Security Now! #704 - 03-05-19

Careers in Bug Hunting

This week on Security Now!

This week we look at a newly available improvement in Spectre mitigation performance being rolled out by Microsoft and who can try it right now, Adobe's ColdFusion emergency and patch, more problems with A/V and self signed certs, a Docker vulnerability being exploited in the wild, the end of Coinhive, a new major Wireshark release, a nifty web browser website screenshot hack, continuing troubles with the over-privileged Thunderbolt interface, bot-based credential stuffing attacks, some SQRL, miscellany, SpinRite and listener feedback... then we examine the increasing feasibility of making a sustainable career out of hunting for software bugs.

An API the was never meant to be publicly exposed:

Security News

Microsoft incorporates Google's Retpoline fix for a Spectre mitigation

As we'll recall, Intel's emergency mitigation for the Variant 2 of Spectre caused a slowdown which was especially noticeable on older Intel processors.

Google designed a clever alternative which they named "Retpoline" for "Return Trampoline." Retpoline required code-level tweaking and recompilation but promised, when it could be used, to reduce the cost of Spectre Variant 2 mitigation.

Microsoft explains:

"... Retpoline works by replacing all indirect call or jumps in kernel-mode binaries with an indirect branch sequence that has safe speculation behavior. This proves to be much faster than running all of kernel mode code with branch speculation restricted (IBRS set to 1). However, this construct is only safe to use on processors where the RET instruction does not speculate based on the contents of the indirect branch predictor. [Unfortunately, Intel's newer processors do that.] Those processors are all AMD processors as well as Intel processors codenamed Broadwell and earlier according to Intel's whitepaper. Retpoline is not applicable to Skylake and later processors from Intel."

AMD and Intel Broadwell and older processors running Windows 10's October 2018 Update 1809 can, since an update last Friday March 1st: KB4482887

https://support.microsoft.com/en-us/help/4482887/windows-10-update-kb4482887

Microsoft explains... Over the coming months, we will enable Retpoline as part of phased rollout via cloud configuration. Due to the complexity of the implementation and changes involved, we are only enabling Retpoline performance benefits for Windows 10, version 1809 and later releases. [In other words, update your Win10 to 1809 if to want it.]

And today Microsoft updated their posting: While the phased rollout is in progress, customers who would like to manually enable Retpoline on their machines can do so with the following registry configuration updates...

Adobe's Cold Fusion gets an emergency patch

Last Friday, March 1st, Adobe released an emergency patch for their Java-based ColdFusion website development platform to close a vulnerability that was being actively exploited in the wild to execute arbitrary code.

The vulnerability allowed an attacker to bypass restrictions for uploading files. To take advantage of it the website must be configured to accept executable uploads. The flaw then allows the uploaded file to be executed via an HTTP request. Whoopsie! :-/

ALL previous ColdFusion versions, on all platforms, are vulnerable to this flaw which has been designated CVE-2019-7816.

https://helpx.adobe.com/security/products/coldfusion/apsb19-14.html

Adobe: Summary

Adobe has released security updates for ColdFusion versions 2018, 2016 and 11.?These updates resolve a?critical?vulnerability that could lead to arbitrary code execution in the context of the running ColdFusion service.???

Adobe is aware of a report that CVE-2019-7816 has been exploited in the wild.

An independent consultant named Charlie Arehart discovered the bug when he discovered that it was being used against one of his clients. After figuring out what was going on, Charlie reported the flow to Adobe along with a proposed solution. To their credit (and doubtless due to the bug's extreme severity affecting all appropriately-configured ColdFusion-based websites) Adobe had the fix ready within just a few days.

Bleeping Computer interviewed Charlie, who said: "Getting folks to implement this fix is of critical importance." and Charlie was not disclosing any additional details of the attack since he didn't wish to help any attackers. However, Charlie also told Bleeping Computer that he believes that a skilled attacker will be able to connect the dots in Adobe's security bulletin and find a way to exploit the glitch.

As an interim mitigation, Adobe wrote:

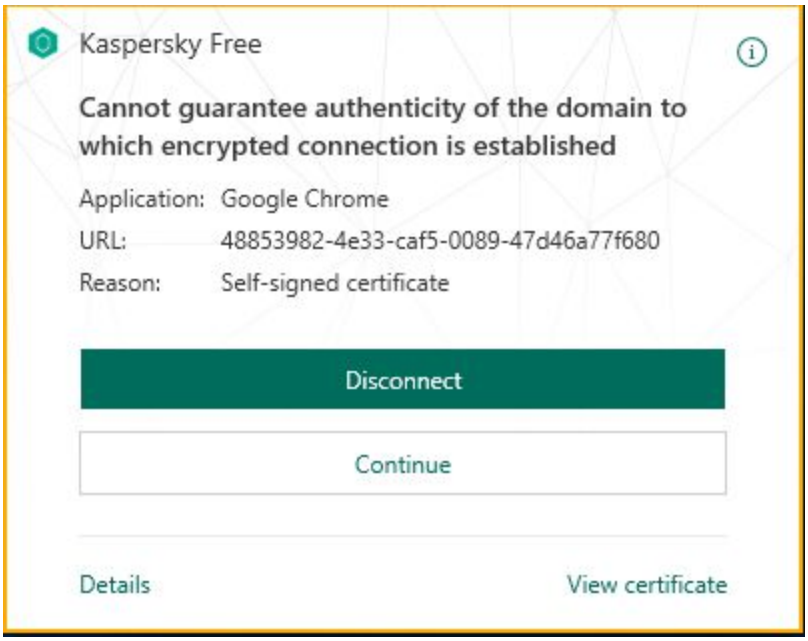
Note: This attack requires the ability to upload executable code to a web-accessible directory, and then execute that code via an HTTP request. Restricting requests to directories where uploaded files are stored will mitigate this attack.

ColdFusion 2018 (update 2 and earlier), 2016 (update 9 and earlier), and ColdFusion 11 (update 17 and prior) are susceptible to attacks.

So... our obvious takeaway is... if you or anyone you know have a ColdFusion-based website be sure to either immediately mitigate or immediately update... but definitely do something immediately.

More self-signed certificate problems... this time with Kaspersky A/V
For more than a month, since before February, Chrome users of Kaspersky's A/V whoSe system was configured to have secure connection filtering enabled (which is, of course, the default now) have been getting and complaining about mysterious pop-up complaints from Kaspersky.

https://bugs.chromium.org/p/chromium/issues/detail?id=936865



Chrome: Any version
Platform: Windows
Third Party Software: Kaspersky

There's been a sudden increase in device discovery reports. Reviewing the reports indicated that it's common on the Windows platform. And reviewing of the logs show a commonality of cast channel authentication errors, which can often be attributed to Anti Virus / security software.

In a similar time frame, some discussions were opened on Kaspersky online forums:

https://forum.kaspersky.com/index.php?/topic/408030-chrome-self-signed-certificate-cannot-guarantee-merged/&

https://forum.kaspersky.com/index.php?/topic/407318-self-signed-certificate-issue-google-docs/

I was able to reproduce the issue with Kaspersky Free. And confirmed with some external users using Kaspersky Total Security.

Repro steps:
1. Have Kaspersky software (Issue is confirmed with the free version and Total Security) installed and running on a Windows machine.
2. Have a Chromecast device connected to the same network as the computer.
3. Open Chrome.

The issue is:

1. 1. Immediately when Chrome is opened on a network with Chromecast devices, a pop up dialog appears from Kaspersky stating that "Kaspersky cannot guarantee authenticity of the domain to which encrypted content is established."

2. Even after clicking continue on the dialog, the Chromecast devices do not appear in the Cast dialog. The error does not appear if Chromecast devices are not on the same network.

3. Disabling "Scan Encrypted Connections" in Kaspersky Network settings allowed device discovery to work, and prevented the Kaspersky error dialog from appearing.

Note that in the Chromecast Help Center, there is an article titled "No Cast Destinations Found?" that lists troubleshooting steps. In this article there has been a longstanding note re Kaspersky, and that disabling "Scan Encrypted Connections" may resolve discovery issues.
https://support.google.com/chromecast/answer/3249268?hl=en

We're reaching out to Kaspersky to resolve the issue.

It turns out that the Chrome browser has some default behavior of scanning its local network for any available and listening Chromecast devices. And that Chromecast devices may be present even when they are unknown. For example, many recent smart TVs include Chromecast built in so that they're able to receive 'Casts.

Chromecast's "device discovery" service listens and accepts connections on TCP port 8009, and it establishes a TLS connection to the client connecting to it using a self-signed cert. So when Chrome is started up, it sends out a "are there any Chromecast devices out there?" local broadcast. Any powered up and online Chromecast device will head the call and reply. Then the Chrome browser will attempt to bring up a TLS connection at the responding device's IP over port 8009.

And, when Kaspersky A/V, which is monitoring that host machine's networking sees a connection being established with a self-signed cert (which is also what a malicious spoofing remote website might attempt) it freaks out, wars its user and denies the connection attempt.

Kaspersky complained that it's not easy to distinguish that event, but they have said that they'll have a fix for it shortly. My guess is that they will simply allow connections to be made from the computer to other devices on the local LAN address space to port 8009 without complaint. In other words, they'll whitelist that exact behavior.

Docker's containers are having another problem.

But we should first back up and talk about Dockers a bit (and I don't mean the pants), since we have never discussed them in any detail before and they are becoming increasingly popular and are therefore becoming an increasingly lucrative target for attack. We all understand the concept of a virtual machine, a VM, since they've been around for a long time. A Docker moves the encapsulation boundary to the other side of the OS, but I'm getting ahead of myself...

The precursor to today's operating system as we know it was the "supervisor". Big impressive looking mainframe computers were managed by a supervisor. The supervisor provided very few functions, many fewer than today's operating systems. Mostly they were loaders for the

free-standing programs that took over the machine while they were running. Over time we realized that it didn't make sense for each program to contain its own redundant code for dealing with magnetic tape and the system printer and its card punch. So reusable common libraries were created. And as machines grew in memory capacity those libraries became more often resident. And with additional evolution they grew into services that could be counted upon to be offered by the system itself. Programs then called upon those resident libraries, and thus we arrived at today's operating system.

In a Virtual Machine environment we have a so-called "Hypervisor." The hypervisor takes advantage of the amazingly complex hardware features of processors to create isolated abstractions of the processor itself. Thus, virtual machines. And once you have an abstraction of a processor -- a virtual machine -- you boot an operating system on that VM to create an instance of a system which can then run operating system client software.

But think for a moment how expensive this is in a cloud computing environment. Say that a given hardware system wants to run six separate tasks. Taking the VM approach the system's RAM is divided up into six partitions, an operating system instance is booted into each one, and then each one is given a task to run. The flexibility this offers is that the hardware could be simultaneously running six different operating systems, each with their own task. But the reality of today's computing environment is that more stuff in the cloud is a Unix or Linux. So booting up six redundant copies of a Linux VM on a cloud computing hardware instance is very wasteful.

Remember when we were talking about RowHammer attacks and how practical VM environments worked very hard to consolidate identical regions of memory. If you have six copies of Linux VMS running a great deal of their memory will be the same. The virtualization hardware allows those duplicate regions to be consolidated so that each VM sees its own memory and isn't aware that it's actually being shared among other instances.

Anyway... under this original VM-based cloud computing model, the encapsulation was a VM -- meaning the OS and an often customized environment with libraries and databases and services that the running task required was configured and packaged up. And each VM was different because the environments required differed. One way to think of this is as "low level" encapsulation where the "capsule" runs, standalone, on a virtual machine.

As I mentioned above, the Docker model moves this encapsulation boundary much higher up -- to the other side of the OS: Whereas in a VM model the encapsulation boundary was between the virtual machine hardware and the VM's OS, Docker places the boundary ABOVE the OS, at the OS service level.

A VM contains an operating system, but a Docker container does not. The Docker container runs ON or above an operating system to which has been added a Docker interface API. The Docker container encapsulates all of the various library and service dependencies and requirements of the task that the Docker is intended to perform.

In a cloud computing environment they are growing increasingly popular since they offer a much more efficient sharing of a hardware instance's resources. The cloud computer has a single highly tuned instance of an OS running with a Docker API that allows it to host and run many independent instances of Docker containers.

Okay...

The Docker container resident runtime module is known as "runc" (run container). runc is an open source command line utility designed to spawn and run containers and, at the moment, it is used as the default runtime for containers with Docker, containerd, Podman, Kubernetes, LXC, etc.

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-5736

From earlier this year we have CVE-2019-5736: *"runc through 1.0-rc6, as used in Docker before 18.09.2 and other products, allows attackers to overwrite the host runc binary (and consequently obtain host root access) by leveraging the ability to execute a command as root within one of these types of containers: (1) a new container with an attacker-controlled image, or (2) an existing container, to which the attacker previously had write access, that can be attached with docker exec. This occurs because of file-descriptor mishandling, related to /proc/self/exe."*

What this means is that a container breakout security flaw has been discovered in the runc container runtime which allows malicious containers (with minimal user interaction) to overwrite the host runc binary to gain root-level code execution on the host machine.

Aleksa Sarai, the senior software engineer for Containers at SUSE Linux is the maintainer of the runc command line utility.

Date: Tue, 12 Feb 2019 00:05:20 +1100
From: Aleksa Sarai <cyphar@...har.com>
To: oss-security@...ts.openwall.com
Cc: dev@...ncontainers.org, security-announce@...ncontainers.org
Subject: CVE-2019-5736: runc container breakout (all versions)

[[Patch CRD: 2019-02-11 15:00 CET]]
[[Exploit Code CRD: 2019-02-18 15:00 CET]]

Hello,

I am one of the maintainers of runc (the underlying container runtime underneath Docker, cri-o, containerd, Kubernetes, and so on). We recently had a vulnerability reported which we have verified and have a patch for. In addition, Aleksa Sarai (me) discovered that LXC was also vulnerable to a more convoluted version of this flaw.

== OVERVIEW ==

The vulnerability allows a malicious container to (with minimal user interaction) overwrite the host runc binary and thus gain root-level code execution on the host. The level of user interaction is being able
to run any command (it doesn't matter if the command is not attacker-controlled) as root within a container in either of these contexts:

  * Creating a new container using an attacker-controlled image.
  * Attaching (docker exec) into an existing container which the
      attacker had previous write access to.

This vulnerability is *not* blocked by the default AppArmor policy, nor by the default SELinux policy on Fedora[++] (because container processes appear to be running as container_runtime_t). However, it *is* blocked through correct use of user namespaces (where the host root is not mapped into the container's user namespace).

Our CVSSv3 vector is (with a score of 7.2):

  AV:L/AC:H/PR:L/UI:R/S:C/C:N/I:H/A:H

The assigned CVE for this issue is CVE-2019-5736.

[++]: This is only the case for the "moby-engine" package on Fedora. The
    "docker" package as well as podman are protected against this
    exploit because they run container processes as container_t.

== PATCHES ==

I have attached the relevant patch which fixes this issue. This patch is based on HEAD, but the code in libcontainer/nsenter/ changes so infrequently that it should apply cleanly to any old version of the runc
codebase you are dealing with.

Please note that the patch I have pushed to runc master[1] is a modified version of this patch -- even though it is functionally identical (though we would recommend using the upstream one if you haven't patched using the attached one already).

== NON-ESSENTIAL EXPLOIT CODE ==

Several vendors have asked for exploit code to ensure that the patches actually solve the issue. Due to the severity of the issue (especially for public cloud vendors), we decided to provide the attached exploit
code. This exploit code was written by me, and is more generic than the original exploit code provided by the researchers and works against LXC (it could likely be used on other vulnerable runtimes with no
significant modification). Details on how to use the exploit code are provided in the README.

As per OpenWall rules, this exploit code will be published *publicly* 7 days after the CRD (which is 2019-02-18). *If you have a container runtime, please verify that you are not vulnerable to this issue
beforehand.*

== IMPACT ON OTHER PROJECTS ==

It should be noted that upon further investigation I've discovered that LXC has a similar vulnerability, and they have also pushed a similar patch[2] which we co-developed. LXC is a bit harder to exploit, but the same fundamental flaw exists.

After some discussion with the systemd-nspawn folks, it appears that they aren't vulnerable (because their method of attaching to a container uses a different method to LXC and runc).

I have been contacted by folks from Apache Mesos who said they were also vulnerable (I believe just using the exploit code that will be provided). It is quite likely that most container runtimes are
vulnerable to this flaw, unless they took very strange mitigations before-hand.

So where are we today? You can all probably guess... The 'runc' container breakout flaw was immediately patched the same day by all of the big cloud service provider vendors -- Amazon, Google, and Docker, etc. But, Docker is popular and is also in use by many smaller operations and individual entities. So there remain, three weeks later, thousands of exposed Docker daemons publicly online and, yes, left unpatched.

At the time of the vulnerability disclosure there were approximately 3,951 Docker daemons exposed and a recent update by Imperva security showed that around 4,042 were currently reachable:

The Docker API is accessible through the network, but it's listening service is only supposed to be bound to the localhost 127.0.0.1 IP on ports 2375 and 2376. However, a Shodan search revealed 3,968 public instances of 2375 port open and 74 public instances of port 2376. And, not very surprisingly, only 103 of the exposed Docker daemons revealed by Shodan have been updated to the patched 18.09.2 version or later, leaving 3939 exposed to exploitation.

The Imperva guys dug a bit deeper, testing to see which of the hosts exposed by the Shodan search can actually be accessed on port 2375. They found that out of the nearly 4,000 IPs that were apparently exposed by the Shodan search engine, about 400 of them were responding.

And... on the unpatched servers that were accessible because they had the Docker API exposed to remote connections they found Docker images of crypto miners, as well as legitimate services and production environments.

Imperva summed it up, yesterday, in their disclosure: "Hundreds of Vulnerable Docker Hosts Exploited by Cryptocurrency Miners"

https://www.imperva.com/blog/hundreds-of-vulnerable-docker-hosts-exploited-by-cryptocurrency-miners/

<quote> *Docker is a technology that allows you to perform operating system level virtualization. An incredible number of companies and production hosts are running Docker to develop, deploy and run applications inside containers.*

You can interact with Docker via the terminal and also via remote API. The Docker remote API is a great way to control your remote Docker host, including automating the deployment process, control and get the state of your containers, and more. With this great power comes a great risk — if the control gets into the wrong hands, your entire network can be in danger.

In February, a new vulnerability (CVE-2019-5736) was discovered that allows you to gain host root access from a docker container. The combination of this new vulnerability and exposed remote Docker API can lead to a fully compromised host.

Coinhive shutting down at the end of this week!
https://coinhive.com/blog/en/discontinuation-of-coinhive

Blog » Discontinuation of Coinhive
Some of you might have anticipated this, some of you will be surprised. The decision has been made. We will discontinue our service on March 8, 2019. It has been a blast working on this project over the past 18 months, but to be completely honest, it isn’t economically viable anymore.

The drop in hash rate (over 50%) after the last Monero hard fork hit us hard. So did the “crash“ of the crypto currency market with the value of XMR depreciating over 85% within a year. This and the announced hard fork and algorithm update of the Monero network on March 9 has lead us to the conclusion that we need to discontinue Coinhive.

Thus, mining will not be operable anymore after March 8, 2019. Your dashboards will still be accessible until April 30, 2019 so you will be able to initiate your payouts if your balance is above the minimum payout threshold.

Thank you all for the great time we had together.

So.... out with Coinhive, in with.... whatever takes its place.

This will put a kink in the Cryptojacking enterprise, but I doubt this will be the end of it altogether. Recall that Coinhive had monthly "revenue" of approximately $250,000 at one point and in terms of "market reach", enjoyed a 62% share of all websites using a JavaScript cryptocurrency miner -- whether with or more likely without their users' knowledge.

And, as we know, Coinhive-driven cryptojacking campaigns infested Android apps in the Google Play Store and managed to infect 200,000 MikroTik routers in several campaigns.

So... MikroTik routers world wide will be running a bit cooler and will be pulling a bit less power... until someone steps up to take Coinhive's place. And it's hard to imagine that won't happen. Cryptocurrency mining is attempting to remain hostile to ASIC-powered mining. And that means that CPU mining has a place mining currencies where ASICs have been made unwelcome.

So stay tuned... You know we'll have the new of it when it happens!

We have a new and worthy Wireshark release: v3.0.0

- Windows Installer 64- and 32-bit
- Windows PortableApps® 32-bit
- macOS 10.12 and later Intel 64-bit .dmg
- Source Code (and part of Unix & Linux distros)

For those who don't already know, Wireshark is =the= go-to utility for capturing and analyzing packetized network traffic. I have been using it for years and things like ShieldsUP! and especially the quite complex DNS Spoofability test service would have been far more difficult to stand up if it were not for Wireshark.

It began way back in 1998 and was originally named Ethereal. The project was later renamed Wireshark in May 2006 to sidestep trademark issues. Wireshark always relied upon the similarly venerable WinPcap packet capture driver. ShieldsUP! originally used the same driver until I wrote my own kernel driver so that I could do more of the work in Ring-0.

The big change is that with last Thursday's release of Wireshark 3.0.0, WinPcap has finally been abandoned in favor of a new driver, Npcap. Npcap is an NDIS v6.0 filter shim driver, meaning that it inserts itself neatly into the network stack in such a way that it's able to watch and inject network traffic without needing to worry about any adapter-specific details since those are handled by the lower layers. The Npcap driver is also EV signed so that the latest Win10 systems will gracefully allow it to slip in between their layers. (what?!)

v3.0.0 also, for the first time, can capture localhost loopback traffic, making it very handy for watching and analyzing all of the goings-on within a system.

And one of Wireshark's very slick features, which it's had for a while now, is the ability to sniff and decrypt TLS traffic... when the user is able to provide the server's private key. This comes in VERY handy in today's world where unencrypted communications is now the exception.

Wireshark's use of the Npcap driver means that it's also able to capture 802.11 WiFi traffic from the air. This used to require an AirPcap hardware radio dongle. But no more.

And... Wireshark now "knows" about many more network protocols, This makes it easy to see what's going on since Wireshark breaks everything apart into nicely labeled fields.

Taking a ScreenShot of a webpage from the Command Line...
This was a sort of random tip from Bleeping Computer's founder Lawrence Abrams. I'm not sure how it might be used, but I will tuck it away in my memory and I suspect that many of our listeners will too, and might find it useful:

It turns out that both FireFox and Chrome can be invoked from the command line to launch in a "headless" mode without any UI, fully download and invisibly render a webpage, and write the webpage's image to an image file.

https://www.bleepingcomputer.com/news/software/chrome-and-firefox-can-take-screenshots-of-sites-from-the-command-line/

"[path_to_chrome]" --headless --screenshot="[path_to]\image.png" "[url]"

The screenshots will only make an image of a portion of the requested web page (it's unclear what the default size would be). So if you wish to create a screenshot of the entire web page, you should add the --window-size=width,height parameters.

Also, Chrome will show the scrollbars, which is probably not what anyone wants. So to remove the scrollbars add the --hide-scrollbars term to the command.

And Firefox can do the same with similar options. Larry indicated that in testing these he found that Firefox was both easier to use and faster to snap and store pages.

So... while it's not immediately obvious how such a facility might be used, I'll bet someone in our audience is going "Holy crap! That's exactly what I've been needing!!"

The over-privileged Thunderbolt interface remains a problem
Thunderclap: Exploring Vulnerabilities in Operating System IOMMU Protection via DMA from Untrustworthy Peripherals

https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_05A-1_Markettos_paper.pdf

Abstract—Direct Memory Access (DMA) attacks have been known for many years: DMA-enabled I/O peripherals have complete access to the state of a computer and can fully compromise it including reading and writing all of system memory. With the popularity of Thunderbolt 3 over USB Type-C and smart internal devices, opportunities for these attacks to be performed casually with only seconds of physical access to a computer have greatly broadened. In response, commodity hardware and operating-system (OS) vendors have incorporated support for Input-Output Memory Management Units (IOMMUs), which impose memory protection on DMA, and are widely believed to protect against DMA attacks. We investigate the state-of-the-art in IOMMU protection across OSes using a novel I/O-security research platform,and find that current protections fall short when faced with a functional network peripheral that uses its complex interactions with the OS for ill intent. We describe vulnerabilities in macOS,FreeBSD, and Linux, which notionally utilize IOMMUs to protect against DMA attackers. Windows uses the IOMMU only in limited cases. and it remains vulnerable. Using Thunderclap, an open-source FPGA research platform that we built, we explore new classes of OS vulnerability arising from inadequate use of the IOMMU. The complex vulnerability space for IOMMU-exposed shared memory available to DMA-enabled peripherals allows attackers to extract private data (sniffing cleartext VPNtraffic) and hijack kernel control flow (launching a root shell)in seconds using devices such as USB-C projectors or power adapters. We have worked closely with OS vendors to remedy these vulnerability classes, and they have now shipped substantial feature improvements and mitigations as a result of our work.

Firewire.

Operating system	Build/ kernel	Can use IOMMU	Default enabled	IOMMU page mappings		Vulnerability				
				Shared	Per-device	Data leakage	Kernel pointer	Shared-allocator	Spatio-temporal	ATS
Windows 7						✓	✓	✓	n/a	
Windows 8.1	9200					✓	✓	✓	n/a	
Win 10 Home/Pro 1709	16299					✓	✓	✓	n/a	
Win 10 Enterprise 1607	14393	✓		✓		✓	✓	✓	n/a	?
Win 10 Enterprise 1703	15063	✓		✓		✓	✓	✓	n/a	?
MacOS 10.10-10.13		✓	✓	✓		✓	< 10.12.4[1]		n/a	
Linux: Ubuntu 16.04	4.8/10	✓			✓	✓		✓	✓	✓
Linux: Fedora 25	4.8	✓			✓	✓		✓	✓	✓
Linux: RHEL 7.1	3.10	✓			✓	✓		✓	✓	✓
FreeBSD 11	11	✓			✓	✓	✓		✓	
PC-BSD/TrueOS 10.3	10.3	✓			✓	✓	✓		✓	

[1] Fixed after our disclosure.

Bot-Based Credential Stuffing

Once upon a time we had what is now a quaint image of a hacker in his basement repetitively trying to log into some target victim's account by guessing their password, typing in candidate after candidate, one at a time, over and over... until "HA!... what-do -you-know! I'm in!"

That evolved into an automated brute force attack against someone, first running through dictionaries of commonly used passwords and keyboard keystroke walks, and eventually getting down to trying every possible password.

Then we have website database breaches where hundreds of thousands of username and (hopefully) hashed passwords were disclosed. The bad guys would then use high-speed ASIC-based hashing rigs to "reverse" the hashes back into their original textual input for use in impersonation attacks.

And now today we have the latest evolution of this with the so-called "Credential Stuffing" attacks where great fleets of Bots, increasingly composed of code loaded into compromised consumer routers, are fanning out across the Internet, only to self-replicate, but to then launch patient and widely distributed username and password guessing attempts against Internet-facing websites.

https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/state-of-the-internet-security-retail-attacks-and-api-traffic-report-2019.pdf

All three of our stories in this issue of the State of the Internet / Securityreport are about things most organizations aren’t examining. Whether the cause is that organizations don’t perceive some issues as important to their environment, if they don’t have tooling to monitor these issues,or if the resources to monitor this traffic are not available, this traffic is often being overlooked.

Although organizations examine the traffic generated by botnets, without specialized tools that traffic is often treated the same as any other type of network activity. There are very few places where this is more dangerous than in the retail sector, where botnet creators and retail defenders are playing a multidimensional game, with real money on the line. Our team looked at All-In-One (AIO) bot tools and considered them in the context of the billions of credential abuse attempts we see on a monthly basis.

TOOLS OF MASS (RETAIL) DESTRUCTION

Between May 1 and December 31, 2018, there were 10,000,585,772 credential stuffing attempts in the retail industry detected on Akamai’s network. When that’s expanded to all other customer industries, Akamai detected 27,985,920,324 credential abuse attempts over eight months. That works out to more than 115 million attempts to compromise or log in to user accounts every day.

The reason for these attempts isn’t complex. The malicious actors responsible for them are looking for data — such as personal information, account balances, and assets — or they’re looking for opportunities to cash in on the online retail market that’s expected to hit $4.88 trillion by 2021.

The credential stuffing attempts logged by Akamai are automated, thanks to bots. Bots can represent up to 60% of overall web traffic, but less than half of them are actually declared as bots — making tracking and blocking difficult. This dilemma is compounded by the fact that not all bots are malicious.

Play the Numbers

For criminals, credential stuffing attacks are a numbers game. They're counting on the fact that people recycle their passwords across different accounts. When this happens, a compromised set of credentials from one website quickly translates into dozens of others.

It's a two-step process; stuff the login page with the maximum amount of credential pairs to verify their validity, and once verified, take control of the compromised account. This second stage is commonly known as account takeover, or ATO.

Consider the 116 million accounts compromised during the LinkedIn data breach. Using this list of email address and password combinations, criminals targeted dozens of other websites in hopes that people were using their LinkedIn credentials elsewhere. These credential stuffing attempts led to several secondary account takeovers. This is why security professionals stress the use of password managers, as well as the use of long and unique password strings for each website.

Fighting Credential Stuffing Attacks Is an Uphill Battle

The battle against credential stuffing isn't an easy one to fight. When asked, 71% of the respondents to an Akamai survey conducted by Ponemon Institute said that preventing credential stuffing attacks is difficult because fixes that prevent such action might diminish the web experience for legitimate users.

On average, organizations report experiencing 12.7 credential stuffing attempts each month, with each attempt targeting 1,252 accounts.The reflexive action to just block the bots responsible for these attempts outright makes sense at first, but such a move might cause serious harm to the business if legitimate customers are impacted.

The same survey revealed 32% of respondents lacked visibility into credential stuffing attacks, and 30% said they were unable to detect and mitigate them. When asked if their organization had sufficient solutions and technologies for containing or preventing credential stuffing attacks, 70% of those responding said their organization was lacking when it came to such defenses.

Credential stuffing attacks are a costly battle to fight as well. The survey determined that the baseline costs associated with such attacks, when considering application downtime, loss of customers, and IT overhead, amounted to annual totals of $1.7 million, $2.7 million, and $1.6 million, respectively.

SQRL

CosmaP

Hi, My phone committed suicide yesterday. :(

Fortunately, my provider (EE in the UK) was on the ball and I received a replacement today (Great Customer services win from EE). :)

Long story, short, reinstalled (well I am still reinstalling) all the apps and came to the SQRL app.

Installed the app, no problem.
Imported my Identity, no problem.
Entered the Recovery Code, no problem.

Signed into the forums using my SQRL password, no problem.
All smooth as silk and I am back operational.
It helped that I have all the required info in one place.

Jobs a goodun.
Regards.
Cosma

Jeff Root

SQRL's "friction' is it's _lack_ of friction

I've had the SQRL client on my Android phone for a while, but nothing to use it with. After watching SN703, I decided to try out the Forum login.

And it was...anti-climatic.

You've been teaching us for years that convenience and security are opposites. The entire security community has been in agreement on that. But SQRL, by being incredibly convenient and easy, appears _insecure_ because of that.

And so, the biggest impediment to SQRL adoption may be that SQRL has zero friction.

Note that SQRL has two very different attributes: security and ease-of-use.

Call for Android App UX assistance:
https://www.grc.com/sqrl/feedback.htm

Miscellany

Matt in London
Subject: Duesy is Duesenberg?
Date: 25 Feb 2019 10:32:02
:
Hey Steve,

I heard that a Duesy is named after the Duesenberg car that was so expensive that no one could afford one. Hence slang for a magnificent failure.

SpinRite

Ralph in New York City
Subject: Spinrite- still working after all these years
Date: 26 Feb 2019 10:08:17
:
I have a LAN with two wifi hi def security cameras on the 2.4GHz band. They are recording to a USB3 120GB SSD plugged into the router. Both cameras frequently stream together but many times only one or the other would record a file, and there were random freezes on many of the files during playback. I was suspecting a bandwidth issue until I ran a level 4 pass of Spinrite on the SSD. Watching the real time screen I could see random pauses, retries on reads and writes. After Spinrite completed both cameras happily record at the same time. This won't be a surprise, but Spinrite REALLY works.

Closing The Loop

Fresher in the UK
Subject: Where the f*** is Jeff‘s iOS SQRL client?
:
Mentioned in the last Security Now. Where is it? It‘s not in the App Store so what are we supposed to do? You didn’t make it US only, I hope?

Neil Taneja in Chandler, AZ
Subject: Drive mounting/unmounting
:
So you mentioned a few Security Now's ago about how you mount and unmount drives automatically for your backups to protect them. How are you doing that?

MOUNTVOL <--- lists all devices but Volume GUID.
MOUNTVOL Y:\ \\?\Volume{bf67527f-cc58-11e5-ad50-806e6f6e6963}\
MOUNTVOL Y:\ /D

A Career in Bug Bounty Hunting

19 year old Santiago Lopez has earned more than $1 Million USD from finding and reporting security vulnerabilities through the HackerOne bug bounty program.
https://twitter.com/santi_lopezz99

https://www.bbc.com/news/av/technology-47407609/how-one-teenager-is-making-millions-by-hacking-legally

Last Friday, March 1st, the BBC ran an interview with Santiago:

How one teenager is making millions by hacking legally

This is 19-year-old Santiago Lopez from Argentina. He's the first millionaire bug-bounty hacker, which means he gets paid to find glitches in the software of some of the world's biggest companies. Mr Lopez made his money on the world's biggest ethical hacking platform: HackerOne. BBC News' Joe Tidy has been to see how he spends the money.

HackerOne:
https://www.hackerone.com/blog/Brace-yourself-50-Million-Bounties-Coming-and-we-are-celebrating-whole-way-there

On February 1st, HackerOne posted: *"Today the HackerOne community hit 45 million in bounty payouts, join us as we celebrate the hackers who are making the internet a safer place every single day. The party is going to last the whole way to a history-making* ***$50 million*** *in bounty" payouts."*
https://www.hackerone.com/sites/default/files/2019-02/the-2019-hacker-report.pdf

"HACK'ER" /'ha–ker/ noun
One who enjoys the intellectual challenge of creatively overcoming limitations.

Welcome to the age of the hacker. Hackers are heroes, they are in it for the good and there is more opportunity than ever before. We share some of their stories and celebrate their impact in this, the third annual Hacker Report.

The Hacker Report details the more than 300,000 individuals that represent our hacker community today.It highlights where hackers live, what motivates them, what their favorite hacking targets & tools are, where they learn, why they collaborate and much more.

In 2018 alone, Hackers earned over $19 million in bounties, almost the entire amount awarded in the years prior combined. And while the most successful find it very lucrative, it's about so much more than money. Many are finding career building opportunities through bug bounties, with companies hiring from within the hacker community at a faster clip than ever before. Companies are utilizing bug bounty reports and hacker engagement as an enhanced resume of proven skills that will impact company goals and security efforts from day one.

The generosity and camaraderie of hackers continues to impress with more emphasis than ever before on education, collaboration, and giving back.

As hacking grows in popularity, training continues to be a focus. With more than 600 hackers registering to join the ranks any given day, in depth training modules such as Hacker 101 capture the flag challenges are in-demand.

This past year we saw incredible individual performances such as hackers earning $100K for one vulnerability and the first hacker passing the $1 million milestone. We also saw unmatched collaboration, like hackers acting as teams to report over 250 valid customer vulnerabilities.

Hackers represent a global force for good, coming together to help address the growing security needs of our increasingly interconnected society. The community welcomes all who enjoy the intellectual challenge to creatively overcome limitations. Their reasons for hacking may vary, but the results are consistently impressing the growing ranks of organizations embracing hackers through hacker-powered security—leaving us all a lot safer than before.

Hacker-powered security is creating opportunities across the entire globe. ***Top earners can make up to 40x the median annual wage of a software engineer in their home country respectively.***

Hacker training continues to take place outside of the traditional classroom, as 81% say they learned their craft mostly through blogs and self-directed educational materials like Hacker 101 and publicly disclosed reports. While just 6% have completed a formal class or certification on hacking.

Hacking for good is growing in popularity as nearly two thirds of Americans (64%) today recognize that not all hackers act maliciously according to recent Harris Poll data. (~30~)