Security Now! #719 - 06-18-19

Exim Under Siege

This week on Security Now!

There were many significant stories this week, which made picking our title story somewhat tricky. But II came up with a rationale. We have a new DRAM problem called "RAMBleed", news of a bad Linux server kernel crashing flaw, and the occurrence of the expected attacks on EXIM Mail servers. Not to mention last week's patch Tuesday, a Bluetooth surprise, another useless warning about the BlueKeep vulnerability, Microsoft missing a 90-day Tavis Ormandy deadline, the good news GandCrab wrapup, Yubico's entropy mistake, a bit of post-announce SQRL news, a favorite iOS security app, and then our title story: Having settled upon the attacks on Exim Mail servers... so that we can talk about the other disasters, which are still pending... next week!

SQRL - Secure Quick Reliable Login (Premium T-Shirt)

$20.34

✓prime

SQRL - Secure Quick Reliable Login (Simply Secure T-Shirt)

$20.34

✓prime

SQRL - Secure Quick Reliable Login (Standard T-Shirt)

$18.36

✓prime

SQRL - Secure Quick Reliable Login (Don't need Passwords)

$20.34

✓prime

SQRL - Secure Quick Reliable Login (Long Sleeve T-Shirt)

$23.26

✓prime

SQRL - Secure Quick Reliable Login

$20.34

✓prime

SQRL - Secure Quick Reliable Login (Long Sleeve T-Shirt)

$23.26

✓prime

SQRL - Secure Quick Reliable Login (Simply Secure T-Shirt)

$20.34

✓prime

https://www.grc.com/sqrl.htm

Security News

Last week was Patch Tuesday

Last Tuesday, patch Tuesday, Microsoft fixed 88 vulnerabilities, more than one quarter of which -- 21 in total -- were rated Critical. Among the remaining 67 non-Critical vulnerabilities fixed were the four deemed "Important" which SandboxEscaper found and irresponsibly publicly released, as we have previously described in some detail. All four of those were various elevation of privilege hacks. (And, last we heard, she still had one more 0-day up her sleeve which she was promising or threatening to disclose.)

Of the Critical vulnerabilities, 8 were located in Microsoft's JavaScript Chakra Scripting Engine, 5 others in their Edge browser scripting engine. All 13 of those scripting problems were memory corruption, though we know that's where remote code execution vulnerabilities begin. And speaking of remote code execution vulnerabilities, there were 3 of those fixed in Hyper-V, one in Microsoft's Speech API, another in ActiveX Data Objects (ADO), and also a critical Adobe Flash security update.

The "Important" vulnerabilities spanned pretty much everything, from the Microsoft Scripting Engine, Internet Explorer, Edge, Windows App Platform and Frameworks, Windows Input and Composition, Media, Shell, Server, Authentication, Cryptography, Datacenter Networking, Storage and Filesystems, SQL components, Microsoft's JET Database Engine, Windows Virtualization, the Kernel, and their IIS web server.

So... yeah... another mega Patch Tuesday, which we're now growing used to. And we'll be keeping a look out for SandboxEscaper to see whether she follows through with her previously threatened 0-day.

June gets a Bluetooth Stack update...

However... one other "Security Update" thing did stand out a bit. After applying this month's latest security update, as Microsoft puts it: "... the affected platforms will experience the new behavior:"

Okay, now, remember, Leo, many many moons ago, when Intel pushed crude, conceptual, sample source code for the new, at that time, UPnP functionality and, though they never intended this to happen, many router vendors simply copied and pasted that engineering source code sample right into their routers and compiled it. The result was a surprisingly widespread vulnerability across router brands, since everyone has used something that was never intended to actually be put into production code.

Well... it turns out that something kind of similar has happened again. The BLE -- the Bluetooth Low Energy -- specification provides some =SAMPLE= LTKs (Lone Term Keys) which were provided in the specification ONLY for illustration and were, of course, NEVER intended to be actually be used in practice. This would be like building the same private key into all web servers. It's nuts. Or, as CVE-2019-2102 puts it...

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-2102

In the Bluetooth Low Energy (BLE) specification, there is a provided example Long Term Key (LTK). If a BLE device were to use this as a hardcoded LTK, it is theoretically possible for a proximate attacker to remotely inject keystrokes on a paired Android host due to improperly used crypto. User interaction is not needed for exploitation. Product: Android. Versions: Android-7.0 Android-7.1.1 Android-7.1.2 Android-8.0 Android-8.1 and Android-9.

https://support.microsoft.com/en-us/help/4507623/some-bluetooth-devices-may-fail-to-pair-or-connect-after-applying-june

Microsoft titled their knowledgebase article: "Some Bluetooth devices may fail to pair or connect after applying June 11, 2019 or later updates."

You may experience issues pairing, connecting or using certain Bluetooth devices after installing security updates released June 11, 2019. These security updates address a security vulnerability by intentionally preventing connections from Windows to unsecure Bluetooth devices. Any device using well-known keys to encrypt connections may be affected, including certain security fobs.

In other words, to protect everyone, Microsoft's June update has added awareness to their Bluetooth Stack of the Long Term Key that was provided as an =example= in the BLE specification and will henceforth refuse to pair with any BLE device which attempts to offer Windows that particular key.

TCP SACK flaws discovered in Linux

We were recently talking about the seemingly nutty idea of China rolling their own Internet-connected Desktop OS from scratch, because a modern OS has become so unbelievably complex. When drawing an example, I chose to use the surprisingly tricky challenge of one tiny part of the whole... which was any Internet-connected operating system's TCP/IP protocol stack. I noting how many problems had historically beset just that relatively small piece of surprisingly complex code.

So I thought "Whoops! I guess we're not quite done yet" when a new flaw was just revealed in Linux's TCP stack. There are problem (three CVE's) with a TCP feature known as "Selective ACKnowledgement" (SACK).

Back in 2011 we recorded a series of three podcasts carefully describing the low-level operation of TCP.

- Security Now! #317, recorded September 8th, 2011 was titled TCP, Part 1.
- Security Now! #323, recorded October 19th, 2011 was titled TCP Part 2: Attacking TCP.
- Security Now! #325, recorded November 2nd, 2011 was titled RTCP Part 3: Necessary Refinements.

Whereas the original ACK packet as I described it back in 2011 allowed the recipient to only specify the last correctly-received byte in the TCP stream that had been received, the fancier Selective ACK allow the recipient to specify a LIST of the byte-ranges it has received, this allowing the sender to resend only those that were lost in transit.

Selective ACKnowledgement was described in RFC #2018 dated October 1996 -- 23 years ago. So it's not exactly a new feature of TCP... yet we still don't all have it implemented correctly. Many of these features are easy to specify yet difficult to implement, since they require holding onto buffers in the stack until missing pieces of data have been received. So they are prone to manipulation. Netflix's chief of Information Security, Jonathan Looney found a subtle flaw in Linux's handling of SACK's which can be used to bring any default Linux kernel since version 2.6.29, released 10 years ago... to its knees.

Yesterday at 5pm RedHat posted a very nice and clean advisory:
https://access.redhat.com/security/vulnerabilities/tcpsack

> RedHat: A remote user can trigger this issue by setting the Maximum Segment Size(MSS) of a TCP connection to its lowest limit of 48 bytes and sending a sequence of specially crafted SACK packets. Lowest MSS leaves merely 8 bytes of data per segment, thus increasing the number of TCP segments required to send all data.

In other words... ANY Linux machine -- typically any publicly exposed Linux Server -- which has open, listening, connection-accepting ports, can now be remotely crashed by inducing a Linux Kernel Panic. Linux machines can be configured to auto-reboot upon a panic, but that's not their default. They normally just halt under the assumption that something very unexpected has just happened.

Ubuntu wrote: https://wiki.ubuntu.com/SecurityTeam/KnowledgeBase/SACKPanic

Jonathan Looney discovered several flaws in the way that the Linux kernel's TCP implementation processes Selective Acknowledgement (SACK) options and handles low Maximum Segment Size (MSS) values. A remote attacker could use these issues to perform denial of service attacks on a server. CVE-2019-11477 is the highest severity issue because a remote attacker can leverage it to immediately crash a system due to an integer overflow when processing TCP SACKs. It affects all current Ubuntu releases.

CVE-2019-11477 and CVE-2019-11478

You should update your kernel to the versions specified below in the Updates section and reboot. Alternatively, Canonical Livepatch updates will be available to mitigate these two issues without the need to reboot.

If neither of those options are possible at this time, you can mitigate the issue by temporarily disabling TCP SACK support:

```
$ sudo sysctl -w net.ipv4.tcp_sack=0
 net.ipv4.tcp_sack = 0
```

IMPORTANT: The sysctl modification shown above is not persistent across reboots

The mitigation described below for CVE-2019-11479 is also sufficient for CVE-2019-11477 and CVE-2019-11478 if disabling TCP SACK support is not viable.

CVE-2019-11479

Ubuntu kernel updates are not yet available for CVE-2019-11479. Future Ubuntu kernel updates will be available for Ubuntu 19.04, Ubuntu 18.10, Ubuntu 18.04 LTS, and Ubuntu 16.04 LTS which will provide a sysctl that allows the system administrator to define the MSS value that the system should honor when outgoing TCP segments.

In the meantime, you may use an iptables rule to define the MSS value accepted for new TCP sessions. The rule will need to be tailored to your network environment in order to ensure that you aren't blocking TCP connections containing reasonable MSS values for your environment. The addition of a simple rule that only allows MSS values greater than or equal to 500 bytes is shown here:

```
 $ sudo iptables -A INPUT -p tcp -m tcpmss --mss 1:500 -j DROP
```

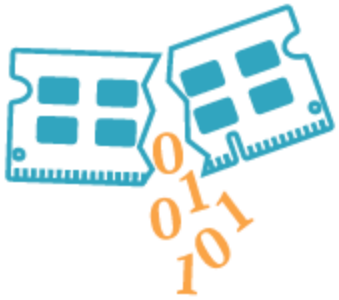
IMPORTANT: The net.ipv4.tcp_mtu_probing sysctl must be disabled (set to 0) when using the iptables rules shown above. Ensure it is disabled using the following command:

```
 $ sysctl net.ipv4.tcp_mtu_probing
net.ipv4.tcp_mtu_probing = 0
```

We had RowHammer and HeartBleed... Now we have "RAMBleed"

It was June of 2014, so five years ago, that we got the first whiff of a problem with DRAM memory not being as stable as we all hoped and assumed it was. Since then this subtle flaw has been developed into a growing family of attacks including Rowhammer, GLitch, RAMpage, Throwhammer, Nethammer and DRAMmer. This that lost we now add "RAMBleed"... and we shoud probably call the prolific teams at the University of Michigan and the Graz University of Technology the "Ram Busters."

Of course, it has a website: https://rambleed.com/ and a logo...



Official title: RAMBleed - Reading Bits in Memory Without Accessing Them

RAMBleed is a side-channel attack that enables an attacker to read out physical memory belonging to other processes. The implications of violating arbitrary privilege boundaries are numerous, and vary in severity based on the other software running on the target machine. As an example, in our paper we demonstrate an attack against OpenSSH in which we use RAMBleed to leak a 2048 bit RSA key. However, RAMBleed can be used for reading other data as well.

RAMBleed is based on a previous side channel called Rowhammer, which enables an attacker to flip bits in the memory space of other processes. We show in our paper that an attacker, by observing Rowhammer-induced bit flips in her own memory, can deduce the values in nearby DRAM rows. Thus, RAMBleed shifts Rowhammer from being a threat not only to integrity, but confidentiality as well. Furthermore, unlike Rowhammer, RAMBleed does not require persistent bit flips, and is thus effective against ECC memory commonly used by server computers.

We will present our paper titled "RAMBleed: Reading Bits in Memory Without Accessing Them" at the 41st IEEE Symposium on Security and Privacy in May, 2020.

Okay... So how is RowHammer different from RAMBleed?

[Explain about memory grid and memory management which might make other process memory adjacent.]

In RowHammer, they pounded on their OWN memory to flip bits in adjacent bits belonging to other processes. Remember the case of flipping some bits in another private key so that it was no longer the product of two primes and could be factored? So, again, RowHammer pounds on their own bits to flip someone else's.

RAMBleed, instead, arranges to bring copies of unknown data, to be exfiltrated, onto either side of the attacker's own memory. Then they pound on their own adjacent bits and observe their success rate in inducing bit flips in their own memory. It turns out that the 1 or 0 state of the bits SURROUNDING a RowHammered bit affects the likelihood of it flipping. So, in other words, they are able to successfully INFER the likely state of bits they cannot directly read by the electrostatic EFFECT the unreadable bits have upon the bits they CAN directly read.

Specifically, '1' bits tend to flip from 1 to 0 when the bits above and below them are 0, but not when the bits above and below them are 1. Similarly, '0' bits tend to flip from 0 to 1 when the bits above and below them are 1, but not when the bits above and below them are 0.

They are able to achieve a bit reading rate of 3 to 4 bits per second. So, once the stage has been set, it doesn't take long to extract a 2048-bit secret key.

And, like with RowHammer, where ECC memory would thwart the attack by flipping the flipped bit back where they belong, In this case, since they are reading their own bits, it turns out that ECC error correction slows down the next access so dramatically that the fact that one of their own bits was flipped to also be inferred even when it cannot be directly observed.

So where is the danger and what does this mean?? It's another example of the trouble we have today with sharing a single machine with another possibly-hostile party. All of last year's Spectre and Meltdown mess was about subtle changes being left behind by other processes with which we were sharing our modern processor micro-architectures. All of these DRAM failures arise from leveraging subtle edge-cases in DRAM integrity. They would normally be nothing to worry about, but in a shared environment they can be used to alter someone else's data, and, we now see, to directly read it, because we're sharing the same physical DRAM chip with someone who may have an interest in stealing our secrets. RAMBleed makes that possible.

On their summary page they asked: How can I mitigate this issue?

> "Users can mitigate their risk by upgrading their memory to DDR4 with targeted row refresh (TRR) enabled. While Rowhammer-induced bit flips have been demonstrated on TRR, it is harder to accomplish in practice. Memory manufacturers can help mitigate this issue by more rigorously testing for faulty DIMMs. Furthermore, publicly documenting vendor specific TRR implementations will facilitate a stronger development process as security researchers probe such implementations for weaknesses."

In other words, DRAM manufacturers need to stop being proprietary about their anti-RowHammering mitigations... lest they become victims of yet another round of mitigation bypass cleverness.

It’s also worth noting that IF physical RAM memory was kept encrypted down at the chip level -- which would require an extremely high-performance in-line hardware ciphering pipeline of some sort, then ALL of these attacks would be thwarted. Another solution would be to never leave encryption keys unencrypted in RAM. That’s the approach I took with my SQRL client for Windows. SQRL identities are always encrypted. The identity is loaded into RAM encrypted. Then only briefly during the moment of its use is it decrypted into RAM, used to key the domain name hash, then immediately wiped. This works for SQRL since our use of the decrypted key is transient and relatively infrequent. It would not work as easily -- or at all -- with a web server which was constantly needing to use its private key to setup new TLS connections.

So… Just as we have learned that our high-performance CPU architectures are flawed at a fundamental level by being altered by their own execution history, we have seen that DRAM is similarly flawed at a fundamental level. It is susceptible to adjacent row read and write interference. The trouble is, this really is a fundamental flaw. Everything we do now is an attempt to mitigate rather than eliminate that problem.

https://rambleed.com/docs/20190603-rambleed-web.pdf

Yet another highly redundant warning about the "BlueKeep" vulnerability.
Just to very quickly inform anyone who has missed the past five podcasts... "BlueKeep" is the zero-authentication Remote Desktop Protocol (RDP) vulnerability which affects Windows XP, Server 203, Windows 7 and Server 2008 whenever remote desktop is exposed on those machines. And, as we learned last week, by combining the vulnerability with the long standing MimiKatz exploiter it's possible to bypass the one mitigation -- the NLA: Network Level Awareness -- that might have protected some of the RDP servers.

So we've had two warnings from Microsoft following their provision of a patch during May's patch Tuesday. Then last week we got a warning from the NSA. And now we have one from the US Department of Homeland Security's CISA, the Cybersecurity and Infrastructure Security Agency, which yesterday published an alert for Windows users to patch the critical severity Remote Desktop Services (RDS) RCE security flaw known as BlueKeep.

https://www.us-cert.gov/ncas/alerts/AA19-168A

The CISA announcement indicates that they "coordinated with external stakeholders and determined that Windows 2000 is vulnerable to BlueKeep." The CISA apparently tested BlueKeep against a Windows 2000 machine and achieved remote code execution.

There's been speculation about how far back this might go. So now we know, for the first time, that it extends all the way back to and including Windows 2000... And everything in between through Windows 7 and Server 2008.

And, at this point, any machine was going to be patched has been patched. So this must simply be CYA by the DHS CISA. They are seriously expecting something bad to happen, and they don't want to be in the position of that happening when they hadn't chimed in and warned everyone to patch. You can just imagine some congressman saying, when this happens "Well... why didn't the DHS warn everyone that they needed to patch their computers to prevent this?" So now, at least, the DHS can say that they DID tell everyone... Lot of good it'll do.

I'm still unconvinced that we're going to see a worm -- although the world surely is asking for one! As I've noted before, given that these machines can be taken over instantly and that they are not difficult to find, and given that cryptocurrency miners are still the favored malware to install, and given that a RAM-based patch to shut down door behind a miner exists, a campaign to quietly install mining malware makes the most sense to me. We'll see. But so far there's been no reports of anything happening.

When Tavis tells you that you have 90 days to fix something... get on it!!
We are, of course, speaking of Google's illustrious bug-finder, Tavis Ormandy.
https://bugs.chromium.org/p/project-zero/issues/detail?id=1804

Issue 1804: cryptoapi: SymCrypt modular inverse algorithm
Reported by taviso@google.com Tue, Mar 12, 2019, 9:15 PM PDT

There's a bug in the SymCrypt multi-precision arithmetic routines that can cause an infinite loop when calculating the modular inverse on specific bit patterns with bcryptprimitives!SymCryptFdefModInvGeneric.

I've been able to construct an X.509 certificate that triggers the bug. I've found that embedding the certificate in an S/MIME message, authenticode signature, schannel connection, and so on will effectively DoS any windows server (e.g. IPsec, IIS, Exchange, etc) and (depending on the context) may require the machine to be rebooted. Obviously, lots of software that processes untrusted content (like antivirus) call these routines on untrusted data, and this will cause them to deadlock.

You can verify it like so, and notice the command never completes:

C:\> certutil.exe testcase.crt

I'm filing this as low severity, although you can take down a windows fleet pretty quickly with it.

This bug is subject to a 90 day disclosure deadline. After 90 days elapse or a patch has been made broadly available (whichever is earlier), the bug report will become visible to the public.

Six days later, on Monday, Mar 18, 2019, 11:19 AM PDT, Tavis added an update:

- "Labels: MSRC-50858"

Eight days later, on Tuesday, Mar 26, 2019, 6:56 AM PDT, Tavis added:

- "Microsoft replied that they would like to issue a bulletin for this issue, but need until June 11th. I count that as 91 days, but within the extension period so it's acceptable."

Then, last Tuesady, Jun 11, 2019, 8:16 AM PDT (6 days ago):

- "Labels: -Restrict-View-Commit Deadline-Exceeded"
- "MSRC reached out and noted that the patch won't ship today and wouldn't be ready until the July release due to issues found in testing. As today is 91 days, derestricting the issue."

Microsoft’s GITHUB: Introduction...

SymCrypt is the core cryptographic function library currently used by Windows.

The library was started in late 2006 with the first sources committed in Feb 2007. Initially the goal was limited to implement symmetric cryptographic operations, hence the name. Starting with Windows 8, it has been the primary crypto library for symmetric algorithms.

In 2015 we started the work of adding asymmetric algorithms to SymCrypt. Since the 1703 release of Windows 10, SymCrypt has been the primary crypto library for all algorithms in Windows.

Version 1703 (Fall Creators Update)

The Windows 10 Fall Creators Update (also known as version 1703 and codenamed "Redstone 2") which began its full roll out on April 11, 2017.

So... any Windows server brought online or updated in the last two year, since 1703, will be using this currently-vulnerable SymCrypt library for its asymmetric encryption (the public key crypto used by certificates).

For Tavis to consider this to be a "low severity" problem it must be that regular IIS-based websites are not vulnerable to a simple certificate-supplying attack. In TLS connections with client credential specification, servers must explicitly request the optional client certificate. I haven't tried it. But it must be that if a client were to supply an unrequested certificate it would either kill the TLS handshake or would NOT be processed by the server. If that's the case, then regular IIS websites would be safe from remote DDoS exploitation. But corporate web servers may well be requesting client certificates... and they would all appear to be vulnerable to this attack. And there are still plenty of non-Web situations where connecting-client certificates are used, as Tavis mentions in his original note. IPsec, eMail, etc.

This may not be a big deal, and it sure seems certain that Microsoft will have this fixed by next month. But it's admittedly a mess that it wasn't fixed within the ample 90-day window that Tavis provided.

GandCrab is gone, but victim files may not be!

We recently covered the history of GandCrab from its startup in January of 2018 through their announcement of planned shutdown, claiming to have "earned" (or extorted) a huge total sum of money was they had then laundered with investments in legitimate businesses.

BleepingComputer, who has been following this saga, as they do much of the ransomware world, noted that the GandCrab command and control servers were hacked and their victim's decryption keys were obtained. Consequently, anyone who was hit by v1, v4, and v5 to v5.2 (which cover the latest releases) can have their machines successfully decrypted for free.

BitDefender is offering a free "GandCrab Removal Tool":
https://labs.bitdefender.com/wp-content/uploads/downloads/gandcrab-removal-tool-v1-v4-v5/

After being installed it will need to connect to the Internet in order to check-in with BitDefender's servers and obtain the keys for that specific machine. After testing to make sure that all is well (perhaps aim it at a small sub-directory first) it can be turned loose on the entire machine for restoration.

https://www.bleepingcomputer.com/news/security/release-of-gandcrab-52-decryptor-ends-a-bad-ransomware-story/

Yubico hit a bump and is replacing FIPS keys...
https://www.yubico.com/support/security-advisories/ysa-2019-02/

Anyone using Yubico's FIPS-compliant hardware dongles with versions 4.4.2 or 4.4.4 should contact Yubico for a replacement to obtain the corrected firmware 4.4.5. In the middle of March, so just about exactly three months ago, Yubico internally discovered that some of the data left behind by the FIPS firmware power-up self-test was lingering in the device's random bits buffer... and was mistakenly being delivered as high-quality entropy. Once all of those bits were consumed the random bits buffer would be refilled with intended high-quality entropy. But this meant that the initial post-power-up state of the device was not generating the intended entropy.

https://www.yubico.com/support/security-advisories/ysa-2019-02/

Who should read this advisory? Customers, IT Managers, or FIPS Crypto Officers who use or manage YubiKey FIPS Series devices.

An issue exists in YubiKey FIPS Series devices, versions 4.4.2 and 4.4.4 (please note, there is no released firmware version 4.4.3.), where the first set of random values used by YubiKey FIPS applications after each device power-up have reduced randomness. This may impact the very first set of cryptographic operations by a YubiKey FIPS device after device power-up. This issue is specific to the YubiKey FIPS Series and is not present in any other YubiKeys, Security Key Series or Yubico products.

The issue only affects certain use cases and scenarios. YubiKey FIPS applications utilizing ECDSA are at higher risk than other use cases. See the Technical Details section below for additional information about how this issue might impact different scenarios, as well as what mitigating factors exist.

Yubico internally found this issue mid-March, 2019, followed by a full investigation of root cause, impact, and mitigations for customers. The issue has been fixed in YubiKey FIPS Series firmware version 4.4.5. Due to the firmware update, FIPS recertification was also necessary. The new firmware, version 4.4.5 achieved FIPS certification on April 30, 2019.

To safeguard the security of our customers, Yubico has been conducting an active key replacement program for affected FIPS devices (versions 4.4.2 and 4.4.4) since the issue was discovered and recertification was achieved. At the time of this advisory, we estimate that the majority of affected YubiKey FIPS Series devices have been replaced, or are in process of replacement with updated, fixed versions of the devices.

However, if you have purchased a YubiKey FIPS Series device or received one from another entity, and have not been contacted by a Yubico representative, we ask that you review this advisory to determine if you may be affected and to use the replacement portal to receive updated keys.

We are not aware of any security breaches due to this issue and are committed to always improve how we help protect our customers and continuously invest in making our products even more secure.

Affected Devices: YubiKey FIPS Series with firmware 4.4.2 and 4.4.4 – there is no released firmware version 4.4.3.

=====

Note that the issue effects ECDSA more than, for example, 2048-bit RSA because the percentage of bad entropy (80-bits) is a much larger fraction of the security guarantee in short-key elliptic curve operations than in traditional prime-factor-based RSA.

Syamon - Now with DNS Logging
https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

Last week I mentioned that Mark Russinovich was enhancing his SysInternals Sysmon logging tool to add DNS query logging. I noted that it was less than useful for the typical desktop user and suggested that our listeners might want to check out NirSoft's DNSQuerySniffer: https://www.nirsoft.net/utils/dns_query_sniffer.html

Several of our listeners wrote to note that in enterprise environments being able to pull event logs from all the worksations for centralized management was a HUGE win, and that the addition of Sysmon DNS logging to the event logs would be wonderful.

So, anyway, Mark released v10.1 of Sysmon... For all of you enterprise log monitors out there! :)

SQRL

- First release of SQRL Explainer forgot to mention "Ask"
- SQRL Tshirts on Amazon!
- Use of WINE
- Call for SQRL user videos.
  Uncompressed video / filemail or Firefox Send
  For secrurity, I'll compress using FFMPEG
  ffmpeg -i "{input.mp4}" -filter:v scale=960:-1 -b:a 64k "{output.mp4}"

SpinRite

Underestimated SR v6.1 performance: On a 7500 RPM 2TB disc, a data recovery level2 scan looks closer to only 3 hours, rather than 4.

Closing The Loop

Andy Pastuszak @amp68
TOTP: I went and enabled TOTP on every site I have an account on that supports it. And now my Google Authenticator is a complete mess. Having one long list of TOTP codes can be quite problematic, especially since I use a Firefox extension that wipes all my cookies when I close the browser. I'd love it if there was a TOTP app that doesn't sync, and allows me to organize my codes by folder or even tags, or even just had a search feature. Until SQRL gets here, we're all stuck with 2FA. Could you maybe recommend a 2FA app that has some organization behind it and has the Steve Gibson seal of approval? I can't be the only listener with this problem.

"OTP Auth" : 2-Factor Auth for Pros / Ronald Moers
https://apps.apple.com/us/app/otp-auth/id659877384
Rating: 4.8
iPhone / iPad / Apple Watch!
https://cooperrs.de/otpauth.html
https://cooperrs.de/otpauth_macos.html (there's also a macOS version)

Features:
- Ads free
- Encrypted iCloud Sync
- Siri Support
- Apple Watch support
- Notification Center widget
- Safari extension for pasting passcodes into websites
- Secure application using Face ID/Touch ID (or password)
- Create encrypted backups of all accounts
- Import/Export encrypted accounts using AirDrop, iCloud, Dropbox, Mail, ...
- Works offline

All data stored by OTP Auth is stored using strong AES-256 encryption. This applies for all data. In particular, for both locally stored data as well as data stored in the iCloud Drive (when iCloud Sync is enabled). The password for those files never leaves your device such that noone but you can read your data.

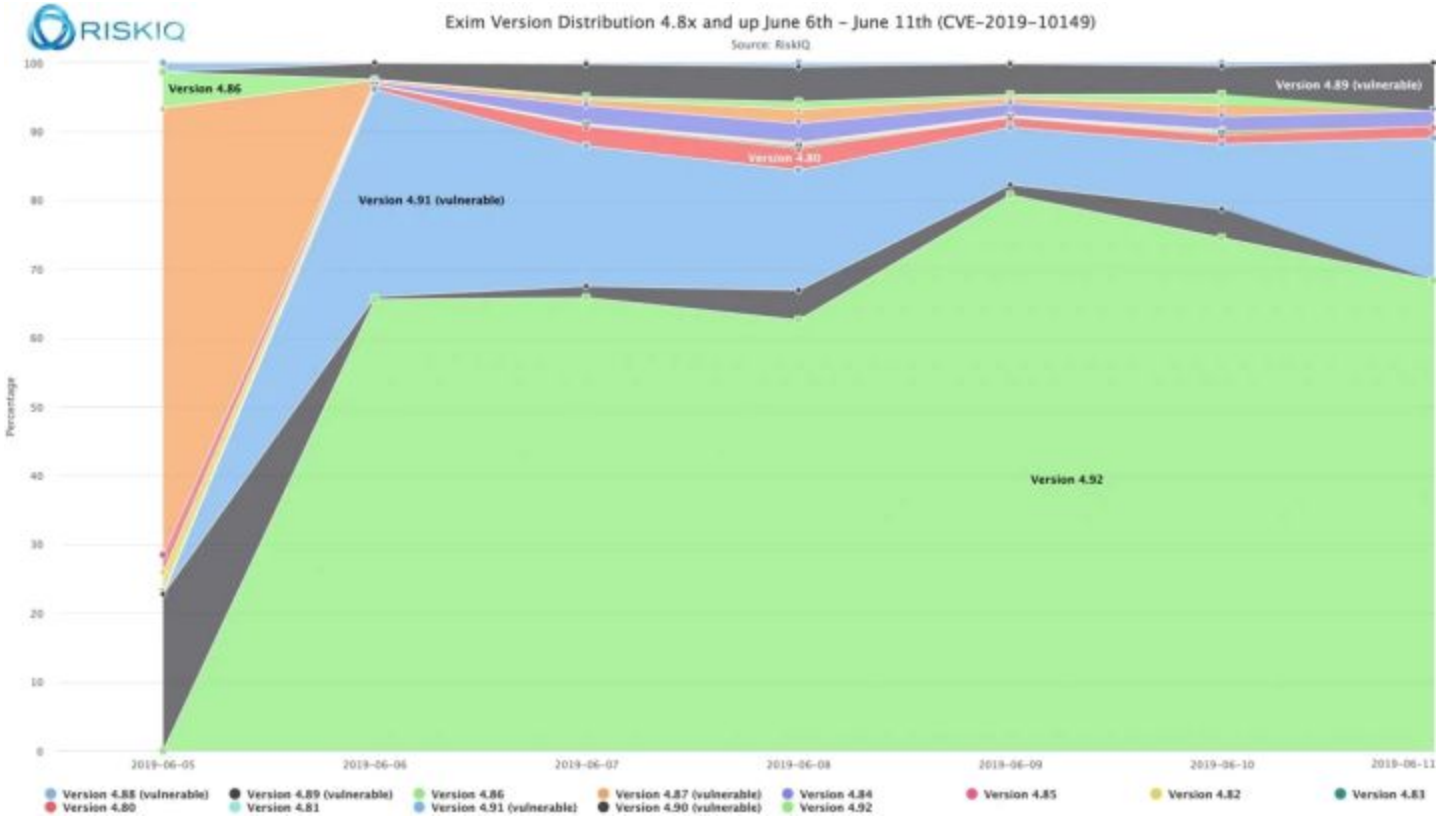
OTP Auth does not collect information about you or send them anywhere. You won't be asked to allow access to your contacts or your location. And most important: OTP Auth does not connect to the internet and will not send your accounts to someone else! (Unless you enable iCloud sync where an AES-256 encrypted copy of your accounts is stored in your iCloud Drive. The password for that copy will never leave your phone.) Follow me on Twitter: @otpauth

"my krol" / @mykrol
Hi Steve! in lot of podcast episodes i hear you talking about the special tabs in firefox, can you explain me what are you talking about?

"Tree Style Tab" - Now skinnable with CSS to create many tabs. :)

Exim Under Siege

June 5th to June 6th -- Significant and nearly immediate adoption of non-vulnerable Exim v4.92



But still there are PLENTY of vulnerable Exim servers:

Last Tuesday's podcast for June 11th was titled: "Update Exim Now!"

Two days later, Thursday, June 13th, Amit Serper of CyberReason posted: "New pervasive worm exploiting Linux Exim Server Vulnerability"

https://www.cybereason.com/blog/new-pervasive-worm-exploiting-linux-exim-server-vulnerability

Summary

There's an active, ongoing campaign exploiting a widespread vulnerability in linux email servers. This attack leverages a week-old vulnerability to gain remote command execution on the target machine, search the Internet for other machines to infect, and initiates a crypto miner.

- Currently, more than 3.5 million servers are at risk worldwide.
- The attack scours the Internet for a vulnerability discovered last week, CVE-2019-10149 using already infected servers to spread to as many as possible.
- The target of this attack, exim servers, run almost 57% of the Internet's email servers.
- The attack culminates in the downloading of a coin miner payload, which as we have seen previously with WannaMine can have a negative impact on any organization.
- These kinds of attacks have big implications for organizations. The recovery process from this type of attack is costly and time consuming.

CVE-2019-10149, which was first discovered on June 5, is now being used as the vulnerability for a widespread campaign to attack exim servers and propagate across the Internet.

We are aware of an initial wave of attacks as described by Freddie Leeman on June 9, 2019. The first hacker group began pushing exploits from a C2 server located on the clear web.

A second round of attacks by a different attacker are being analyzed by the Nocturnus team.

The campaign uses a private authentication key that is installed on the target machine for root authentication.

Once remote command execution is established, it deploys a port scanner to search for additional vulnerable servers to infect. It subsequently removes any existing coin miners on the target, along with any defenses against coinminers, before installing its own.

Note: This is a very long script that downloads additional scripts and changes or adds many configurations on Linux servers. This blog has the highlights of what the script is doing to provide a fast reference guide to this attack. Some of the things that the script is doing are not documented in this blog post. The hash of the script is available at the end of this article. It has also been uploaded to VirusTotal.

Campaign Highlights

This is a highly pervasive campaign that installs cronjobs for persistence and downloads several payloads for different stages of the attack.

In one of those stages, one of the payloads is a portscanner written in python. It looks for additional vulnerable servers on the Internet, connects to them, and infects them with the initial script.

In the attack, the attackers add an RSA authentication key to the SSH server which allows them to connect to the server as root and own it completely.

If you are running an updated version of the Exim mail server or you think that your server is compromised, please look for the following entry in our SSH configurations in /root/.ssh and in every .ssh directory on your system.

"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC1Sdr0tIIL8yPhKTLzVMnRKj1zzGqtR4tKpM2bfBEx+AHyv
BL8jDZDJ6fuVwEB+aZ8bl/pA5qhFWRRWhONLnLN9RWFx/880msXITwOXjCT3Qa6VpAFPPMazJpbp
pIg+LTkbOEjdDHvdZ8RhEt7tTXc2DoTDcs73EeepZbJmDFP8TCY7hwgLi0XcG8YHkDFoKFUhvSHPkz
AsQd9hyOWaI1taLX2VZHAk8rOaYqaRG3URWH3hZvk8Hcgggm2q/IQQa9VLlX4cSM4SifM/ZNbLYAJ
hH1x3ZgscliZVmjB55wZWRL5oOZztOKJT2oczUuhDHM1qoUJjnxopqtZ5DrA76WH user@localhost"

Entry in our SSH configurations in /root/.ssh

In the final stage of the infection, the script downloads what appears to be a windows icon file (.ico) that has a specific icon. Its headers were modified to appear as an ico file.

However, the icon file is actually a password protected zip archive with password “no-password”. A 64-bit statically linked, stripped, and UPX-packed ELF file is extracted from the archive. When unpacked,there is another ELF executable that is a coin miner.

All in all, there were four scripts downloaded. We are currently working on identifying more information about the campaign.

Conclusions So Far

At this point we are still conducting research to dig up more details on the attack, the breadth of the campaign, the payloads being used, etc. Since this campaign has such a broad scope, we felt it would be wise to share as soon as we became aware. This document will continue to update as we dig up more information.

It is clear that the attackers went to great lengths to try to hide the intentions of their newly-created worm. They used hidden services on the TOR network to host their payloads and created deceiving windows icon files in an attempt to throw off researchers and even system administrators who are looking at their logs.

The prevalence of vulnerable exim servers (3,683,029 across the globe according to Shodan) allows attackers to compromise many servers in a relatively short period of time, as well as generate a nice stream of cryptocurrency revenue.

=====

The following day, last Friday, June 14th, Microsoft's Technet blog posting:
"Prevent the impact of a Linux worm by updating Exim"
https://blogs.technet.microsoft.com/msrc/2019/06/14/prevent-the-impact-of-a-linux-worm-by-updating-exim-cve-2019-10149/

> This week, MSRC confirmed the presence of an active Linux worm leveraging a critical Remote Code Execution (RCE) vulnerability, CVE-2019-10149, in Linux Exim email servers running Exim version 4.87 to 4.91. Azure customers running VMs with Exim 4.92 are not affected by this vulnerability.
>
> Azure has controls in place to help limit the spread of this worm from work we've already done to combat SPAM, but customers using the vulnerable software would still be susceptible to infection.
>
> Customers using Azure virtual machines (VMs) are responsible for updating the operating systems running on their VMs. As this vulnerability is being actively exploited by worm activity, MSRC urges customers to observe Azure security best practices and patterns and to patch or restrict network access to VMs running the affected versions of Exim.
>
> There is a partial mitigation for affected systems that can filter or block network traffic via Network Security Groups (NSGs). The affected systems can mitigate Internet-based 'wormable' malware or advanced malware threats that could exploit the vulnerability. However, affected systems are still vulnerable to Remote Code Execution (RCE) exploitation if the attacker's IP Address is permitted through Network Security Groups.
>
> It is for these reasons that we strongly advise that all affected systems – irrespective of whether NSGs are filtering traffic or not – should be updated as soon as possible.

And the next day, in article posted last Saturday, the Microsoft Security Response Center (MSRC) confirmed that they have detected this worm targeting Azure customers.

=====

So why a worm for this and not for BlueKeep? It's the week long delay to obtain access. That makes all the difference. If you need to setup a connection and camp out, sending a byte every four minutes, you don't want to do that from any low number of machines. This Exim vulnerability is PERFECT for a worm's use.

BlueKeep doesn't need a worm... it's too easy. (Though, yeah, since the whole industry has been expecting one, it might be a self-fulfilling prophecy.)