and gain complete control of the affected NAS device, including their ability to view, copy, delete and overwrite any files stored on the device.

Securify: <quote> Whenever an admin authenticates, a server-side session is created that is bound to the user's IP address. After the session is created it is possible to call authenticated CGI modules by sending the cookie *username=admin* in the HTTP request. The invoked CGI will check if a valid session is present and bound to the user's IP address.

It was found that it is possible for an unauthenticated attacker to create a valid session without being required to authenticate. The **network_mgr.cgi** CGI module contains a command called **cgi_get_ipv6** that starts an **admin session** that is tied to the IP address of the user making the request when invoked with the parameter **flag** equal to 1. Subsequent invocation of commands that would normally require admin privileges are now authorized if an attacker sets the **username=admin** cookie. Proof of Concept:

```
POST /cgi-bin/network_mgr.cgi HTTP/1.1
Host: wdmycloud.local
Content-Type:
application/x-www-form-urlencoded
Cookie: username=admin
Content-Length: 23

cmd=cgi_get_ipv6&flag=1
```

Next, call an endpoint (e.g., *cgi_get_ssh_pw_status*) that requires admin privileges and authenticate as admin by adding the cookie **username=admin**. Setting the cookie in the browser through the console before visiting the dashboard will authenticate the user as the administrator.

Western Digital Blog: https://blog.westerndigital.com/western-digital-my-cloud-update/
“Recently, security researcher Securify published an authentication bypass vulnerability for our My Cloud products (My Cloud Home is exempt from the vulnerability). We are in the process of finalizing a scheduled firmware update that will resolve the reported issue. We expect to post the update on our technical support site at https://support.wdc.com/ within a few weeks.”

Reality Check:

- 09 April 2017: Discovered vulnerability.
- 10 April 2017: Reported to Western Digital customer support. No vendor response :/
- 17 September 2018: Requested CVE
- 18 September 2018: CVE-2018-17153 assigned
- 18 September 2018: Published details

[Western Digital] **Update 9/21/18:** The issue stated below concerning an authentication bypass vulnerability has been addressed with a hotfix that can be immediately downloaded here: https://support.wdc.com/knowledgebase/answer.aspx?ID=25952&s

The moral of the story? Western Digital took 17 months from first private disclosure, which they blew off, and just three days from public disclosure to a patch. So… put Western Digital on a very short leash and disclose publicly, as Google does, after 90 days, tops. And in WD's case, it might be better to give them, perhaps, two weeks, since they can obviously fix something immediately if they choose to.

And a quick note about "Exploitee.rs"
https://twitter.com/Exploiteers
https://www.exploitee.rs/
https://www.exploitee.rs/index.php/Sony_BDP-S5100

A bug exists in the MTK supplied SDK which affects many Blu-Ray players, including the BDP-S5100. The main binary, which controls all aspects of the player has leftover debug instructions for the VUDU app. When the VUDU app is run, if a file exists named "vudu.txt" , in a directory labeled "vudu" on a FAT formatted flash drive it will attempt to execute "vudu/vudu.sh", and deletes vudu.txt. It runs this sh as root. Using the commands below, you can spawn a root telnet shell, allowing access to the device:

- Create a folder named "vudu" on a FAT formatted flash drive.
- Inside that folder, create a blank file named "vudu.txt"
- Also in that folder, create a file named "vudu.sh" containing the following:

```
#!/bin/sh

echo "executing" > /mnt/sda1/vudu.txt
mount -t overlayfs -o overlayfs /etc/passwd
echo "root::0:0:root:/root:/bin/sh" > /etc/passwd

/mnt/rootfs_normal/usr/sbin/telnetd
```

- Start the player with the flash drive plugged in, and execute the VUDU app. The code has been executed, and a telnet shell now exists for you to connect to on port 23 as root. Following this, you will be brought back to the main menu.

Will the Internet split into separate US and China halves?
Last Wednesday in San Francisco, during an event hosted by venture capital firm Village Global VC, the economist Tyler Cowen asked about the possibility of the Internet fragmenting into different sub-Internets with different regulations and limited access between them in coming years: *"What's the chance, say, 10 to 15 years, we have just three to four separate internets?"*

Eric Schmidt, who is, of course, the past CEO of Google and executive chairman of its parent company, Alphabet, predicted that within the next decade there will be two distinct internets: one led by the U.S. and the other by China. Eric said:

> "I think the most likely scenario now is not a splintering, but rather a bifurcation into a Chinese-led internet and a non-Chinese internet led by America.
>
> If you look at China, and I was just there, the scale of the companies that are being built, the services being built, the wealth that is being created is phenomenal. Chinese Internet is a greater percentage of the GDP of China, which is a big number, than the same percentage of the US, which is also a big number.
>
> If you think of China as like 'Oh yeah, they're good with the Internet,' you're missing the point. Globalization means that they get to play too. I think you're going to see fantastic leadership in products and services from China. There's a real danger that along with those products and services comes a different leadership regime from government, with censorship, controls, etc.
>
> Look at the way BRI works – their Belt and Road Initiative, which involves 60-ish countries – it's perfectly possible those countries will begin to take on the infrastructure that China has with some loss of freedom."

I'm skeptical. And what does that mean and look like from the technological side?

We have an existing and much older model: The global telephone system. As far as I know, someone in China is able to phone someone in the US, and vise versa. It would be deeply weird, and economically non-viable, for there to be two separate telephone systems whose subscribers could not reach each other.

Unwiped Drives and Servers from NCIX Retailer for Sale on Craigslist
PCMag: "Unwiped Servers With Data on Millions Sold on Craigslist"
"Database servers sold at NCIX auction allegedly without being wiped."
eTeknix: "NCIX Database Servers Sold at Craigslist Without Being Wiped"
BleepingComputer: "Unwiped Drives and Servers from NCIX Retailer for Sale on Craigslist"
Sophos Naked Security: Bankrupt NCIX customer data resold on Craigslist
https://nakedsecurity.sophos.com/2018/09/24/bankrupt-ncix-customer-data-resold-on-craigslist/

Sophos: "For Canadian or US customers of NCIX during the last 15 years, they should assume any personal data or credit card information logged with them is now potentially in the hands of cybercriminals and raise any suspicious transactions with their bank."

> What happens to sensitive customer data when a large company that has collected it over many years suddenly goes bust?
>
> It’s easy to assume that databases are wiped by diligent IT staff just before they turn off the lights and close the door for the last time. At the very least that data should have been encrypted.
>
> It has now emerged that something entirely different and more troubling took place when Canadian computer and electronics retailer Netlink Computer Inc (NCIX) declared bankruptcy in December 2017.
>
> According to Privacy Fly researcher Travis Doering, the company simply abandoned much of its equipment in a hurry, which he discovered when it was offered for sale on Craigslist this August.
>
> After arranging a meeting with the seller to examine the hardware, it turned out to comprise 20 Dell PowerEdge and Supermicro servers, 300 desktop PCs, 109 hard drives, and another 400-500 drives that had been inside NCIX desktops or sent to it for repair.
>
> Now for the disturbing bit – it soon became clear that the valuable part of the deal was not the drives themselves but what was on them – 13 terabytes of data all told, including 385,000 database records containing names, email addresses, phone numbers and account passwords, 258,000 of which included full credit card payment details.
>
> A separate Canadian database contained 3.8 million customer records gathered by NCIX between January 2007 and July 2010.
>
> The seller had got hold of passwords to access the databases, while significant amounts of the data were not encrypted in the first place. The price for the data on its own: $15,000.
>
> How did such a data catastrophe come to pass?
>
> Doering’s guess is that NCIX’s landlord was owed money and quickly sold the dead company’s equipment to an auction house, where it was picked up cheaply by the contact who had offered it to him.
>
> Given the calculated way the data was marketed for its value, it seems likely the equipment was targeted precisely because it might contain something that could be sold.
>
> This could have been avoided by implementing full disk encryption within the organization or destroying the drives as their bankruptcy loomed. But the company doesn’t appear to have been storing its databases securely… and nobody seems to be paying any attention to what happens to customer data when companies die.

I think this raises a good point. The Internet is driving a consolidation of retailers. Other background on the NCIX troubles suggested that the root of their problems was their attempt to hold onto physical brick and mortar retail outlets, resisting the push to go fully virtual and online. I recently learned that another past retailer I often used -- iHerb.com -- is now struggling after Amazon began offering a wide range of dietary supplements. I haven't purchased anything from iHerb in several years. With this consolidation comes a concentration of huge amounts of valuable and sensitive information into a single location.

It looks as though bankruptcy proceedings are going to need to start taking a petitioning company's data assets very carefully.

Mirai Botnet Creators Helping FBI Fight Cybercrime to Stay Out of Jail
https://www.justice.gov/usao-ak/pr/hackers-cooperation-fbi-leads-substantial-assistance-other-complex-cybercrime

As we know, the Mirai Botnet and its many rapidly developing descendents (after its source code was deliberately released) is currently the preeminent IoT botnet whose capabilities and IoT inventory are growing with distressing speed. Last week we looked the massive growth in number of attacks but mostly in the startling size of attacks.

Last week, the US Department of Justice's U.S. Attorney's Office for the District of Alaska, where the charges were originally tried, published an update on the status of the three young men who were convicted of cybercrimes related to their original creation and use of the Mirai botnet. The DoJ's Alaska office wrote:

Anchorage, Alaska – U.S. Attorney Bryan Schroder announced today that three defendants have been sentenced for their roles in creating and operating two botnets, which targeted "Internet of Things" (IoT) devices. Paras Jha, 22, of Fanwood, New Jersey; Josiah White, 21, of Washington, Pennsylvania; and Dalton Norman, 22, of Metairie, Louisiana, were sentenced today by Chief U.S. District Judge Timothy M. Burgess. On Dec. 8, 2017, Jha, White, and Norman pleaded guilty to criminal Informations in the District of Alaska charging them each with conspiracy to violate the Computer Fraud & Abuse Act in operating the Mirai Botnet. Jha and Norman also pleaded guilty to two counts each of the same charge, one in relation to the Mirai botnet and the other in relation to the Clickfraud botnet.

After cooperating extensively with the FBI, Jha, White, and Norman were each sentenced to serve a five-year period of probation, 2,500 hours of community service, ordered to pay restitution in the amount of $127,000, and have voluntarily abandoned significant amounts of cryptocurrency seized during the course of the investigation. As part of their sentences, Jha, White, and Norman must continue to cooperate with the FBI on cybercrime and cybersecurity matters, as well as continued cooperation with and assistance to law enforcement and the broader research community. According to court documents, the defendants have provided assistance that substantially contributed to active complex cybercrime investigations as well as the broader defensive effort by law enforcement and the cybersecurity research community.

Jha, White, and Norman became subjects of a federal investigation when, in the summer and fall of 2016, they created a powerful botnet – a collection of computers infected with malicious software and controlled as a group without the knowledge or permission of the computers' owners. The Mirai Botnet targeted IoT devices – non-traditional computing devices that were *connected to the Internet, including wireless cameras, routers, and digital video recorders. The defendants attempted to discover both known and previously undisclosed vulnerabilities that allowed them to surreptitiously attain control over the victim devices for the purpose of forcing the devices to participate in the Mirai Botnet. At its peak, Mirai consisted of hundreds of thousands of compromised devices. The defendants used the botnet to conduct a number of powerful distributed denial-of-service, or "DDoS" attacks, which occur when multiple computers, acting in unison, flood the Internet connection of a targeted computer or computers. The defendants' involvement with the original Mirai variant ended in the fall of 2016, when Jha posted the source code for Mirai on a criminal forum. Since then, other criminal actors have used Mirai variants in a variety of other attacks.*

UK Regulator Fines Equifax £500,000 ($658,419) Over 2017 Data Breach

As we know, last year Equifax suffered a significant data breach after leaving a widely known and long since patched flaw in Apache Struts present and unpatched in their Internet-facing systems. This allowed bad guys to get into the Equifax servers and exfiltrate their customers' extremely sensitive financial data.

It's always easy to pick on someone after the fact, but a company such as Equifax is collecting and vacuuming up sensitive data without consumer's knowledge and permission and reselling what they have amassed. So it's not unreasonable to expect them to be really really careful with the data they have collected. I think that having a full time IT person, whose entire job description is to do **nothing** other than check for available updates to any of the software packages they are using and pursue their installation with suitable priority… would not be an unreasonable expectation.

In any event… the shoes continue to drop to good end, I think. The Equifax Breach has grown into a meme that no other CIO's want aimed at them.

This latest shoe to drop, courtesy of the UK's privacy watchdog, the Information Commissioner's Office (ICO), has issued its largest possible monetary penalty under the country's Data Protection Act for the massive data breach which affected 15 million UK citizens.

- 19,993 UK customers had their names, dates of birth, telephone numbers and driving license numbers exposed.
- 637,430 UK customers had their names, dates of birth and telephone numbers exposed.
- Up to 15 million UK customers had names and dates of birth exposed.
- Some 27,000 Britishers also had their Equifax account email addresses swiped.
- 15,000 UK customers also had their names, dates of birth, addresses, account usernames and plaintext passwords, account recovery secret questions, and answers, obscured credit card numbers, and spending amounts stolen by hackers.

And… this was readily preventable. Although it's been noted that this is just a hand slap for a $15 Billion company, it returns Equifax to the news and hopefully further encourages other CIOs to stay off the similar public radar.

SpinRite

Louis Vincent in Ottawa, Ontario, Canada
Subject: Spinrite fixes Task Manager
Date: 14 Sep 2018 19:31:51
:
Hi Steve,

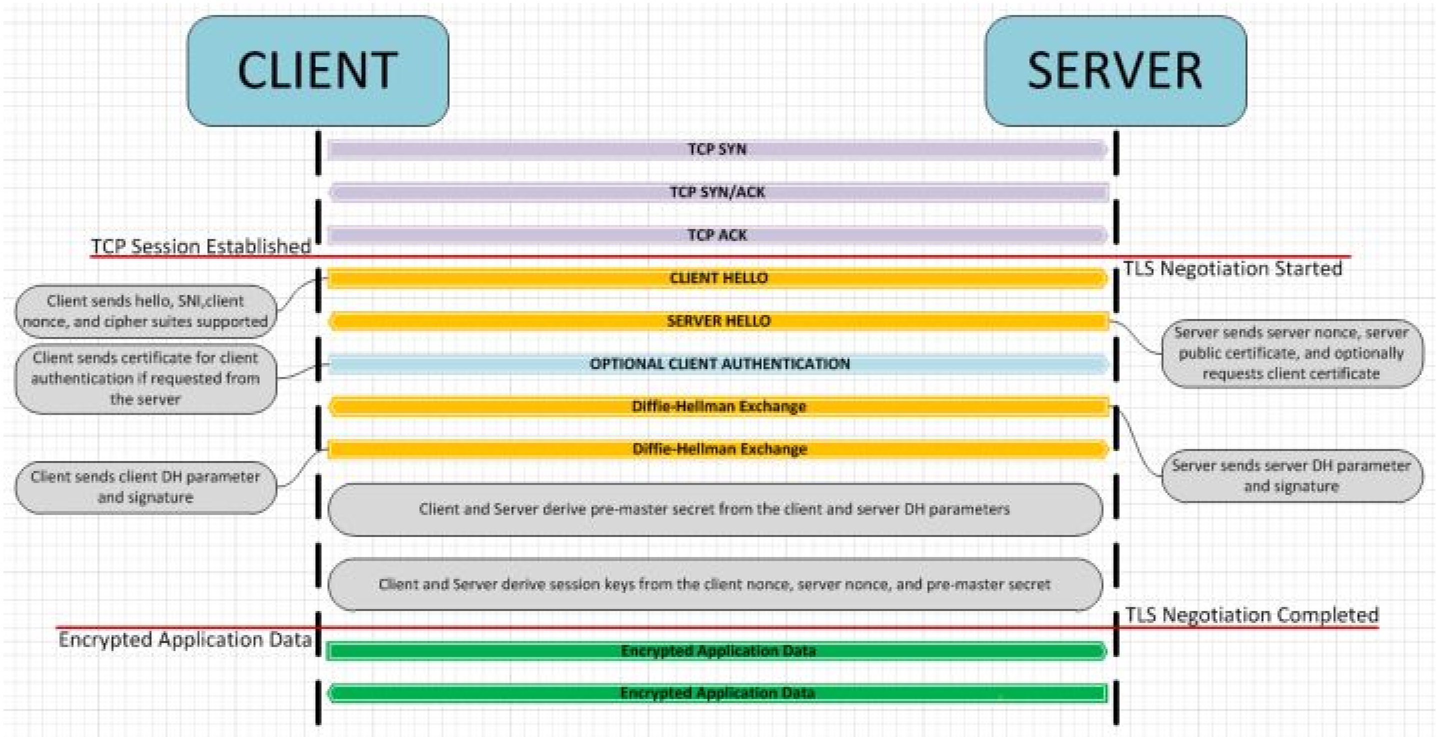
Spinrite owner since 2007. Security Now listener... yadi yada.

This past week my laptop started grinding to a halt. With no program running, Task Manager would show that the CPU was pinned at 100%. Did I have a cryptominer on my device I wondered? The weird thing was that Task Manager was showing that the process taskmgr.exe was itself taking 40%-50% of the CPU with McAfee taking most of the rest. That was odd at least for the taskmgr.exe. I did not find the issue while in Safe Mode but every time I logged back in normally, taskmgr.exe was back hogging the CPU.

I decided to “SpinRite” the 500 MB hard drive. As you might predict, a couple of hours later SpinRite reports nothing crucial found on the drive but a reboot later and everything is working as it was at the beginning of the week.

Thanks for the great product.

SNI Encryption


CLIENT
SERVER
TCP SYN
TCP SYN/ACK
TCP ACK
TCP Session Established
TLS Negotiation Started
CLIENT HELLO
Client sends hello, SNI,client nonce, and cipher suites supported
SERVER HELLO
Server sends server nonce, server public certificate, and optionally requests client certificate
OPTIONAL CLIENT AUTHENTICATION
Client sends certificate for client authentication if requested from the server
Diffie-Hellman Exchange
Diffie-Hellman Exchange
Client sends client DH parameter and signature
Server sends server DH parameter and signature
Client and Server derive pre-master secret from the client and server DH parameters
Client and Server derive session keys from the client nonce, server nonce, and pre-master secret
TLS Negotiation Completed
Encrypted Application Data
Encrypted Application Data
Encrypted Application Data

Yesterday, Cloudflare's Matthew Prince posted:
"Encrypting SNI: Fixing One of the Core Internet Bugs"
https://blog.cloudflare.com/esni/

(Also, this coming Thursday is the 8th birthday of Cloudflare.)

Matthew reminds us that although HTTPS encrypts the content of our communications, the privacy of **where** we go on the Internet remains more elusive. In other words, we still have unencrypted metadata.

Matthew writes:

> "While the contents sent to or received from a HTTPS site are protected, the fact that you visited the site can be observed easily in a couple of ways. Traditionally, one of these has been via DNS. DNS queries are, by default, unencrypted so your ISP or anyone else can see where you're going online. That's why last April, we launched 1.1.1.1 — a free (and screaming fast) public DNS resolver with support for DNS over TLS and DNS over HTTPS. 1.1.1.1 has been a huge success and we've significantly increased the percentage of DNS queries sent over an encrypted connection. Critics, however, rightly pointed out that the identity of the sites that you visit still can leak in other ways. The most problematic is something called the Server Name Indication (SNI) extension."

SNI??

Shared hosting, disambiguating, selecting the target cert.

Encrypted Server Name Indication for TLS 1.3
https://tools.ietf.org/html/draft-ietf-tls-esni-01

Eric Rescorla, RTFM, Inc. Kazuho Oku, Fastly
Chris Wood, Apple, Inc. Nick Sullivan, Cloudflare

Dated: Last Tuesday, September 18th…

DISCLAIMER: This is very early a work-in-progress design and has not yet seen significant (or really any) security analysis. It should not be used as a basis for building production systems.

Although TLS 1.3 [RFC8446] encrypts most of the handshake, including the server certificate, there are several other channels that allow an on-path attacker to determine the domain name the client is trying to connect to, including:

o Cleartext client DNS queries.
o Visible server IP addresses, assuming the the server is not doing domain-based virtual hosting.
o Cleartext Server Name Indication (SNI) [RFC6066] in ClientHello messages.

DNS over HTTPS and DNS over DTLS provide mechanisms for clients to conceal DNS lookups from network inspection, and many TLS servers host multiple domains on the same IP address. In such environments, SNI is an explicit signal used to determine the server's identity. Indirect mechanisms such as traffic analysis also exist.

The TLS WG has extensively studied the problem of protecting SNI, but has been unable to develop a completely generic solution. One of the more difficult problems is "Do not stick out": If only sensitive/private services use SNI encryption, then SNI encryption is a signal that a client is going to such a service. For this reason, much recent work has focused on concealing the fact that SNI is being protected. Unfortunately, the result often has undesirable performance consequences, incomplete coverage, or both.

The design in this document takes a different approach: it assumes that private origins will co-locate with or hide behind a provider (CDN, app server, etc.) which is able to activate encrypted SNI (ESNI) for all of the domains it hosts. Thus, the use of encrypted SNI does not indicate that the client is attempting to reach a private origin, but only that it is going to a particular service provider, which the observer could already tell from the IP address.

The protocol designed in this document is quite straightforward.

First, the provider publishes a public key which is used for SNI encryption for all the domains for which it serves. This document defines a publication mechanism using DNS, but other mechanisms are also possible. In particular, if some of the clients of a private server are applications rather than Web browsers, those applications might have the public key preconfigured.

When a client wants to form a TLS connection to any of the domains served by an ESNI-supporting provider, it replaces the "server_name" extension in the ClientHello with an "encrypted_server_name" extension, which contains the true extension encrypted under the provider's public key. The provider can then decrypt the extension.

```
struct {
   NamedGroup group;
   opaque key_exchange<1..2^16-1>;
} KeyShareEntry;

struct {
   uint16 version;
   uint8 checksum[4];
   KeyShareEntry keys<4..2^16-1>;
   CipherSuite cipher_suites<2..2^16-2>;
   uint16 padded_length;
   uint64 not_before;
   uint64 not_after;
   Extension extensions<0..2^16-1>;
} ESNIKeys;
```

This structure is placed in the RRData (Resource Record) section of a TXT record as a base64-encoded string. If this encoding exceeds the 255 octet limit of TXT strings, it must be split across multiple concatenated strings as per Section 3.1.3 of [RFC4408]. Servers MAY supply multiple ESNIKeys values, either of the same or of different versions. This allows a server to support multiple versions at once.

IANA is requested to Create an entry, encrypted_server_name(0xffce), in the existing registry for ExtensionType (defined in [RFC8446]).