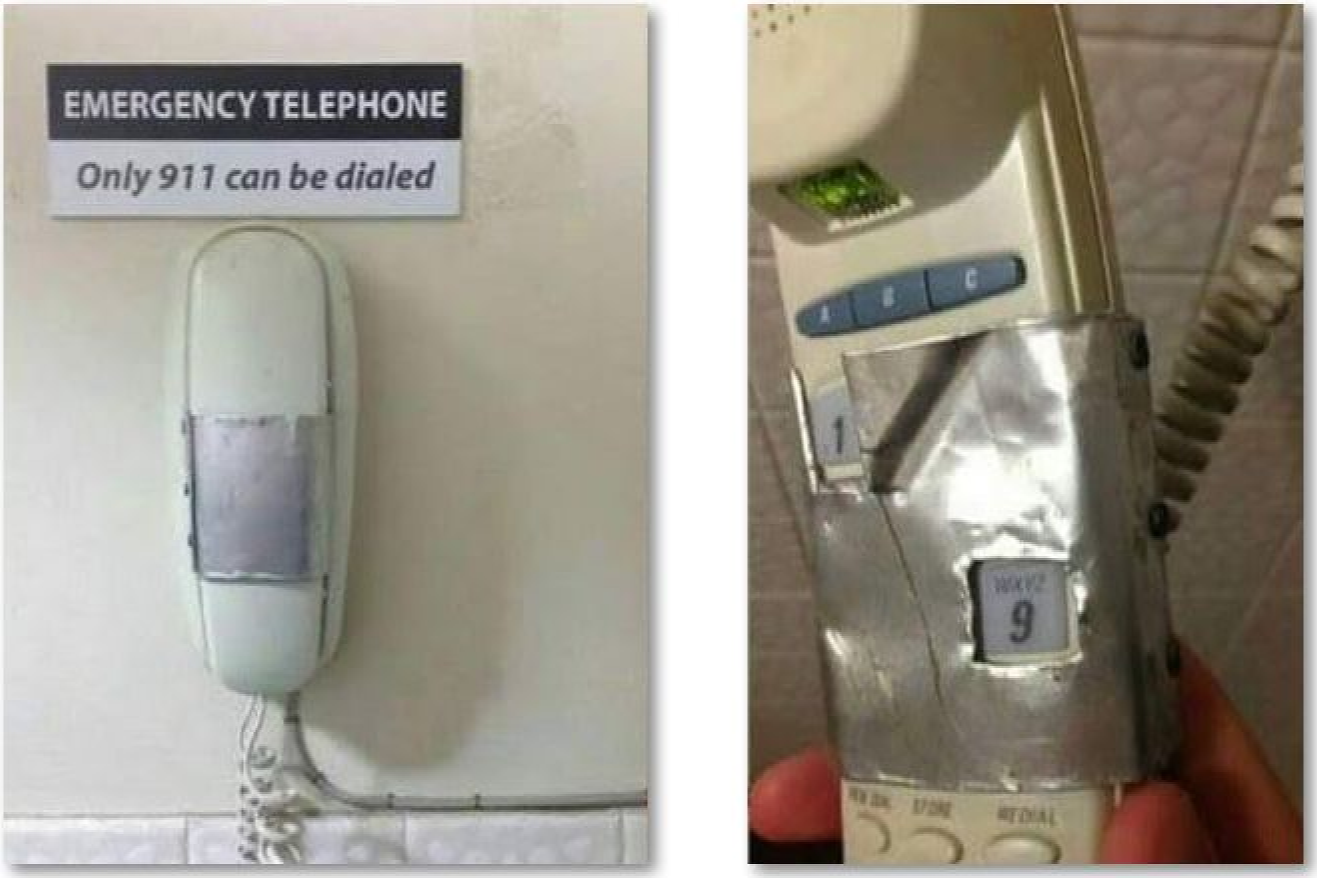
Security Now! #614 - 05-30-17
Vulnerabilities Galore!

This week on Security Now!

This week we discuss a new non-eMail medium for spear phishing, Chipotle can't catch a break, social engineering WannaCry exploits on Android, video subtitling now able to takeover our machines, a serious Android UI design flaw that Google appears to be stubbornly refusing to address, Linux gets its own version of WannaCry, another dangerous NSA exploit remains unpatched and publicly exploitable on WinXP and Server 2003 machines, a look at 1Password's brilliant and perfect new "Travel Mode", Google extends its ad-tracking into the offline world, some follow-ups, miscellany, and closing-the-loop feedback from our terrific listeners... concluding with my possibly-useful analogy to explain the somewhat confusing value of open versus closed source.

Our Picture of the Week



Really?

Security News

Heads UP! : Spear Phishing -- not just for eMail anymore.
Security research firms are reporting a shift from eMail to the Twitter and Facebook social media networks where there is generally a feeling of community and trust. The message of not clicking links in eMail is finally beginning to take hold... so the bad guys are shifting their attacks to non-eMail channels.

In an experiment conducted late last year, an automated program sent links to 819 people. Of those, 275 users opened the links (just over 1 in 3).

Chipotle says ‘most’ of its restaurants were infected with credit card stealing malware
https://www.theverge.com/2017/5/26/15701776/chipotle-restaurants-hacked-credit-card-malware

Credit card details of purchases made from March 24 through April 18th may have been exposed.

They said "most restaurants" -- but since this was a breach of their central charge processing system, it's more likely that ALL branches were vulnerable through that period.

There's really no recourse available to consumers.

Just scrutinize your credit card statements spanning that period and bring any charges you don't recognize to the attention of your CC company, who will then credit your balance.
Just explain that you used that card at a Chipotle restaurant, and they experienced a major credit card processing breach during the brief period their food was making its way rapidly through your digestive tract.

From our "We should have seen this coming" department...
Even though WannaCry is entirely a Windows problem, the recent high profile worldwide WannaCry malware attack has spawned a myriad of fake Android "WannsCry protection" apps. They are largely adware leveraging the current WannaCry hysteria to get themselves onto people's Android devices, where they then attempt to get their user to install additional junk.

And here's one no one saw coming: The Attack of the Subtitles...
Or as CheckPoint called it: Hacked in Translation - from Subtitles to Complete Takeover
http://blog.checkpoint.com/2017/05/23/hacked-in-translation/

Researchers at CheckPoint have discovered a new attack vector which threatens millions of users worldwide: attack by subtitles. By crafting malicious subtitle files, which are then downloaded by a victim’s media player, attackers can take complete control over any type of device via vulnerabilities found in many popular streaming platforms, including VLC, Kodi (XBMC), Popcorn-Time and strem.io.

CheckPoint writes that there are approximately 200 million video players and streamers that currently run the vulnerable software, making this one of the most widespread, easily accessed and zero-resistance vulnerabilities reported in recent years.

They've tested and found multiple serious subtitle-parsing vulnerabilities in every video player they've examined starting with the four of the most prominent media players: VLC, Kodi, Popcorn Time and Stremio. They have reason to believe similar vulnerabilities exist in other media players as well.

They did follow responsible disclosure practice and reported all vulnerabilities and exploits to the developers of the vulnerable media players they examined. Some of the issues were already fixed, while others are still under investigation. To allow the developers more time to address the vulnerabilities, thay've decided not to publish any further technical details at this point.

VLAN (the player I, and 170 million others, use) is at v2.2.6 for Windows.

Platforms Update:

- VLC– Officially fixed and available to download on their website
  Link: http://get.videolan.org/vlc/2.2.5.1/win32/vlc-2.2.5.1-win32.exe

- Kodi– Officialy fixed and available to download on their website.
  Link: https://kodi.tv/download

- PopcornTime– Created a Fixed version, however it is not yet available to download in the official website. The fixed version can be manually downloaded via the following
  Link: https://ci.popcorntime.sh/job/Popcorn-Time-Desktop/249

- Stremio– Officially Fixed and avilable to download on their website
  Link: https://www.strem.io/

Also… I discovered "VLMC" -- VideoLAN Movie Creator, a free, multi-platform (Windows, Linux, MacOS) non-linear editing system. It's still under development and not yet ready for pre-release beta testing... but it's on the way.

Links:

- Hackers are hiding malware in subtitle files
  https://techcrunch.com/2017/05/24/hackers-are-hiding-malware-in-subtitle-files/amp/

- Beware! Subtitle Files Can Hack Your Computer While You're Enjoying Movies
  http://thehackernews.com/2017/05/movie-subtitles-malware.html

“Cloak & Dagger” vulnerability uses Android's own features to fool users.
The team composed of researchers from University of California, Santa Barbara and the Georgia Institute of Technology are back with a new and worrisome finding…

http://cloak-and-dagger.org/

In the Proceedings of the IEEE Symposium on Security and Privacy (S&P), San Jose, CA, May 2017 this work was awarded the Distinguished Practical Paper Award!

Will be shown during this summer's July 2017 BlackHat in Las Vegas.

<quote> Cloak & Dagger is a new class of potential attacks affecting Android devices. These attacks allow a malicious app to completely control the UI feedback loop and take over the device — without giving the user a chance to notice the malicious activity.

These attacks only require two permissions that, in case the app is installed from the Play Store, the user does not need to explicitly grant and for which they are not even notified.

Our user study indicates that these attacks are practical. These attacks affect all recent versions of Android (including the latest version, Android 7.1.2), and they are yet to be fixed.

Main Takeaways

- We uncover a series of vulnerabilities and design shortcomings affecting the Android UI.
- These attacks abuse one or both of the SYSTEM_ALERT_WINDOW ("draw on top") and BIND_ACCESSIBILITY_SERVICE ("a11y").
- If the malicious app is installed from the Play Store, the user is not notified about the permissions and she does not need to explicitly grant them for the attacks to succeed. In fact, in this scenario, "draw on top" is automatically granted, and this permission is enough to lure the user into unknowingly enable a11y (through clickjacking).
- The possible attacks include advanced clickjacking, unconstrained keystroke recording, stealthy phishing, the silent installation of a God-mode app (with all permissions enabled), and silent phone unlocking + arbitrary actions (while keeping the screen off). See the full list below.
- These attacks are practical: we performed a user study (with 20 human subjects), and no user understood what happened.
- Most of these attacks are due to design issues, and they are thus challenging to prevent. In fact, one may say that some of these functionality work "as intended"; Nonetheless, this work shows that this functionality can be abused.
- To date, all these attacks are still practical (see "Which versions of Android are affected" and "Responsible Disclosure" below).

List of Attacks:

Attacks that abuse the "draw on top" permission:

- Context-aware clickjacking & Context hiding: two techniques that make luring the user to enable the accessibility service practical, even when the latest security mechanisms (e.g., "obscured flag") are correctly implemented and enabled. (Note: others have identified ways to use clickjacking to get a11y. See "FAQ" below.)

- Invisible Grid Attack, allowing unconstrained keystroke recording, including password, private messages, etc.

Attacks that abuse "accessibility service" permission:

- Unconstrained keystroke recording, including passwords. According to the documentation, this should not be possible (See "security note" here)

- Security PIN stealing

- Device unlock through PIN injection + perform arbitrary actions while keeping the screen off!

- Stealing two-factor authentication tokens (SMS-based, Google Authenticator, and other app-based tokens)

- Ad hijacking

- Web exploration

Attacks that abuse both permissions:

- Silent installation of God-mode app (with all permissions enabled)

- Stealthy phishing (for which the user finds herself logged in, as she would expect)

Responsible Disclosure:

We responsibly disclosed our findings to Google's Android security team. A timeline of the disclosure steps and responses from Google are as follows:

- August 22nd, 2016 — We opened several issues on the bug tracker for the Android Open Source Project (AOSP).

- August 31st, 2016 — Android security team sets severity as "Moderate" for one of the bugs ("draw on top" ? unconstrained keystroke recording).

- September 30th, 2016 — Android security team marks one of the reported bugs ("a11y"-unconstrained keystroke recording + leaking security PIN + unlocking device while keeping the screen off) as "work as intended".

- October 10th, 2016 — We follow up pointing out that the Accessibility Service's documentation states that a11y should not be able to access passwords and that a11y should not be able to unlock the device and perform arbitrary actions while keeping the screen off.

- October 13th, 2016 — Android security team states that "The password will be repeated if the user explicitly turns that on in settings under Settings - Accessibility - Speak passwords. The option is off by default. If the user explicitly enables this feature, it is not a security vulnerability."

- October 14th, 2016 — We follow up clarifying that our attack works even without this feature (In fact, our report does not even mention that.)

- October 18th, 2016 — Android security team marks this bug as "High severity".

- November 28th, 2016 — Android security team downgrades this bug as "not a security issue" and marks it as "Won't Fix (Intended Behavior)" because "limiting those services would render the device unusable".

- December 19th, 2016 — A draft of our IEEE S&P'17 paper is shared with Adrian Ludwig, director of Android Security.

- March 15th, 2017 — We follow up, pointing out that the documentation states otherwise. We also follow up on all the other bugs we opened, asking for a status update.

- May 3rd, 2017 — We follow up a second time asking for a status update for the bugs we reported.

- May 4th, 2017 — Android security team keeps our a11y findings as "won't fix", but they state they will update the documentation. We did not receive updates about the other bugs we reported.

- May 8th, 2017 — We have a telco with the anti-malware and a11y Google teams during which we thoroughly discussed all the details of our research.

- May 19th, 2017 — The a11y team confirms the a11y-related issues we reported as "won't fix".

- May 22nd, 2017 — This website and our research paper at IEEE S&P are made public.

Current — All the attacks discussed by this work are still practical, even with latest version of Android (Android 7.1.2, with security patches of May 5th installed).

Frequently Asked Questions

How can an app stealthily obtain the two permissions?

A malicious app can declare the "draw on top" and the a11y permissions through its manifest file. If the app is hosted and installed through the Google Play Store, the attacks will not require the user to explicitly notify the user about the two permissions. In fact, in this scenario, the "draw on top" permission is automatically granted, and this work shows how it is possible to get access to a11y (through clickjacking) even when the latest security mechanisms are enabled and correctly implemented.

Why is the "draw on top" permission automatically granted?

This behavior appears to be a deliberate decision by Google, and not an oversight. To the best of our understanding, Google's rationale behind this decision is that an explicit security prompt would interfere too much with the user experience, especially because it is requested by apps used by hundreds of millions of users. For example, Facebook requires this permission to implement "Android chat heads", one of its very popular features.

Are these permissions shown to the user after the app is installed?

They are not. For Android 5 (or older versions), the list of permissions requested by the app is shown at installation time: however, the "draw on top" and the a11y are not included in this list (because they are treated as "special"). For Android 6 (and later), the list of permissions is not shown to the user at installation time (and, as mentioned above, the "draw on top" is automatically granted).

How difficult is it to get an app with these two permissions approved to the Google Play Store?

A quick experiment shows that it is trivial to get such an app accepted on the Google Play Store. In particular, we submitted an app requiring these two permissions and containing a non-obfuscated functionality to download and execute arbitrary code (attempting to simulate a clearly-malicious behavior): this app got approved after just a few hours (and it is still available on the Google Play Store).

What do you recommend to users?

We recommend users to check which applications have access to the "draw on top" and the a11y permissions. Unfortunately, both permissions are considered "special" and, for this reason, certain versions of Android may show "no permission required" even if, in fact, the app has access to both the permissions required for our attack. We provide instructions for several versions of Android

This work shows that the user should not consider her device's UI as a trusted source of information. Thus, from a conceptual point of view, the user should rely on other means than the device's UI itself. An alternative solution is to use command line tools (such as adb) or to determine the permissions requested by each app through the Play Store website. For example, to check the permissions of the official LastPass app (which requires both permissions), you can go to its Play Store page, scroll down, and click "View details" under "Permissions". The "draw on top" permission will appear under the "Others" / "draw over other apps" label, while the a11y will appear under "Others" / "bind to an accessibility service".

Why are these bugs not fixed yet?

Some of the issues uncovered by this work are design-related issues — not simple bugs — and it thus necessarily takes more time to fix them. Moreover, these are not "classic" low-level issues, but UI-related problems. These issues may be more challenging to fix since changes may introduce backward compatibility issues and changes that are "visible" to end users. Finally, some of the bugs were marked as "won't fix", and they will thus not be fixed.

Are these attacks practical?

We believe so. We performed a user study with 20 human subjects, and none detected to be under attack. We report more details in our paper.

SambaCry on Linux

The Hacker News Headline is: "7-Year-Old Samba Flaw Lets Hackers Access Thousands of Linux PCs Remotely"

- http://thehackernews.com/2017/05/samba-rce-exploit.html?m=1

BleepingComputer writes: Over 104,000 Samba Installations Vulnerable to Remote Takeover Attacks

- https://www.bleepingcomputer.com/news/security/over-104-000-samba-installations-vulnerable-to-remote-takeover-attacks/

Hack-A-Day

- http://hackaday.com/2017/05/25/linux-sambacry/

The Samba security advisory published last Wednesday wrote: "All versions of Samba from 3.5.0 onwards [since March 1st, 2010] are vulnerable to a remote code execution vulnerability, allowing a malicious client to upload a shared library to a writable share, then cause the server to load and execute it."

And since the smbd typically runs at root, the uploaded and executed module runs with the same full ROOT privileges!

A shodan.io query of "port:445 !os:windows" shows approximately one million non-Windows hosts that have tcp/445 open to the Internet, more than half of which exist in the United Arab Emirates (36%) and the U.S. (16%). However, it isn’t clear how many of them are running vulnerable versions of Samba.

The vulnerability can be exploited with a single line of code. A malicious client can upload and cause the smbd server to execute a shared library from a writable share. Exploit modules are already available from Metasploit to exploit this issue on Linux ARM, X86 and X86_64 architectures.

The bug has nothing to do with EternalBlue. Whereas Eternalblue is a buffer overflow exploit, the Linux/SMB vulnerability leverages an arbitrary shared library load and execute.

Vulnerable: All versions between Samba 3.5.0 and 4.6.4/4.5.10/4.4.14

This is a TOTALLY WORMABLE vulnerability.

Or, as security firm F5 put it: "Network + Remote Code Execution + Root = Drop What You're Doing and Patch."

A patch addressing this defect has been posted to the official website and Samba 4.6.4, 4.5.10 and 4.4.14 have been issued as security releases to correct the defect. Patches against older Samba versions are also available.

Debian, Ubuntu, CentOS and Red Hat have all taken immediate action to protect their users and have released patches for their supported versions. Additionally, security workarounds have also been provided for unsupported ones.

If the patch cannot be added immediately, a workaround is to add the parameter "nt pipe support = no" to the [global] section of your smb.conf and restart smbd. Note that this may disable some expected functionality for Windows clients.

NAS vendors have work on their hands. Different brands and models that use Samba for file sharing (a lot, if not all, of them provide this functionality) will have to issue firmware updates to patch this flaw. If the firmware updates for these appliances take the same time they usually do, we will have this bug around for quite some time.

A great page with detailed mitigation instructions:
https://www.tecmint.com/fix-sambacry-vulnerability-cve-2017-7494-in-linux/

Meanwhile, the NSA's Windows 'EsteemAudit' RDP Exploit Remains Unpatched on Windows XP and Server 2003
Shodan reveals 24,734 vulnerable WinXP & Server 2003 systems.

"EsteemAudit" is another very dangerous NSA-developed Windows hacking tool leaked by the Shadow Brokers. It targets the RDP (Remote Desktop Protocol) service (port 3389) on Microsoft Windows Server 2003 / Windows XP machines.

EsteemAudit can also be weaponized into wormable malware, similar to the WannaCry ransomware, which allows hackers to propagate in the enterprise networks, leaving thousands of systems vulnerable to ransomware, espionage and other malicious attacks.

As we know, Microsoft no longer supports Windows Server 2003 and Windows XP, and unlike EternalBlue Microsoft has not released an emergency patch for the EsteemAudit exploit, so somewhere around 24,734 vulnerable systems remain exposed on the Internet for anyone to hack.

An unofficial patch is available from ensilo security:
http://pages.ensilo.com/download-the-patch-for-esteemaudit-exploit

They write: The patch for Windows XP and Server 2003 supports silent installation and does not require a reboot, which helps users avoid the required downtime typically associated with patch installations. Upon patching, any attempt to use an ESTEEMAUDIT exploit to infect a patched machine will fail.

1Password's Travel Mode: The perfect solution for TDF: "Travel Data Frisking"

Rick Fillion: Security for AgileBits
https://blog.agilebits.com/2017/05/18/introducing-travel-mode-protect-your-data-when-crossing-borders/

They did it all EXACTLY right:

First, your various secrets are partitioned into vaults, and vaults can be individually flagged as "Safe for Travel" or not.

So, as you approach customs or a particularly officious-looking TSA agent, you login to your 1Password.com account and activate "Travel Mode"... whereupon all non "Safe for Travel" vaults are removed from your connected devices.

The vaults are NOT obscured or hidden -- they are wiped.

So you travel through customs smiling and trying not to bow too deep... and, if you are "Data Frisked" the duly anointed officials will see only those secrets you previously decided to allow them to see.

Once you are safely through inspection, you log back into your 1Password.com account, turn off Travel Mode... and all of your "unsafe for travel" vaults immediately reappear and are repopulated with your data.

Hudson: "Stop your grinning and drop your linen."
Google now knows when its users go to the store and buy stuff
https://www.washingtonpost.com/news/the-switch/wp/2017/05/23/google-now-knows-when-you-are-at-a-cash-register-and-how-much-you-are-spending/

The Washington Post:

Google has begun using billions of credit-card transaction records to prove that its online ads are prompting people to make purchases – even when they happen offline in brick-and-mortar stores, the company revealed [last] Tuesday.

The advance allows Google to determine how many sales have been generated by digital ad campaigns, a goal that industry insiders have long described as "the holy grail" of online advertising. But the announcement also renewed long-standing privacy complaints about how the company uses personal information.

To power its multibillion-dollar advertising juggernaut, Google already analyzes users' Web browsing, search history and geographic locations, using data from popular Google-owned apps like YouTube, Gmail, Google Maps and the Google Play store. All that information is tied to the real identities of users when they log into Google's services.

The new credit-card data enables the tech giant to connect these digital trails to real-world purchase records in a far more extensive way than was possible before. But in doing so, Google is again treading in territory that consumers may consider too intimate and potentially sensitive. Privacy advocates said few people understand that their purchases are being analyzed in this way and could feel uneasy, despite assurances from Google that it has taken steps to protect the personal information of its users.

Google also declined to detail how the new system works or what companies are analyzing records of credit and debit cards on Google's behalf. Google, which saw $79 billion in revenue last year, said it would not handle the records directly but that its undisclosed partner companies had access to 70 percent of transactions for credit and debit cards in the United States.

Marc Rotenberg, executive director of the Electronic Privacy Information Center, was quoted in The Washington Post's story, saying: "What's really fascinating to me is that as companies become increasingly intrusive in terms of their data collection, they also become more secretive." He has urged government regulators and Congress to demand answers about how Google and other technology companies are collecting and using data from their users.

Google said it took pains to protect to protect user privacy.

"While we developed the concept for this product years ago, it required years of effort to develop a solution that could meet our stringent user privacy requirements," Google said in a statement. "To accomplish this, we developed a new, custom encryption technology that ensures users' data remains private, secure, and anonymous."

The FCC's public feedback system is once again up and taking opinions

- http://gofccyourself.com

StackExchange / DigiCert Follow-up
From: Clint Wilson, DigiCert Product Manager
Subject: Stack Exchange Certificate insights and clarifications

Hi Steve!

I got to the Stack Exchange portion of SN 613. Great discussion! I did have some insight I thought I could share. I know you already know most, if not all, of the below, but I'm writing it with your audience in mind as well, where some of these details are probably not readily available or apparent.

Stack Exchange uses an Organization Validation level certificate. The certificate includes Organization identification information, similar to EV, but without some of the details of EV. OV certificates are not distinguished by browsers any differently than DV certificates, so it's essentially impossible to tell the difference without looking at the certificate details.

An example of an EV Subject Distinguished Name would look like:
CN = www.digicert.com
O = DigiCert, Inc.
L = Lehi
S = Utah
C = US
PostalCode = 84043
STREET = 2600 West Executive Parkway
STREET = Suite 500
SERIALNUMBER = 5299537-0142 [The SERIALNUMBER value matches the government registration number of the company; in our case with the Utah Chamber of Commerce: https://secure.utah.gov/bes/details.html?entity=5299537-0142]
1.3.6.1.4.1.311.60.2.1.2 = Utah
1.3.6.1.4.1.311.60.2.1.3 = US
2.5.4.15 = Private Organization

While an OV Subject Distinguished Name is a bit truncated:
CN = blog.digicert.com
O = DigiCert, Inc.
S = Utah
L = Lehi
C = US

Stack Exchange uses OV certificates because they need to use Wildcard names in order to support non-SNI clients and relying party software; EV certificates are not allowed to include Wildcard DNS names except for Tor Hidden Services .onion addresses.

Stack Exchange's main certificate includes multiple Wildcard names to further support non-SNI clients (and to simplify some of the deployment requirements). This is something I'd love to say is unique to DigiCert, but it has become relatively readily available from most commercial CAs in recent years (though I believe we were among the first to offer these, notably to Wikipedia, Facebook, and other similar organizations).

Stack Exchange's main certificate is issued from the "DigiCert High Assurance EV Root CA" Root. The Intermediate used is "DigiCert SHA2 High Assurance Server CA". DigiCert has two "primary" Roots which are used to issue SSL certificates: DigiCert High Assurance EV Root CA (HAEVR) and DigiCert Global Root CA (GR). Of these, only HAEVR is enabled by Root stores such as Mozilla and Google to show the EV "green bar", but both GR and HAEVR can issue OV certificates where needed. Additionally, HAEVR is used to issue some types of EV Code Signing certificates, for things like Microsoft kernel mode enabled signatures.

Though HAEVR is able to issue EV certificates, the qualification as "EV" is somewhat complex. Most of what people associate with "EV" is the extra security indicator Browsers show. It's important to understand this indicator is not part of any Industry Standard or Browser agreement; it's simply something that has been common practice and so, in general, Browsers continue to support this extra indicator. They will likely not do so forever. Despite that, "EV" represents additional validation efforts by CAs regardless of whether relying party software, such as Browsers, chooses to differentiate those efforts. Under current practice, Browsers will only display the EV indicator if certain criteria are met. For all Browsers, this includes looking for the CA/B Forum EV object identifier (OID) in the Certificate Policies extension of the certificate. That is identified here (https://cabforum.org/object-registry/) as 2.23.140.1.1. Additionally, Browsers can have their own EV rules, such as Chrome's Certificate Transparency policy and requirement. DigiCert will only assert the EV OID for certificates validated in accordance with the CA/B Forum's EV Guidelines, DigiCert's CPS, and any Root store requirements.

In Stack Exchange's case, we did not assert the EV OID and instead issued it from an Intermediate which is unable to assert that OID (DigiCert SHA2 High Assurance Server CA). So why would Stack Exchange use an EV and OV Root instead of the, seemingly, more appropriate OV-only Root for their OV Wildcard certificates? I'm actually not positive if this was requested by Stack Exchange, but we would have issued it this way regardless. The reasoning is that DigiCert High Assurance EV Root CA has compatibility with a very few, very old, programs and libraries which DigiCert Global Root CA does not. The difference in compatibility is on the order of hundredths of a percent, but for sites like Stack Exchange, we feel that difference is worth taking into account. We do the same for Facebook and similar sites, even though they use primarily OV certificates.

I think that was everything I thought of while listening to the podcast. Thanks so much for all the awesome work you do, Steve! It's so much appreciated.

Miscellany

Scott Foger (@fogrito)
My 8 yr old son is listening to @SecurityNow podcast while swinging at the playground. @SGgrc is his favorite. Gotta start 'em young

Alien Covenant

Twin Peaks

Increase Your Windows Resources, Use the Most Efficient Software
http://www.makeuseof.com/tag/resource-efficient-windows-software/

Ben Stegner posted on "MakeUseOf" his careful comparison of application efficiency looking at CPU usage and memory consumption.

Ben writes: For browsers, we tested Google Chrome, Mozilla Firefox, and Microsoft Edge. While Edge is the default browser in Windows 10, it's somewhat underwhelming despite Microsoft's claims that it's faster and more secure than Chrome. Thus, we wanted to put it to the test.

Firefox only keeps one process in the Task Manager, making it a lot easier to check its usage. With the MakeUseOf tab opened, Firefox was using 0.1 percent of the CPU and 324.9 MB of RAM. This is a lot more efficient than Chrome for the same tabs!

In one snapshot, Chrome was using 3.2 percent of the CPU and 745.9 MB of RAM. Chrome is known for hogging RAM, so you'd expect to see high numbers here. This isn't atrocious, but three-quarters of a gigabyte is a lot for just five tabs. You probably have many more than that open on normal days!

Like Chrome, Edge breaks its tabs into separate processes. Thus, the main process of 0 percent CPU usage and 36.4 MB of RAM isn't telling the whole story. Adding everything up shows that Edge uses about 18.5 percent of the CPU and 1.459 GB of RAM. For the same five tabs, that's a lot more resource usage than Chrome, where Chrome is more than twice the usage of Firefox.

Jason White (@berlingoff)
Thank you! The Frontiers Saga is everything I could hope for in a Sci-Fi story. Outstanding action, great characters, intriguing technology, oh and can I say again, ACTION, the primary battle in book 3 is absolutely amazing! I have an hour and a half commute several times a week and this series is 2nd on my list, after Security Now. Thanks for the recommendation!

Steven Doyle (@northmendo)
Hey Steve, I just had to say thanks. I absolutely love the Frontier Saga. I have started listening a week or so after you mentioned them and am about to finish book 10. They have consumed all of my free listening time. Pro tip for everyone who doesn't want to spend the money to purchase all of the audio-books. I have been able to find all of them at my county library.

- Steven,
- Thanks for your "Frontier's Saga" feedback. I am currently re-reading them more leisurely and slowly this time. It's so much fun to watch the relationships forming... to know who Tug and Dumar really are. To know about Jalea. To watch Josh and Loki develop. And to watch Cameron and Jessica grow to understand who Nathan is becoming.

SpinRite

From: "Jeff"
Subject: I <3 Spinrite!
Location: Columbus, Ohio

Just a quick note to say how much I love SpinRite. I bought it after listening to my first episode of Security Now something like a decade ago. Thankfully, I have never "had" to use your product. But there is no comparison for keeping things running at peak performance. Recently a 2TB backup drive was running slow at the start of each backup, transferring just BYTEs for several minutes before eventually returning to normal speed. I knew SpinRite would fix it. Sure enough I just backed up 40 gig in half the time took me to type this note. Thanks again for all you do! PS -- if you need a beta tester, I'm your guy!!

Closing The Loop

Subject: Not all memory leaks are important
Many years ago I worked on an SS7 Surveillance system (an equipment alarm monitoring app). After many of our UNIX machines got updated they started rebooting every 60 hours or so. It turned out that there was a bug that caused a memory leak on each unsuccessful connection attempt. We didn't know that at the time (it was a driver issue) so we turned on logging in the app in the hopes of finding out what was wrong. Needless to say the memory leak disappeared (a heisenbug). In the spirit of "Not all memory leaks are important" we just left the logging on and piped it into /dev/null. Problem mitigated.

Jerome Shidel (@RadBlasted)
Steve please help. Last time I was at my moms house, she is being bombarded by scammers and telemarketers with upwards of 20-30 a day. Calls coming from local and all over the country. She gets the mortgage, debt consolidation and IRS is filed a lawsuit final notice scams. She is on the DNC list. Other than terminating land-line service, what can we do? Any advice would be great. Thanks.

- The problem appears to be getting worse.
- CallerID is being spoofed. First it was same area code, now it's same prefix.
- And both of those tricks initially worked.
- The Do Not Call registry: https://www.donotcall.gov/
- All three of my landlines were registered on July 27th, 2003 (nearly 14 years ago).
- Cellular phones are not immune.
- I have a huge and growing list of blocked numbers on my iPhone.
- iOS apps can manage a Do Not Disturb whitelist, but cannot return SIT disconnect tones.
- What we need is a phone firewall.
- Mark Thompson setup an Asterisk system years ago. "Enter the extension number you wish to call"... and only his friends know the secret extension number.
- And, as we learned with Internet firewalls, it cannot be a "block malicious" it must be a "permit only known".
- Building a solution around an $18 VoiceModem.
  - The TAM - Telephone Answering Machine (Voice) feature set.
  - CallerID, SIT tone generation, Auto-Whitelisting, etc.

Wayne Patton (@K5UNX)
@SGgrc isn't printing out your 2nd factor QR codes sort of like writing down your passwords?? I am probably missing something??

- Yes, exactly... and writing down passwords is the officially recommended "best practice". As Bruce Schneider once brilliantly noted: "It's far more secure to use a password that you cannot remember. So write it down on paper. We already have plenty of systems available for managing bits of paper, like physical wallets." While a local physical attack DOES need to be considered, the BIG threat is the automatable remote network attack... for which strong passwords provide the best security.

Simon Zerafa (@SimonZerafa)
@SGgrc OCSP seems to be alive and well! ??

- Josh Aas (@0xjosh)
  @letsencrypt handles ~25 billion OCSP requests per month, average around 10,000 per second.

- Cause of the recent Let's Encrypt outage…
  https://community.letsencrypt.org/t/may-19-2017-ocsp-and-issuance-outage-postmortem/34922

- From 2017-05-18 17:25 UTC to 2017-05-19 06:05 UTC Let's Encrypt had a minor OCSP outage, serving HTTP 400's to a subset of OCSP clients that were making well-formed requests.

- From 2017-05-19 06:05 UTC to 2017-05-19 22:58 UTC, this became a major outage of both OCSP and the ACME API used for certificate issuance. Approximately 80% of requests failed during this phase. Most users experienced either consistent failure or consistent success.

- The initial cause was a code deploy aiming to fix a problem with slash collapsing. The OCSP protocol specifies that requests can be made via POST or GET. Since the request body is binary data, GET requests must be encoded in some ASCII-safe way. The OCSP RFC46 in 1999 predated by several years the common use of base64url encoding, which was first standardized in RFC 354813 (2003), so it defined the GET request as the base64 encoding of the binary OCSP request. Unfortunately, the base64 alphabet includes slash ("/"). Most HTTP implementations default to merging repeated slashes, which will corrupt the base64 data and generally cause decoding to fail.

- We noticed that a small number of our OCSP requests were failing due to this decoding error. Most (about 75%) of our OCSP requests arrive via POST and are unaffected; of the GET requests, only requests that encoded to base64 with a doubled slash were affected. Generally this would happen because a random serial number happened to encode that way; it's also possible for OCSP request nonces to yield a double slash when base64 decoded. For a service to correctly respond to these requests, it must not merge repeated slashes.

- Merging repeated slashes is such a common behavior that we had to disable it in three separate places: In our Go code, in our internal web server, and at our CDN. Thursday's deploy included a fix to disable slash merging in our Go code; previous updates had disabled slash merging in our web server and CDN. Unfortunately, we had missed part of RFC 696015:

- An OCSP request using the GET method is constructed as follows:

- GET {url}/{url-encoding of base-64 encoding of the DER encoding of the OCSPRequest}

- The SQRL protocol also uses both GET and POST queries... but it has always been aware of the need for URL-safe encoding, so the SQRL spec requires Base64url encoding. Whereas base64 uses '+' and '/' for the final two symbols, base64url simply replaces those with '-' and '_'. And none of those problems ever occur.

Christopher Meacham (@Meachamus_Prime)
@SGgrc , Despite #AntiTrump , the Mar-a-Lago vulnerability was not responsibly disclosed. https://www.propublica.org/article/any-half-decent-hacker-could-break-into-mar-a-lago

Hipposec (@hipposec)
@SGgrc would love to hear a tidbit on SN about becoming a security researcher. Especially transitioning from hobby to career.

- The biggest (and really only) requirement is an understanding of low-level code.

Jason Werner (@jkwphysics)
@SGgrc Just wondering what warranted such a high opinion of Edge. Seems lacking in configurability and choice, let alone privacy...

Mark Dean (@smarkdean)
Analogy needed. Hey Steve, I was wondering how you would explain this. I have a non-technical friend who is against software companies releasing their code as Open Source. He thinks that it is giving hackers access to just go into systems. I tried to tell him that having the source code in if itself does not mean hackers have the ability to hack into a system. They may know more about the details of the system but that's all (unless things like passwords are in the source...). I tried to use the analogy about encryption, where the algorithm is well known but that doesn't mean the keys or values can be easily derived, but that probably confused him more. Any suggestions on explaining how knowledge of the source doesn't automatically mean an easy hack? Thanks.

Here's an analogy:

- Not publishing the source code would be like not printing a map to get from point A to point B.

- Having a map would allow people to more easily inspect the route, and find a better solution. But NOT having a map does not PREVENT anyone from driving the route to obtain the same information. In other words... it is inherently IMPOSSIBLE to keep the map's route a secret, just as it's inherently impossible to keep computer code a secret.

- Anyone who is interested can obtain the same information... and bad guys will be motivated to do so. But publishing the map makes it easier for others without the strong nefarious motivation of exploitation to casually glance and the map and suggest improvements, and help out. :)

- And note, also, that publishing the map also doesn't guarantee that good guys will help, but it does makes it much easier and more likely that someone will.