Security Now! #845 - 11-16-21

Blacksmith

This week on Security Now!

This week we look at a critical 9.8-rated vulnerability affecting Palo Alto Network's widely deployed VPN/Firewall appliance, and at a welcome new micropatch from the 0patch guys, the nature of which leads me into a bit of philosophical musing about the Zen of coding. We're then rocketed back to reality by a review of last week's Patch Tuesday, looking at what it broke and happily what more it fixed, including hints that Christmas might finally be coming to printing by December. We have some more encouraging ransomware vs the law news, and we examine the question of how to make big money defrauding online advertisers. I'll then share some fun and interesting closing the loop feedback from our listeners, update on my SpinRite work, and then we're going to take a look at "Blacksmith" — the evolution of Rowhammer attacks on DRAM.

Windows 11 taking too long to download over your slow connection?

0-Day News

~10,000 VPN/Firewall appliances from Palo Alto Networks vulnerable.

Once again, the tech press is labeling this a "0-day." Nothing about is a 0-day, but it definitely is a very serious vulnerability with a critical vulnerability severity score of 9.8. The flaw affects Palo Alto Network's "GlobalProtect" firewall, allowing for unauthenticated remote code execution on multiple versions of PAN-OS v8.1, prior to 8.1.17, and on both physical and virtual firewalls.

The discovery was made by a group we've never referred to before: Randori. Their disclosure, which sadly labeled this as a 0-day, too (https://www.randori.com/blog/cve-2021-3064/) said:

The Randori Attack Team developed a reliable working exploit and leveraged the capability as part of Randori's continuous and automated red team platform. Our team was able to gain a shell on the affected target, access sensitive configuration data, extract credentials, and more. Once an attacker has control over the firewall, they will have visibility into the internal network and can proceed to move laterally.

In an effort to avoid enabling misuse, technical details related to CVE-2021-3064 will be withheld from public dissemination for a period of 30 days from the date of this publication. More information will be released at that time.

The following are the key takeaways from the Randori Attack Team's discovery and research surrounding this flaw:

- *The vulnerability chain consists of a method for bypassing validations made by an external web server (HTTP smuggling) and a stack-based buffer overflow.*
- *It affects Palo Alto firewalls running the 8.1 series of PAN-OS with GlobalProtect enabled (specifically versions < 8.1.17).*
- *Exploitation of the vulnerability chain has been proven and allows for remote code execution on both physical and virtual firewall products.*
- *Publicly available exploit code does not exist at this time.*
- *Patches are available from the vendor.*
- *Palo Alto Networks Threat Prevention Signatures are also available (IDs 91820 and 91855) to block exploitation of the issue.*
- *Public exploit code is likely to surface as:*
- *VPN devices are attractive targets for malicious actors, and*
- *Exploitation of PA-VM virtual devices in particular is made easier due to their lack of Address Space Layout Randomization (ASLR).*

They referred to "HTTP Smuggling" which I'll be explaining in detail as next week's topic.

The good news is, the Randori guys are acting responsibly and are allowing 30 days before making any further disclosures. But anyone who might be affected by this will definitely want to jump on Palo Alto's update since Shodan has indexed all of the affected VPNs and the bad guys will be all over this as a means to get into high value users of Palo Alto Networks VPN gateways.

The 0-Patch Guys Produce a Micropatch

Recall that we recently discussed a worrisome local privilege escalation vulnerability in the Windows User Profile Service which affected — or, I should say, affects — all versions of Windows. This was the one that Abdelhamid Naceri discovered and privately disclosed to Microsoft many months ago. They then listed this as patched and fixed in August. But after reverse engineering the change that Microsoft made, Naceri discovered that the only thing "fixed" was that the simple proof-of-concept demo he had provided to demonstrate the more severe underlying problem no longer worked. The underlying problem remained. So it wasn't fixed in August when IT was supposed to be, nor in September, October or last Tuesday, in November.

On the one hand, if it's exploited it allows an attacker to elevate themselves to full SYSTEM root privilege. But on the other hand, any such attacker would need to have knowledge of some second credential for the same machine. But Neceri reminds us that any domain credential would be sufficient. So that somewhat lowers the bar for an attacker.

I won't comment on the fact that when BleepingComputer asked Microsoft about this still-unresolved problem, they were told that they are aware of the issue and "will take appropriate action to keep customers protected." But what I will comment on is that last Thursday, after seeing that Microsoft still hadn't fixed this for the 4th month in a row, the guys over at 0patch created one of their very cool little "micropatches" that, believe it or not, actually works and solves the problem. It can be applied to any Windows system needing interim protection. It's available in both 32 and 64 bit versions, and can be applied on any Windows 10 system with at least the October or November updates.

I haven't taken any time to research what's going on in detail, but the founder of 0patch did say that while this issue could, in theory, also impact earlier Windows versions, "the vulnerable code is different there, making the window for winning the race condition extremely narrow and probably unexploitable." That's interesting... a race condition.

We've never discussed "race conditions" on this podcast. But it's a common problem anytime multiple things might be happening at once, where the precise sequence of them happening is important. A few weeks ago I mentioned the counter/timer in the original PC which used what was known as a ripple counter. A ripple counter is a series of individual toggling flip-flops connected in a chain. When a single toggling flip-flop's input goes to logic '1' nothing happens. But when that input drops back to logic '0', the flip-flop flips to its other on or off state. If the output of that flip-flop is connected to the input of the next flip-flop, you start to obtain a binary counter where the number of bits in the counter is the number of flip-flops in the chain.

It's fun to draw it on paper. But the problem is, flip-flops in the real world do not toggle instantly. They have what's known as a "propagation delay" which means that their output changes just a little bit **after** their input goes to '0'. The flip-flop's internal logic takes a bit of time to sort things out. This means that when the input at the front of this ripple counter goes from '1' to '0', the changes in the count might "ripple down" through the length of the counter.

Now let's say that you want to take a snapshot of the counter's current count, but you have no idea when the counter might be counting. This is where we get into a possible race condition.

If we happen to take a snapshot of the binary ripple counter WHILE the new count is rippling down the length of the counter, from one stage to the next, we will obtain an erroneous reading due to a race condition which we've created between the count event and our sampling.

And race conditions aren't just electronic; they can occur socially, too. They're a favorite vehicle of screenwriters who have one person racing to answer the phone before someone else gets the chance to avoid some embarrassing outcome. That's a race condition.

In electronics and in software, these race conditions can be quite tricky to spot, and they've been known to introduce some of the hardest-to-find bugs there are. The reason may be, that they may not occur often or predictably, only maddeningly. The state of that ripple counter might be properly sampled thousands of times before the sampling just happens to catch the counter in the process of rippling. The other maddening thing is that the act of observing the system might change it. Adding the tiny capacitance of a high impedance oscilloscope probe when checking a high-frequency signal can change its timing just enough to prevent the problem from occurring **while** you're looking for it. And the same goes for debugging code. If you step through the trouble area of some time-sensitive code, everything works. In fact, the problem only occurs on odd Tuesday's when the moon is full and the mailman is late. Otherwise... everything's great.

So our lesson here is the need for true vigilance. In any asynchronous multi-threaded application where more than one thread might **share** access to variables or objects, consider every implication of that very carefully, always keep the possibility of these sorts of collisions front of mind, and work very hard to never create such problems in the first place... because it's FAR better to prevent them than to go back and try to find those gremlins later.

This brings me to "The Zen of Code"

All new coders begin their craft without any appreciation for the subtleties and nuances of the way complex coding problems are **best** solved. The word "best" is the critical key here. For newbies, it might at first seem that any solution to a problem is as good as any other. After all, they're all solutions aren't they? But with time, patience and lots of practice at coding, a coder gradually matures to learn that there are superior, more elegant and more robust ways to think about, to frame and then to solve a problem that may have many solutions. Most coding problems do. Yet it's often the case that there is one best way. And I think that the great joy of coding comes from the satisfaction of knowing that you have found the best solution.

I'll often code something, but then I'll wonder if I can't find a better solution. What I think happens is that the act of solving the problem the first time teaches me something about the problem that I hadn't initially appreciated. For example, I might have needed to add an awkward bit of code at the end to handle an edge case that I hadn't seen coming. But if I'd appreciated it at the start, as I do now at the end, might I have come up with an overall superior solution that inherently incorporated and didn't need that special case handling because it solved the problem the right way?

The point is, I may have just learned something about the problem that might now suggest an entirely different and superior approach. So I'll leave what I did in place, so that I can go back to it if my intuition is wrong and I don't find a better way. I'll open up a blank space in my editor, and immediately tackle the same problem again, but this time starting with a deeper understanding of the problem I'm trying to solve.

It's true, that it will make no difference in **what** that chunk of code does. But it might make a difference in the **way** it does it. One solution may be esthetically cleaner and superior to the other. And as utterly abstract as code may be, within this little microcosm, it really is possible to create little works of art. Little constructed gems.

I know that not everyone obtains fulfillment from building things. And not everyone is as curious as I am. And not everyone may have the luxury of taking the time to find that one best solution. If you are those things, then what I've just articulated makes all the sense to you in the world. And if you aren't those things, maybe now you can understand a little bit better those of us who are **so weird** in this particular way.

Okay, now back down to Earth…

Windows News

November's Patch Tuesday

We already know what Patch Tuesday didn't patch. So did it patch?

Last Tuesday, Windows users received fixes for a total of 55 flaws in Windows, two of which were 0-day vulnerabilities in Exchange and Excel, both under active exploitation in the wild and, by coincidence, the 0-day in Exchange had previously surfaced during the recent Tianfu hacking contest. Four other vulnerabilities also qualify as 0-days because, while they are now under active exploitation, they were first learned of through public disclosure without any available patches ready or even started.

Of the 55 total, 6 were classified as CRITICAL and the other 49 as Important. The exploitation breakdown was:

- 20 Elevation of Privilege vulnerabilities
- 15 Remote Code Execution vulnerabilities
- 10 Information Disclosure vulnerabilities
- 4 Spoofing vulnerabilities
- 3 Denial of Service vulnerabilities
- 2 Security Feature Bypass vulnerabilities

The two actively exploited 0-days are:

- CVE-2021-42292 - Microsoft Excel Security Feature Bypass Vulnerability
- CVE-2021-42321 - Microsoft Exchange Server Remote Code Execution Vulnerability

The Excel flaw was discovered by the Microsoft Threat Intelligence Center and has been actively used in malicious attacks. The four publicly disclosed vulnerabilities that have not been seen being exploited in the wild are:

- CVE-2021-38631 - Windows RDP Information Disclosure Vulnerability
- CVE-2021-41371 - Windows RDP Information Disclosure Vulnerability
- CVE-2021-43208 - 3D Viewer Remote Code Execution Vulnerability
- CVE-2021-43209 - 3D Viewer Remote Code Execution Vulnerability

So that's the good news. How'd this turn out so far this month?

November broke something, but don't ask me what...
After installing last Tuesday's updates, Microsoft said that users might experience authentication problems on Domain Controllers running Windows Server. They explained, and I'm just going too quote them because I only have a vague idea what they're talking about:

"Kerberos authentication will fail on Kerberos delegation scenarios that rely on the front-end service to retrieve a Kerberos ticket on behalf of a user to access a backend service. Important Kerberos delegation scenarios where a Kerberos client provides the front-end service with an evidence ticket are not impacted. Pure Azure Active Directory environments are not impacted by this issue."

And as if to clarify, on the Windows Health Dashboard Microsoft wrote: "After installing the November security updates, you might have authentication failures on servers relating to Kerberos Tickets acquired via S4u2self. The authentication failures are a result of Kerberos Tickets acquired via S4u2self and used as evidence tickets for protocol transition to delegate to backend services which fail signature validation."

The good news is, whatever it was that broke, they fixed yesterday with the release of an out-of-cycle update. So, if anyone had updated last Tuesday and their use of Kerberos for authentication to domain controllers died, it's now fixed as of yesterday.

Windows 11 received KB5007215
KB5007215 is a cumulative update to fix security vulnerabilities and bugs newly introduced, or at least occurring, in previous versions. This update is mandatory because it contains security updates, performance improvements, and bug fixes for Windows 11 21H2. After installing the update, Windows 11 will have its build number changed to 22000.318.

There was a newly introduced problem with GDI+ which prevented the proper display of UI elements. That's fixed and GDI+ applications should be working again.

The update also fixes that mysterious built-in application signature certificate chain verification bug which was preventing a subset of apps, such as the Snipping Tool, from working in some instances for some users on some planets. Now, everything is supposed to be working correctly once again.

And, finally, Microsoft also stated that they made changes to the servicing stack to resolve bugs or issues preventing Windows updates from properly being installed. So now, presumably, Windows Updates will no longer be prevented from being properly installed, to the tremendous relief of all concerned.

December promises to be Christmas for Printing and more!
It appears that the raft of continuous, embarrassing and unrelenting problems Windows has been experiencing has caught Microsoft's attention. Yay! Late last week they released a new

build for Windows 11 Insiders in the Beta and Release Preview Channels and it promises to have fixed a long list of issues impacting Windows 11.

Among the significant bugs addressed in this Build 22000.346 (KB5007262), are fixes for known issues preventing USB Print devices with support for Internet Printing Protocol (IPP) Over USB from completing the installation and leading to some USB Print installers reporting that they don't printer after they were plugged in.

Another printing issue fixed in this preview build was behind those three printing errors we described weeks ago:

- 0x000006e4 (RPC_S_CANNOT_SUPPORT)
- 0x0000007c (ERROR_INVALID_LEVEL)
- 0x00000709 (ERROR_INVALID_PRINTER_NAME)

Which users were reporting when connecting to remote printers shared on Windows print servers. So Microsoft promises that those are finally fixed. And the best news is that these fixes, now in Beta and Release Preview, are expected to roll out to all impacted Windows 10 and Windows 11 customers during the December 2021 Patch Tuesday.

Ransomware News

US detains crypto-exchange exec for helping Ryuk ransomware gang launder profits
I wanted to keep everyone in the loop about news of the continuing US led pursuit of those who have been profiting from ransomware attacks.

In this case the suspect, named Denis Dubnikov, was arrested two weeks ago, on November 2, while attempting to vacation in Mexico. He was denied entry into the country, pending an arrest warrant, and Mexican officials forwarded him to Amsterdam, where he was officially detained by Dutch police at the request of the FBI.

Although arrests of crime suspects usually remain unreported until official charges are filed, in this case the news of the arrest leaked via Dubnikov himself, who revealed his fate in an Instagram story he posted on his now-deleted account while he was in custody in Mexico, according to screenshots posted on Russian Telegram channels. But the details surrounding Dubnikov’s arrest had remained secret until just recently with neither Dutch police nor US officials responding to requests for comment. So there's still much that's not public. The Wall Street Journal obtained and reported upon an extradition request which revealed that Dubnikov stands accused of money laundering. According to court documents, around $400,000 in cryptocurrency assets tied to a Ryuk ransom payment passed through one of Dubnikov’s accounts in 2018. And it's unclear whether the $400K passed through Dubnikov’s personal account or through two cryptocurrency platforms — Crypto Coyote and EggChange — which Dubnikov, a Moscow businessman, founded previously. A Bloomberg article published on November 3, the day after Dubnikov’s arrest, named EggChange as one of the multiple shady cryptocurrency exchanges that are headquartered in a Moscow office building that has been tied to cybercrime money laundering.

The best news is that the arrest has triggered outrage within the Russian cryptocurrency community, with several prominent figures demanding an official response and condemnation of Dubnikov's arrest from the Russian government. I think it's great that something that happened three years ago is being loudly dredged up and acted upon today, since you can imagine all of the other crypto-weenies now starting to worry about what evidence from their own pasts can be reverse-engineered from the public cryptocurrency blockchains to produce irrefutable evidence of their past misdeeds. As we noted last week, the money obtained from ransomware is seeming less free everyday.

Security News

How do you defraud web-based advertisers?

- You're a 41-year-old Russian national by the name of Aleksandr Zhukov.
- It occurs to you that advertisers pay websites to have their ads displayed to people who are presumably influenced to some degree, and who may even occasionally click on an ad to find out more.
- So, you establish thousands of entirely fake public websites hosted by legitimate cloud service providers and contract with those advertisers to have their ads shown on those site's pages. And because video ads pay more, you focus upon having those hosted.
- But no one's going to those bogus sites because they have crappy synthetic content.
- So you create a wide-ranging network of Bots called "Methbot" to visit those bogus pages from IP addresses scattered across the world as if they were actual people. You design the bots to poke around the various website pages, to "view" those ads and videos, and to occasionally click on one to learn more.
- ... And the money starts rolling in. How much money? Is everyone sitting down? Between 3 and 5 million dollars per day!

But it's fraud, of course, since no one whose behavior might be influenced to purchase anything ever sees these ads for which advertisers are paying and money is being received.

But something sets off some scam alarms which cause the FBI, Google, and 20 other tech and security firms to probe and dismantle Zhukov's operation. They soon discover a giant botnet running across infected devices and legitimate data centers. And, oddly, that Botnet is visiting these bogus websites and clicking links and ads.

Shortly afterward, back in November 2018 while traveling to Bulgaria, Zhukov is arrested on an FBI-issued arrest warrant and is formally charged a week later.

Tracked under names such as 3ve, Methbot, Boaxxe, and Miuref, Zhukov is believed to have run one of the largest ad fraud botnets ever created, generating as I said at one point in 2016 between $3 million and $5 million in revenue per day. Zhukov hired programmers to help him build and manage this operation, which he disguised as a legitimate advertising network named "Media Methane". And according to prosecutors, Zhukov referred to himself as the "king of fraud" and his employees as "my developers."

But of course, instead of running an actual advertising company, prosecutors said that Zhukov created thousands of fake websites where he loaded ads from legitimate publishers. These

websites ran on more than 2,000 servers rented across data centers and had no real visitors. Instead, Zhukov's team used automated tools to simulate real visitors. Prosecutors said Zhukov often leased IP addresses for this activity, and in other cases, he also hijacked legitimate IP addresses from their legitimate owners.

But all that came to a just end last week when a US judge sentenced Zhukov to 10 years in prison for running Methbot between 2014 and 2018.

Closing The Loop

"TIP"s SCSI Commands

Hello,

This may seem like a strange request in 2021, but would you be able to offer some insight on the exact SCSI commands used by your "click-of-death" tool Trouble in Paradise?

It might surprise you that anyone is still using Zip drives in 2021, but SCSI Iomega Zip drives are still used by people who collect vintage Apple Macintosh computers.

Obviously these computers can't run TIP, but I recently came across a PC SCSI adaptor card at a surplus sale and decided to see if I could get TIP to run on a Windows PC. It turned out to be an ordeal due to the fact Windows 98 simply doesn't run very well on newer hardware. But eventually I got it all to work and I am very impressed with the tool. I was quickly able to troubleshoot my collection of Zip media and drives.

Anyhow, I would very much like to see if I can recreate a similar tool that runs natively on my vintage Macintosh computer so I can ditch the jerry-rigged Windows 98 PC and maybe help other Mac users which also are using Zip drives.

Could you provide any insight on the SCSI commands used?

-- Marcio

Hardware IRQs: Robert Sciabbarrasi / @Prof_RAS6
Thanks for the dive into IRQs! I assigned that segment of the show as a reading/listening/watching assignment for my Computer Architecture class. Keep the deep dives coming!

Bell Labs Patent Attorney

From: Stephen Phillips <.....@bellsouth.net>
Subject: UNIX Programmer's Manual SN #844

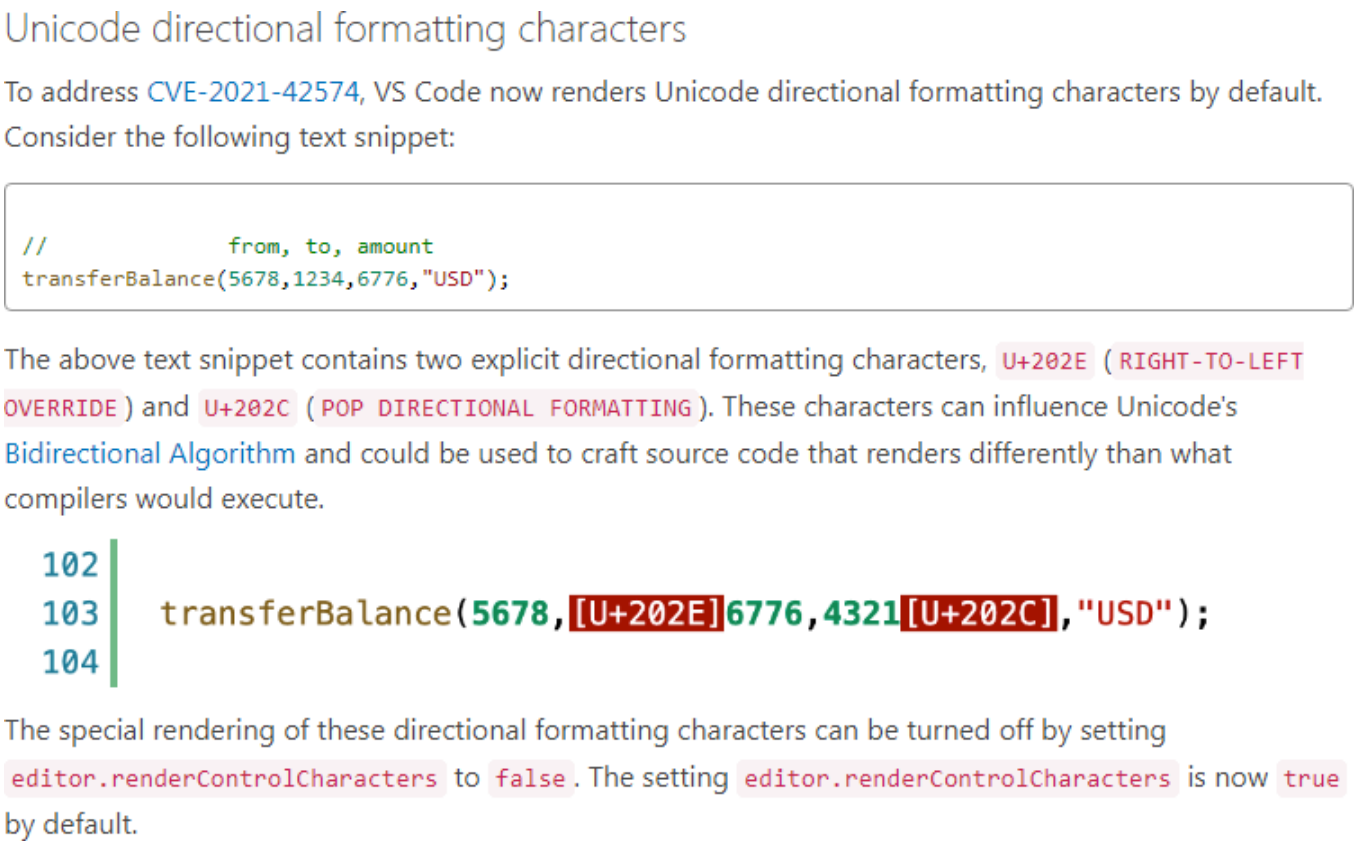
The cover of the UNIX manual brought back many memories for me. I was the Bell Labs Patent Attorney who consulted for the Mathematics Research Department where Ken and Dennis worked. My bookshelf was lined with all the manuals as they issued. I spent many hours with the manuals and with Ken and Dennis trying to find out what made UNIX tick. They were very modest about their work and thought they hadn't really done anything new. For the record, I filed two patents, now long expired. One for the Set User ID Bit which allowed a non-administrator process to have administrative rights for the duration of its existence. The second was for the block-oriented UNIX file system structure.

I think our Patent Department was the first production user of UNIX. We were looking for a word processing system and they set us up with a PDP 11 doing text editing with ed. Later we built a home-grown filing system for our patent applications and called it the File Mechanization project. Early days. We had fun after UNIX became famous, when we were called into AT&T corporate headquarters to explain to the executive suite what in the world Bell Labs was doing creating software when telephones were our real business.

Stephen J. Phillips

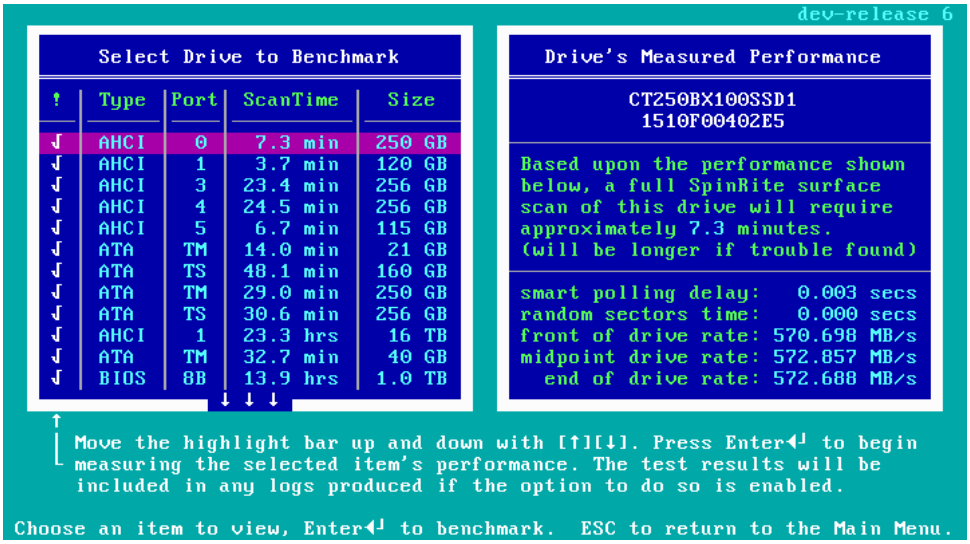
Ben Clapp @BenClapp
Hey Steve! Love the show. FYI in case no other listeners have shared, Visual Studio Code is now rendering the Unicode directional formatting characters by default. I hope more IDEs do this!

https://code.visualstudio.com/updates/v1_62#_unicode-directional-formatting-characters



SpinRite

I'm nearly finished with the 6th development release. The screen snapshot below shows SpinRite's updated drive selection pane on the left which now displays the estimated time to scan each of the drives upon which SpinRite's benchmark has been run:



The first line item shows that the performance of a 250GB Crucial SATA 3 SSD under SpinRite is 573 megabytes/second. As we know, SATA 3 has a line rate of 6 gigabits/second. But SATA is a serial self-clocking protocol which requires some encoding of the user's raw data. So, 8 bits of data is encoded into 10 bit symbols for transit over the wire. Therefore, the 6 gigabits/second on the wire translates to exactly 600 megabytes/second of data. But a SATA serial link doesn't only carry the user's data, since all of the link's non-data control-signalling also shares the same wire. SpinRite's new hardware AHCI drivers are achieving 573 megabytes/second, which is 95.5% of 600 megabytes/second. So we're only losing 4.5% to control signalling overhead and there's no way to squeeze that down any further. Given that most SATA 3 benchmarks show between 550 to 560 megabytes/second, I'm very pleased that SpinRite is achieving a sustained rate of 573 megabytes/second.

Since I think anyone who's interested in SpinRite might be curious to know more now, I created a 4-page PDF which shows a bunch of SpinRite's new screens with some commentary about the various drives and controllers shown. It's our shortcut of the week for this episode #845.

https://grc.sc/845

Blacksmith

The Return of RowHammer: Probably the biggest and most significant security news of the week appeared yesterday with a report published by the COMSEC Computer Security Group. We've spoken of the work of these guys before, but only identifying them as being a team at ETH Zürich. COMSEC is the computer security group of the Department of Information Technology and Electrical Engineering (D-ITET) at ETH Zürich. Their primary research goal is the construction of reliable and secure computing systems. So their research often touches on all layers of the computing stack, from software, all the way down to hardware. They explain that they use novel analysis techniques to better understand the attack surface of modern systems and when appropriate, build systems that can withstand different classes of attacks.

Their just published work is their design of "Blacksmith" which represents the maturation of Rowhammer attacks.

To briefly remind everyone, "Rowhammer" refers to a broad class of so-called bit-flipping attacks upon the unfortunately fundamentally flaky nature of today's dynamic RAM memories. I'll crib from Wikipedia since it's perfectly summarized there:

> *Rowhammer is a security exploit that takes advantage of an unintended and undesirable side effect in dynamic random-access memory (DRAM) in which memory cells interact electrically between themselves by leaking their charges, possibly changing the contents of nearby memory rows that were not addressed in the original memory access. This circumvention of the isolation between DRAM memory cells results from the high cell density in modern DRAM, and can be triggered by specially crafted memory access patterns that rapidly activate the same memory rows numerous times.*

We've been tracking this since it's first appearance in 2014, and it's been a constant background issue for the industry which never quite goes away.

The same research group developed their "SMASH" attack, a JavaScript-based attack that gives the attacker an arbitrary read and write primitive in the browser. Then there was "Drammer" Deterministic Rowhammer Attack on Mobile Platforms. And we have "Flip Feng Shui", Hammering a Needle in the Software Stack. And then there was over-the-network "Throwhammer". And now we have Blacksmith.

The reason we cannot seem to shake this is that the problem the first Rowhammer research revealed is not, as we say, a bug of DRAM... it's an unwanted feature. It cannot truly be fixed, because it has always been an essential truth incorporated into the way DRAM operates.

Historically, there have been other "noisy" and usage-pattern sensitive mass storage systems. The core memory technology used in the early mainframe and minicomputers of the 1960's and 70's were similarly sensitive to their access patterns with adjacent bits. Memory diagnostics were run with things like "checkerboard" patterns in an attempt to store and retrieve "worse-case" data patterns. If your core memory was acting up you wanted to determine that while running a diagnostic and not while processing real world data.

And even today, the venerable "Memtest86+" at memtest.org remains a viable and worthwhile test of a system's memory. I've had occasion to run it through the years when something seem flaky and I want to rule out a problem with DRAM.

In the case of DRAM's inherent vulnerability to Rowhammer attacks, the best we can do is attempt to minimize and mitigate the pervasive threat. We've heard, with a great deal of hope and expectation that newer generations of DRAM, like DDR4, were going to resolve this problem by incorporating specific anti-Rowhammer mitigations. How'd that work out? The short and depressing answer is that the DDR4 memory protections have been broken wide open by Blacksmith.

The COMSEC guys explain:

We demonstrate that it is possible to trigger Rowhammer bit flips on all DRAM devices today with little effort, despite deployed mitigations on commodity off-the-shelf systems. This result has a significant impact on the system's security as DRAM devices in the wild cannot easily be fixed, and previous work showed real-world Rowhammer attacks are practical, for example, in the browser using JavaScript, on smartphones, across VMs in the cloud, and even over the network.

Rowhammer is a vulnerability caused by leaking charges in DRAM cells that enables attackers to induce bit flips in DRAM memory. To stop Rowhammer, DRAM implements a mitigation known as Target Row Refresh (TRR). Our previous work showed that the new n-sided patterns can still trigger bit flips on 31% of today's PC-DDR4 devices. We propose a new highly effective approach for crafting non-uniform and frequency-based Rowhammer access patterns that can bypass TRR from standard PCs. We implement these patterns in our Rowhammer fuzzer named Blacksmith and show that it can bypass TRR on 100% of the PC-DDR4 DRAM devices in our test pool. Further, our work provides new insights on the deployed mitigations.

How did they do it?

They conducted a series of experiments beginning with the observation that all of the previously used Rowhammer attacks used uniform patterns such as single-sided, double-sided, and n-sided. The patterns were uniform in both pattern and in number. So this made them curious about DRAM's sensitivity to non-uniform attack patterns.

Since the search space of non-uniform patterns is huge, they conducted a series of experiments to determine the structure of patterns that effectively bypass the Target Row Refresh mitigations. And they discovered that the order, regularity, and intensity of accessing aggressor rows in non-uniform patterns form an essential component of successful attacks. They noted that their observations matched well known parameters of the frequency domain, namely frequency, phase, and amplitude. So they used those frequency domain parameters to design frequency-based Rowhammer patterns that can effectively explore the space of non-uniform patterns. They implemented these patterns in their black-box fuzzer which they named Blacksmith. Blacksmith is able to determine suitable parameter values for any specific targeted device. So, essentially, it uses fuzz-based attacks to learn about a specific DRAM.

So how powerful and practical are the resulting attacks? They wrote:

For our evaluation, we considered a test pool of 40 DDR4 devices covering the three major manufacturers (Samsung, Micron, SK Hynix), including 4 devices that did not report their manufacturer. We let our Blacksmith fuzzer run for 12 hours to assess its capability to find effective patterns. Thereafter, we swept the best pattern (based on the number of total bit flips triggered) over a contiguous memory area of 256 MB and report the number of bit flips. The results were that our Blacksmith fuzzer is able to trigger bit flips on all 40 DRAM devices with a large number of bit flips, especially on devices of manufacturers A and D — whom they do not identify.

We also evaluated the exploitability of these bit flips based on three attacks from previous work: an attack targeting the page frame number of a page table entry (PTE) to pivot it to an attacker-controlled page table page, an attack on the RSA-2048 public key that allows recovering the associated private key used to authenticate to an SSH host, and an attack on the password verification logic of the sudoers.so library that enables gaining root privileges. All of the attacks were successful.

Our work confirms that the DRAM vendors' claims about Rowhammer protections are false and lure you into a false sense of security. All currently deployed mitigations are insufficient to fully protect against Rowhammer. Our novel patterns show that attackers can more easily exploit systems than was previously assumed.

This work has been assigned a CVE number — CVE-2021-42114 — and their paper, titled "BLACKSMITH: Scalable Rowhammering in the Frequency Domain" will be presented during the 43rd IEEE Symposium on Security and Privacy. I have a link to their 19-page PDF paper in the show notes for anyone who's interested: https://comsec.ethz.ch/wp-content/files/blacksmith_sp22.pdf

They assembled an FAQ to ask and answer some top-of-mind questions:

Are there any DIMMs that are safe?
We did not find any DIMMs that are completely safe. According to our data, some DIMMs are more vulnerable to our new Rowhammer patterns than others.

Which implications do these new results have for me?
Triggering bit flips has become more easy on current DDR4 devices, which facilitates attacks. As DRAM devices in the wild cannot be updated, they will remain vulnerable for many years.

How can I check whether my DRAM is vulnerable?
The code of our Blacksmith Rowhammer fuzzer, which you can use to assess your DRAM device for bit flips, is available on GitHub. We also have an early FPGA version of Blacksmith, and we are working with Google to fully integrate it into an open-source FPGA Rowhammer-testing platform.

Why hasn't JEDEC fixed this issue yet?
A very good question! By now we know, thanks to a better understanding, that solving Rowhammer is hard but not impossible. We believe that there is a lot of bureaucracy involved inside JEDEC that makes it very difficult.

What if I have ECC-capable DIMMs?
Previous work showed that due to the large number of bit flips in current DDR4 devices, ECC cannot provide complete protection against Rowhammer but makes exploitation harder.

What if my system runs with a double refresh rate?
Besides an increased performance overhead and power consumption, previous work showed that doubling the refresh rate is a weak solution not providing complete protection.

Why did you anonymize the name of the memory vendors?
We were forced to anonymize the DRAM vendors of our evaluation. If you are a researcher, please get in touch with us to receive more information.

Does Blacksmith work on DIMMs from other vendors?
According to statistics, the three DRAM chip manufacturers considered by us cover 94% of the DRAM market. However, we have tested Blacksmith on three DRAM devices from another vendor and we could successfully trigger bit flips on these devices too.

So, this is not a story that has a happy ending or anything that we can update or patch.

But this is crucially important research for the industry. We all understand that today's security is porous. And we've never been in a climate that's placing more pressure against our porous security boundaries than now. Bad guys are going to weaponize this research. That's what they do. The problem, as has been observed, is that the world is already filled with DDR3 and DDR4 DRAM, all of which has now been shown to be much more vulnerable, and indeed practically vulnerable, to the consequences of successful bit flipping attacks.

The reason there's been talk of an FPGA — a field programmable gate array — is that their research showed a vast **range** of existing vulnerability, from "very" to "not so much." So it would be possible to create a DRAM access pattern vulnerability testing jig that candidate DRAM could be plugged into to pre-qualify it before deployment. That won't make the DRAM vendors happy, but ultimately this is their problem to solve.