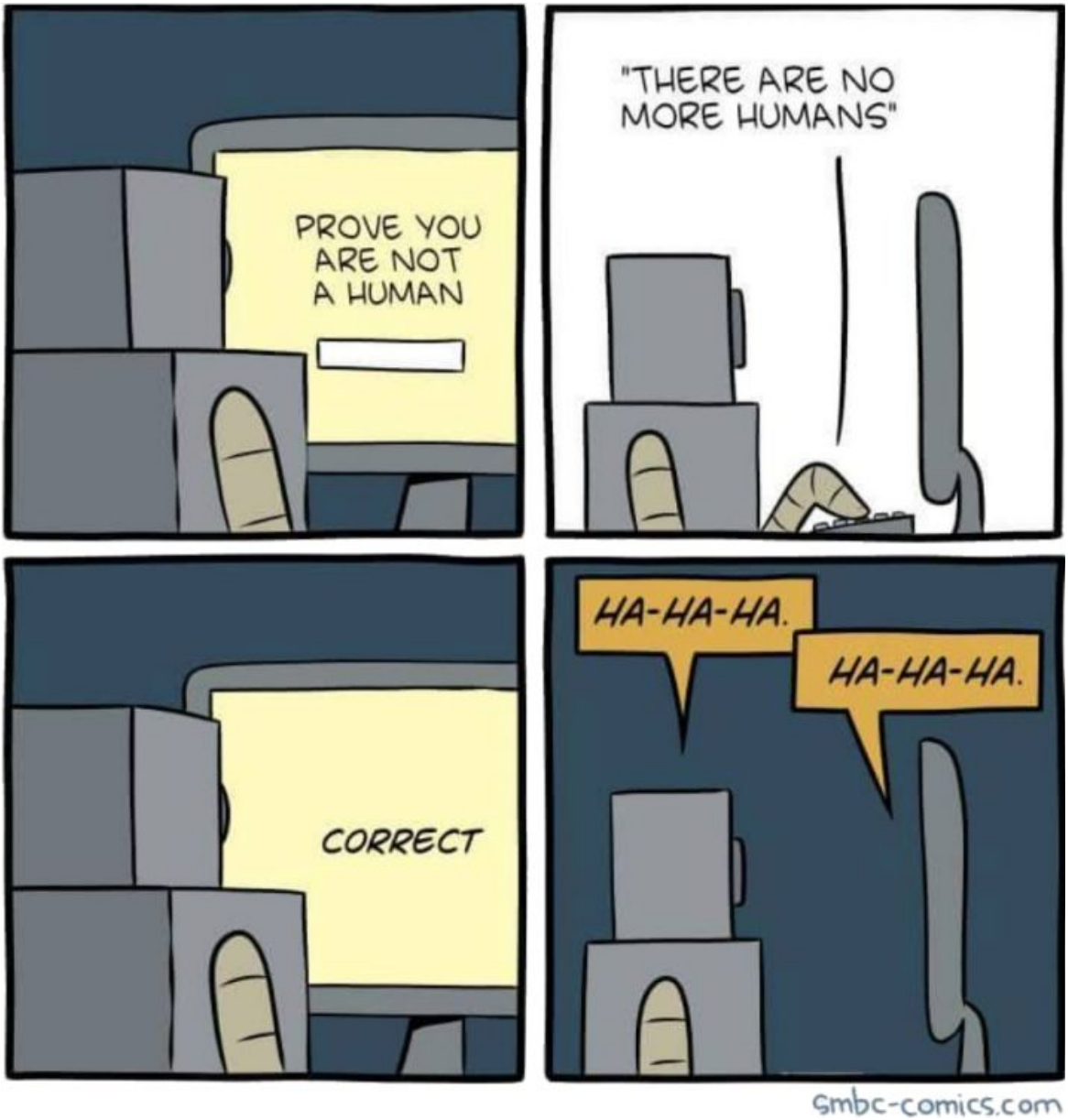
Security Now! #678 - 08-28-18

Never a Dull Moment

This week on Security Now!

This week we catch-up with another busy week. We look at Firefox's changing certificate policies, the danger of grabbing a second-hand domain, the Fortnite mess on Android, another patch-it-now Apache Struts RCE, a frightening jump in Mirai Botnet capability, an unpatched Windows 0-day privilege elevation, malware with a tricky new C&C channel, A/V companies are predictably unhappy with Chrome, Tavis found more serious problems in GhostScript, a breakthrough in contactless RSA key extraction, a worrisome flaw that has always been present in OpenSSH, and problems with never-dying Hayes AT commands in Android devices.



Security News

Firefox begins saying no to Symantec
The first nightly release track of Mozilla's Firefox has started distrusting all certificated issued by Symantec, or chained up to the Symantec CA root.

- First Firefox Nightly 63.
- Then early next month the Firefox beta release.
- The on October 23, the stable release.

As we know, this follows Google, which took the same actions with Chrome's 70 nightly build for developers toward the end of last month, on July 20.

Google will then move their "insecure connection warning" into the Chrome beta September 13,

And this will land in Chrome's stable release on October 16.

Interestingly, not all sites appear to be prepared for this to happen, even now. Remember that this transition also affects all of the smaller CA fish Symantec had subsumed during previous years, including GeoTrust, Thawte, and RapidSSL.

Since these changes will potentially be significant, and the browser vendors wish to minimize trouble for their own users, they have been monitoring the certificate chains out in the real world... and they have found that some significant sites are still using certificates which will soon become distrusted:

Those include: Sony PlayStation Store, Navy Federal Credit Union's online banking page, First National Bank of Pennsylvania's online banking, Estonian LHV Bank, Canadian telecom Freedom, La Banque Postale, La Banque Populaire Val de France, First National Bank in South Africa and Intel's Japanese website.

As we know, my own chosen CA, DigiCert, was able to acquire Symantec's certificate business in a cach and equity deal, which made sense to everyone. This jumped DigiCert from the 6th largest CA into the #3 slot by market share.

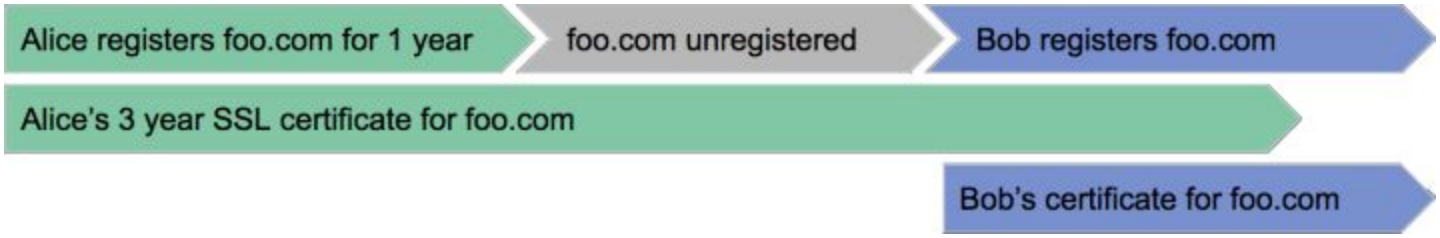
Before the deal, the top 6 ranking from largest to smaller was: Comodo, IdenTrust, Symantec, GoDaddy, GlobalSign & DigiCert.

Now it's: IdenTrust (this is because, remember, IdenTrust cross-signs Let's Encrypt's free certs), then Comodo, then DigiCert... then GoDaddy then GlobalSign.

And speaking of Certificates
... an interesting glitch in the Certificate trust model recently came to light as a result of some interesting research. "BygoneSSL" more residual fun from DEFCON.

BygoneSSL ... or perhaps "Gone but (unfortunately) not forgotten."

https://insecure.design/



Revoking:
The CA/Browser Forum, which sets the rules by which Certificate Authorities and Browser should operate, states that if any information in a certificate becomes incorrect or inaccurate it should be revoked. Additionally if the domain registrant has failed to renew their domain, the CA should revoke the certificate within 24 hours.

(I expect to hear some gasping of breath as I read this...)

Abstract:
When purchasing a new domain name you would expect that you are the only one who can obtain a valid SSL certificate for it, however that is not always the case. When the domain had a prior owner(s), even several years prior, they may still possess a valid SSL certificate for it and there is very little you can do about it.

Using Certificate Transparency, we examined millions of domains and certificates and found thousands of examples where the previous owner for a domain still possessed a valid SSL certificate for the domain long after it changed ownership. We will review the results from our ongoing large scale quantitative analysis over past and current domains and certificates. We'll explore the massive scale of the problem, what we can do about it, how you can protect yourself, and a proposed process change to make this less of a problem going forwards.

We end by introducing BygoneSSL, a new tool and dashboard that shows an up to date view of affected domains and certificates using publicly available DNS data and Certificate Transparency logs. BygoneSSL will demonstrate how widespread the issue is, let domain owners determine if they could be affected, and can be used to track the number of affected domains over time.

The researchers took a sample set of 3 million domains and 7.7 million certificates to find how many pre-dated the registration of a domain and were still valid after registration expired. 1.5 million entries fit these parameters and 25% of them had not expired at the time of the investigation.

Fortnite Android App is Vulnerable to Man-in-the-Disk Attacks
But first a bit of background on Fortnite vs Google...

Google takes a 30% cut of all revenue generated by apps downloaded through the Google Play Store... and as often been observed on the TWiT Network, Fortnite is a GOLDMINE.

Over on iOS, with a player base of over 125 million users, during the first ten days of Season 5, the iOS release netted Epic Games $2 million per day. Total mobile revenue on iOS between its March 15th release and the end of July was $150 million. Since there is no other method of distributing apps in the Apple ecosystem other than the App Store, Epic Games had no choice but to give Apple a piece of their action.

But not so on Android. Epic Games has elected to bypass Google and the Play Store and to teach how to and encourage people -- because there's no other way to obtain Fortnite -- how to sideload Fortnite from an APK loaded from Fortnite's website.

This requires instructing people to push past all of the sideload warnings present... and the big worry is that this will have the effect of normalizing this process and behavior... and pave the way for Fortnite clone malware.

Remember that we've already seen Fortnite cheating add-ons installing malware into people's devices.

And... it turns out that Fortnite =IS= vulnerable to the Man-In-The-Disk attack we talked about last week and which Check Point researchers recently made more clear.

Google's public disclosure was titled: "Fortnite Installer downloads are vulnerable to hijacking"

<quote> The Fortnite APK (com.epicgames.fortnite) is downloaded by the Fortnite Installer (com.epicgames.portal) to external storage:

```
dream2lte:/ $ ls -al
/sdcard/Android/data/com.epicgames.portal/files/downloads/fn.4fe75bbc5a674f4f9b356b5c90567da5.Fortnite/
total 73360
drwxrwx--x 2 u0_a288 sdcard_rw       4096 2018-08-15 14:38 .
drwxrwx--x 3 u0_a288 sdcard_rw       4096 2018-08-15 14:38 ..
-rw-rw---- 1 u0_a288 sdcard_rw 75078149 2018-08-15 14:38 x1xlDRyBix-YbeDRrU2a8XPbT5ggIQ.apk
-rw-rw---- 1 u0_a288 sdcard_rw       31230 2018-08-15 14:38 x1xlDRyBix-YbeDRrU2a8XPbT5ggIQ.manifest
```

Any app with the WRITE_EXTERNAL_STORAGE permission can substitute the APK immediately after the download is completed and the fingerprint is verified. This is easily done using a FileObserver. The Fortnite Installer will proceed to install the substituted (fake) APK.

On Samsung devices, the Fortnite Installer performs the APK install silently via a private Galaxy Apps API. This API checks that the APK being installed has the package name

com.epicgames.fortnite. Consequently the fake APK with a matching package name can be silently installed.

If the fake APK has a targetSdkVersion of 22 or lower, it will be granted all permissions it requests at install-time. This vulnerability allows an app on the device to hijack the Fortnite Installer to instead install a fake APK with any permissions that would normally require user disclosure.

In response, Epic has released v2.1... But Epic is unhappy with Google's short-notice public disclosure: <quote> Tim Sweeney at Epic Tweeted: "We asked Google to hold the disclosure until the update was more widely installed. They refused, creating an unnecessary risk for Android users in order to score cheap PR points."

Google defended their timing saying that their instrumentation had shown that most Fortnite users had already updated to v2.1.

But the enduring problem is the tacit endorsement and possible normalization of non Google Play sideloading...

CNET:
A storm of security issues is closing in on the Android version of Fortnite. And it isn't likely to pass anytime soon.

Developer Epic Games just fixed a security flaw with Fortnite's installer for Android devices, but researchers are expecting a flurry of problems for the online game as it gets more popular on Android.

That's because Fortnite isn't available through Google's Play Store. Epic instead chose an unorthodox -- and more dangerous -- route for the game's fans. Rather than download it through the official Google app store, players need to download the game and "sideload" the app on their Android devices instead.

That Epic is allowed to do this underscores why Google's Android often gets knocked for its security chops. While Apple locks down its iPhone so you can only download apps through its App Store, Android lets you download programs in multiple ways. But that freedom comes at a risk: Apps outside of the Play Store are nine times more likely to be malware, according to Google.

With Fortnite's influence over more than 125 million players, teaching people to download apps outside of the official store is exposing millions of people to a risky practice, researchers warned. Even if Epic means no harm, other apps may have more nefarious intentions.

Craig Williams, a security researcher and outreach director for Ciscos' Talos Intelligence Group was quoted, saying: "The problem with Fortnite is that it's so attractive, and people are going to think sideloading is completely normal." Craig said of Epic: "They've made themselves an attractive target."

And the Apache STRUTS Java framework is back in the doghouse:
First of all, let's all recall that it was not yet a ago that Equifax's failure to patch a long-known bug in Apache Struts was leveraged to expose the personal details of its 147 million consumers... at an eventual cost to Equifax of $600 million (CVE-2017-5638).

Apache Struts is widely used by enterprises globally, with last year estimates suggesting at least 65% of the Fortune 100 companies are using web applications built with the Apache Struts framework.

And now we have a new Remote Code Execution flaw which allows attackers to remotely commandeer web servers.

https://lgtm.com/blog/apache_struts_CVE-2018-11776
https://semmle.com/news/apache-struts-CVE-2018-11776

August 22nd -- Semmle Security
Today, the Apache Software Foundation announced a critical remote code execution vulnerability in Apache Struts, a popular open source framework for developing web applications in the Java programming language. Applications developed using Apache Struts are potentially vulnerable. The vulnerability (CVE-2018-11776) was identified and reported by Man Yue Mo from the Semmle Security Research Team, which works to find and report critical vulnerabilities in widely used open source software.

Organizations and developers who use Struts are urgently advised to upgrade their Struts components immediately. Previous disclosures of similarly critical vulnerabilities have resulted in exploits being published within a day, putting critical infrastructure and customer data at risk.

Mitigation:
This new remote code execution vulnerability affects all supported versions of Apache Struts 2. A patched version has been released today. Users of Struts 2.3 are strongly advised to upgrade to 2.3.35; users of Struts 2.5 need to upgrade to 2.5.17. The vulnerability is located in the core of Apache Struts. All applications that use Struts are potentially vulnerable, even when no additional plugins have been enabled.

Struts applications are often facing the public internet, and in most situations an attacker does not require any existing privileges to a vulnerable Struts application in order to launch an attack against it. To make matters worse, it is very easy for an attacker to assess whether an application is vulnerable, and it is likely that dedicated scanning tools will be available soon. Such tools will enable a malicious actor to quickly and automatically identify vulnerable applications.

Whether or not a Struts application is vulnerable to remote code execution largely depends on the exact configuration and architecture of the application. For more details, please see the section "Was I vulnerable?" below. Note that even if an application is currently not vulnerable, an inadvertent change to a Struts configuration file may render the application vulnerable in the future. You are therefore strongly advised to upgrade your Struts components, even if you believe your configuration not to be vulnerable right now.

https://cwiki.apache.org/confluence/display/WW/S2-057
Solution:

- Upgrade to Apache Struts version 2.3.35 or 2.5.17.

Backward compatibility:

- Both 2.3.35 and 2.5.17 versions contain the security fixes only, nothing more.
  No backward incompatibility issues are expected.

The Mirai Botnet gains a worrisome new trick...
By leveraging "Aboriginal Linux", Mirai is suddenly far more dangerous.

Aboriginal's slogan is prophetic: "We cross compile so you don't have to."

The new "Sora" variant of Mirai is compiled with Aboriginal Linux which offers a toolchain utility which can take its single base source and generate binaries for a large number of different platforms... many more than previous variants of Mirai could infect. Not so any longer.

Once Mirai/Sora acceses a device by guessing its SSH password, its infection routine successively downloads and executes from a long list of Sora binaries, one by one, until it finds the one which is appropriate for the infected device's platform.

As a consequence, Mirai has made a cross-species jump to Android and Debian... where Mirai has never been present before.

Windows Privilege Elevation 0-Day...
SandboxEscaper (@SandboxEscaper) tweets a github link
"Here isthe alpc bug as 0day: (github.com/SandboxEscaper...) I don't f'ing care about life anymore. Neither do I ever again want to submit to MSFT anyway. F' all this stuff." [and he didn't say stuff]

Shortly after "SandboxEscaper's" tweet, CERT/CC vulnerability analyst Will Dormann verified the authenticity of the zero-day bug, tweeting: "I've confirmed that this works well in a fully-patched 64-bit Windows 10 system. LPE right to SYSTEM!"

According to the short advisory published by CERT/CC, the zero-day flaw, if exploited, could allow local users to obtain elevated (SYSTEM) privileges. Since Local Procedure Calls are, as their name suggests, "Local", the impact of the vulnerability is limited and so it received a CVSS rating of 6.4 to 6.8. However, as we know, privilege elevation bugs are still highly sought after since they allow anything that gets into the machine to make much deeper changes.

As we know, SandboxEscaper did not notify Microsoft of the zero-day vulnerability, which leaves all Windows users vulnerable until a patch is released to address the issue. That will hopefully be the next patch Tuesday, September 11th.

I'll admit to being curious to see whether the PoC was still there on Github since Microsoft is now Github's owner... and sure enough...

A tricky new C&C channel for malware has been discovered in the wild...
Exchange PDFs by eMail! "Turla" MS Outlook backdoor

Turla has been around, but mostly dormant since first being spotted back in 2013, though some code contains timestamps from 2009 (which might not be authentic.)

https://www.welivesecurity.com/wp-content/uploads/2018/08/Eset-Turla-Outlook-Backdoor.pdf

Researchers at ESET recently found a much updated and advanced version... which transacts with its masters by encoding C&C instructions in PDFs and deleting them on-the-fly from the recipient's inbox.

While the infection is present:
- All outgoing emails are forwarded to the attackers.
- Metadata (email addresses, subject, and attachment names) of incoming emails is logged.
- Any file requested by the attackers can be sent via the backdoor.

The Backdoor can
- Execute additional programs and/or commands
- Download additional files
- Exfiltrate files
- Receive commands via PDF eMail attachments
- And is highly resilient against take-down

ESET's report states that the backdoor is a standalone Dynamic Link Library (DLL) that has code

for installing itself and interacting with the mail clients Outlook and The Bat!, even if only the installation for Outlook is implemented. Thus, it can easily be dropped by any other Turla component that is able to execute additional processes.

There is no hardcoded path so the DLL file may be located anywhere on the disk.

So here we see that a channel that's typically present and wide open on most systems is being repurposed.

The A/V companies are unhappy with Google & Chrome
https://www.bleepingcomputer.com/news/google/bitdefender-disables-anti-exploit-monitoring-in-chrome-after-google-policy-change/

BitDefender's Bogdan Botezatu, a senior e-threat analyst for Bitdefender, told Bleeping Computer's reporters that as of last Monday, August 20th, Bitdefender would no longer be monitoring Chrome 66 and later with their anti-exploit technology. Here's what Bogdan said, and his take on the whole thing:

"Starting with the Chrome browser version 66, Google has gradually rolled out a new feature that prevents third party software from monitoring the application's processes. While this measure ensures that rogue applications do not interfere with the Google product, it also prevents security solutions from inspecting the browser's memory in search of potentially dangerous exploit code.

With version 66, Google Chrome displays a post-crash warning asking users to remove the security solution if it monitors the browser's processes, even if the security solution is not responsible for the respective crash. In order to prevent this message from occurring and having users unwarily uninstall the security solution - which would leave them exposed to a vast array of online threats, Bitdefender has issued an update to stop the Anti-Exploit technology from monitoring the Chrome browser. The update was delivered to customers on August 20th at 7:00 AM ET.

As a leading global cybersecurity technology company, Bitdefender is committed to providing cutting edge end-to-end cyber security solutions and advanced threat protection to more than 500 million users in more than 150 countries. We regret being forced into removing protection for one of the world's most popular browser and we urge users to not uninstall their security solution they have installed on their computers."

Similarly:
Kaspersky Lab: "Kaspersky Lab is aware of Google Chrome showing alerts that the company's applications are incompatible with the browser. We have contacted Google to find a solution and we are continuing to look for possible workarounds to resolve this issue.

Having our code injected into the Chrome browser is an important part of the overall internet security approach implemented by security vendors to provide users with safe web surfing. For example, it is critical for a feature of Secure Input that blocks attempts of stealing sensitive data like credit card number, login, password, with malware (keyloggers) installed on user's devices."

- Kaspersky Lab

Pedro Bustamante of Malwarebytes told Bleeping Computer:
"I was going to mention that in the current implementation, Chrome doesn't actually check whether the "incompatible app" is actually causing crashes or not. They are simply taking a list of popular apps and adding them to their warnings, regardless of whether those apps introduce crashes, conflicts, or any other issues. They do this by simply looking at a registry key to see if a particular app is installed or not. They don't actually validate whether the app causes crashes.

In the case of Malwarebytes, we keep a pretty close eye on any potential issues caused by our products in Chrome, and as far as we know there aren't any currently nor have there been any for a long period of time. So we're just as puzzled as everybody else as to why Google would blanket tag us as "incompatible" even though we are 100% compatible and problem-free in all versions of Chrome.

FWIW, we are aware of other popular applications which DO hook into Chrome and DO cause crashes and conflicts regularly, but these are not included in Google's "incompatible" list of apps. Maybe there is a "friends of Google" preferential list to these warnings?

We reached out to Google through different channels, but so far we have not gotten any insightful or useful response from them. There's absolutely no logical reason for them to take this approach. They should either stop all code injection, allow only verified security vendors to inject, or at least verify if an app is actually injecting or not before labeling them "incompatible". Any of these approaches is a better option than what they're currently doing."

Avast/AVG simply replied: "We have fixed this issue and our products are not reported by Chrome."

The super-popular and widespread Ghostscript interpreter has a big problem.
Brought to us by none other than Tavis Ormandy of Google's Project Zero.

Ghostscript is =THE= industry's open-source interpreter for Adobe's PostScript and PDF page description languages. It is embedded in hundreds of applications and used by code libraries that allow desktop software and web servers to handle PostScript and PDF-based documents. Among the popular applications that use GhostScript are: ImageMagick, Evince and GIMP... and pretty much any non-Adobe software which supports PDF creation, editing, viewing or printing.

Exploiting the vulnerability allows an attacker to take over applications and servers that use Ghostscript. (No patch is available, yet.)

Tavis tweeted: "This is your annual reminder to disable all the GhostScript coders in policy.xml."
Tavis' tweet reads as it does because he has previously identified problems in 2016 and 2017.

Tavis posted to Chromium bugs: "ghostscript: multiple critical vulnerabilities, including remote command execution"
https://bugs.chromium.org/p/project-zero/issues/detail?id=1640
I sent the following mail to the oss-security mailing list:

http://seclists.org/oss-sec/2018/q3/142

These are critical and trivial remote code execution bugs in things like ImageMagick, Evince, GIMP, and most other PDF/PS tools.

Hello, this was discussed on the distros list, but it was suggested to move discussion to oss-security.

You might recall I posted a bunch of -dSAFER sandbox escapes in ghostscript a few years ago:

http://seclists.org/oss-sec/2016/q4/29

I found a few file disclosure, shell command execution, memory corruption and type confusion bugs. There was also one that was found exploited in the wild. There was also a similar widely exploited issue that could be exploited identically.

TL;DR: I *strongly* suggest that distributions start disabling PS, EPS, PDF and XPS coders in policy.xml by default.

<<snip>>

This means you can create a file in any directory (I don't think you can prevent the random suffix). Additionally, I have a trick to let you read and unlink any file you have permission to.

...and...

This can be used to steal arbitrary files from webservers that use ImageMagick by encoding file contents into the image output, see my previous PoC here for an example. i.e. You can make convert malicious.jpg thumbnail.jpg produce an image with the contents of a file visible.

These bugs were found manually, I also wrote a fuzzer and I'm working on minimizing a very large number of testcases that I'm planning to report over the next few days. I will just file those issues upstream and not post each individual one here, you can monitor https://bugs.ghostscript.com/ if you want to. I expect there to be several dozen unique bugs.

In the meantime, I really *strongly* suggest that distributions start disabling PS, EPS, PDF and XPS coders in policy.xml by default. I think this is the number one "unexpected ghostscript" vector, imho this should happen asap. IMHO, -dSAFER is a fragile security boundary at the moment, and executing untrusted postscript should be discouraged, at least by default.

("policy.xml" refers to ImageMagick's configuration file.)

Recovering RSA keys over RF in a single shot.
Under the category of "attacks never get weaker, they only get stronger"
A presentation at that recent USENIX conference two weeks ago detailed a new technique for retrieving the public key encryption keys from electronic devices... in a single shot... making it VASTLY more effective than previously known possible.

The RF recovery of keying material is not new, of course. It's been known for decades. However, previous techniques required devices to be disassembled and probes placed on chip backs... and many many hours of recording.

But in Baltimore two weeks ago, a team from Georgia State University suddenly made this style of attack practical in the real world. (This is a good lesson in why it's worth pursuing even weak vulnerabilities -- like Meltdown and Spectre.)

The paper they delivered was chillingly titled: "One&Done: A Single-Decryption EM-Based Attack on OpenSSL's Constant-Time Blinded RSA"

Abstract
This paper presents the first side channel attack approach that, without relying on the cache organization and/or timing, retrieves the secret exponent from a single decryption on arbitrary ciphertext in a modern (current version of OpenSSL) fixed-window constant-time implementation of RSA. Specifically, the attack recovers the exponent's bits during modular exponentiation from analog signals that are unintentionally produced by the processor as it executes the constant-time code that constructs the value of each "window" in the exponent, rather than the signals that correspond to squaring/multiplication operations and/or cache behavior during multiplicand table lookup operations. The approach is demonstrated using electromagnetic (EM) emanations on two mobile phones and an embedded system, and after only one decryption in a fixed-window RSA implementation it recovers enough bits of the secret exponents to enable very efficient (within seconds) reconstruction of the full private RSA key.

Since the value of the ciphertext is irrelevant to our attack, the attack succeeds even when the ciphertext is unknown and/or when message randomization (blinding) is used. Our evaluation uses signals obtained by demodulating the signal from a relatively narrow band (40 MHz) around the processor's clock frequency (around 1GHz), which is within the capabilities of compact sub-$1,000 software-defined radio (SDR) receivers.

Finally, we propose a mitigation where the bits of the exponent are only obtained from an exponent in integer-sized groups (tens of bits) rather than obtaining them one bit at a time. This mitigation is effective because it forces the attacker to attempt recovery of tens of bits from a single brief snippet of signal, rather than having a separate signal snippet for each individual bit. This mitigation has been submitted to OpenSSL and was merged into its master source code branch prior to the publication of this paper.

In our experiments we place probes very close, but without physical contact with the (unopened) case of the phone, while for the embedded system board we position the probes 20 cm away from the board, so we consider the demonstrated attacks close-proximity but non-intrusive."

OpenSSH has always had (as in for the past two decades) a worrisome security flaw.
http://seclists.org/oss-sec/2018/q3/124
Title: OpenSSH Username Enumeration

In examining an unrelated source code fix / commit, researchers at securitum.pl realized that a potentially bigger problem had also been inadvertently fixed. They wrote:

"We realized that without this patch, a remote attacker can easily test whether a certain user exists or not (username enumeration) on a target OpenSSH server:"

The attacker can try to authenticate a user with a malformed packet (for example, a truncated packet), and:

- if the user is invalid (it does not exist), then userauth_pubkey() returns immediately, and the server sends an SSH2_MSG_USERAUTH_FAILURE to the attacker;

- if the user is valid (it exists), then sshpkt_get_u8() fails, and the server calls fatal() and closes its connection to the attacker.

We believe that this issue warrants a CVE; it affects all operating systems, all OpenSSH versions (we went back as far as OpenSSH 2.3.0, released in November 2000), and is easier to exploit than previous OpenSSH username enumerations (which were all timing attacks):

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0190
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-5229
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-6210

We also believe that this should be posted to oss-security right away: the issue (commit) is already public, and if we spotted it, then others (not so well intentioned) did too. We are at your disposal for questions, comments, and further discussions.

Hayes modem AT Commands never die... unfortunately.
https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-tian.pdf

Also from the Baltimore USENIX conference, in a paper titled: "ATtention Spanned: Comprehensive Vulnerability Analysis of AT Commands Within the Android Ecosystem"

A multi-source team of 11 researchers from the Samsung Research America, Stony Brook University and the University of Florida, took a close look into the undocumented AT commands currently supported on contemporary Android devices... and what they found was surprising:

The team analyzed and reverse-engineered the firmware images of move than 2,000 Android devices from eleven Android OEMs including ASUS, Google, HTC, Huawei, Lenovo, LG, LineageOS, Motorola, Samsung, Sony, and ZTE. They discovered that these devices support over 3,500 different types of AT commands, some of which grant access to very dangerous functions.

ABSTRACT:
AT commands, originally designed in the early 80s for controlling modems, are still in use in most modern smartphones to support telephony functions. The role of AT commands in these devices has vastly expanded through vendor-specific customizations, yet the extent of their functionality is unclear and poorly documented [In other words -- deliberately undocumented]. In this paper, we systematically retrieve and extract 3,500 AT commands from over 2,000 Android smartphone firmware images across 11 vendors. We methodically test our corpus of AT commands against eight Android devices from four different vendors through their USB interface and characterize the powerful functionality exposed, including the ability to rewrite device firmware, bypass Android security mechanisms, exfiltrate sensitive device information, perform screen unlocks, and inject touch events solely through the use of AT commands. We demonstrate that the AT command interface contains an alarming amount of unconstrained functionality and represents a broad attack surface on Android devices.

And... all available through most devices' USB port.

https://atcommands.org/
https://atcommands.org/atdb/vendors

How does this affect me?
On some Android smartphones, an AT command interface is exposed over USB without USB debugging enabled. Unfortunately, some devices do not authenticate this interface or allow it to be used from the lockscreen. We found that in some cases the "charge-only" USB mode may also fail to block AT commands. This means unsuspecting users who plug in their phones to a USB port for charging or data transfer may have their devices locally compromised by a (possibly pre-recorded) sequence of AT commands. Furthermore, many commands, such as those for ex-filtrating sensitive data, have no visible side-effects.

Have you found any vulnerabilities?
Yes. We have notified each vendor of any relevant findings and have worked with their security teams to remediate the issues.

Did you find any remotely exploitable vulnerabilities?
No. All of our investigation centered on the device's USB connection. We did not investigate remote AT attack surface, but the first places we would look would be the BlueTooth interface and the baseband.

Miscellany

Ryan Dodds (@RyanDodds92) - 8/17/18, 5:22 AM

@SGgrc Steve, did you ever find the perfect coffee travel cup? I’m struggling all travel cups are usually plastic and make the coffee taste awful!

=> Contigo

SpinRite

Floris (@Floris) - 8/4/18, 4:58 AM

Just had to help a company figure out moving their super low rpm old half broken 100+ hard drives on slow usb1 and usb2 to a new thunderbolt raid system .. I have a feeling I will be using @SGgrc's Spinrite a lot next week. Time to push that company to buying a proper license.

~30~