Security Now! #644 - 01-02-18

NSA Fingerprints

This week on Security Now!

This week we discuss a new clever and disheartening abuse of our browser's handy dandy username and password autofill, some recent and frantic scurrying around by many OS kernel developers, a just-released MacOS 0day allowing full local system compromise, another massively popular router falls to the IoT botnets, even high-quality IoT devices have problems, the evolution of adblocking and countermeasures, an important update for Mozilla's Thunderbird, a bit of miscellany, listener feedback, and an update on the NSA's possible intervention into secure encryption standards.

Our Picture of the Week

Security News

Betrayed by Our Browser's AutoFill
Researchers at Princeton's Center for Information Technology Policy have published some new and distressing research:
http://freedom-to-tinker.com/2017/12/27/no-boundaries-for-user-identities-web-trackers-exploit-browser-login-managers/

At least two companies are offering scripts which deliberately trick our web browsers' username & password auto-fill into providing a new source of tracking information.

Adthink (audienceinsights.net)

> What is it?
>
> AudienceInsights is a Web application offering analysis and statistics based on anonymous data collection.
>
> How information is collected?
>
> Our partners' websites and applications host our program that collects information and sends it to our servers where it is stored and analyzed.
>
> Your id on our network is : A008bc977d7aa23e4a31bc5da5fac5f2dea7.
>
> What is collected?
>
> We collect only anonymous data through anonymous cookies and technologies that record:
>
> o Events related to your activity on the partner’s website (such as the number of pages viewed or your searches made on the partner's website),
>
> o Information provided by trusted partners that may include socio-demographic data such as age range.
>
> We do not collect any personal information. We do not know who you are. We do not know your residential address, your email address, your phone number or any other personally identifiable information about you.

What they DON'T SAY is that they deploy hidden scripts running on these "trusted partners" web pages which deliberately trick the user's browser's auto form-fill into populating the user's login username and password for that website... and then send that data back to "AdThink's" server farm for collection.

The raw data IS hashed and converted into a fingerprint, but if usernames are eMail addresses or passwords are reused across sites, this collection allows cross-domain trans-Internet tracking.

And they did also state that "Information provided by trusted partners that may include socio-demographic data such as age range." So while nominally anonymous, it's clear that demographic data is being sought and compiled.

The Princeton researchers looked further and wrote:

The Adthink script contains very detailed categories for personal, financial, physical traits, as well as intents, interests and demographics. It is hard to comment on the exact use of these categories but it gives a glimpse of what our online profiles are made up of:

birth date, age, gender, nationality, height, weight, BMI (body mass index), hair_color (black, brown, blond, auburn, chestnut, red, gray, white), eye_color (amber, blue, brown, grey, green), education, occupation, net_income, raw_income, relationship states, seek_for_gender (m, f, transman, transwoman, couple), pets, location (postcode, town, state, country), loan (type, amount, duration, overindebted), insurance (car, motorbike, home, pet, health, life), card_risk (chargeback, fraud_attempt), has_car(make, model, type, registration, model year, fuel type), tobacco, alcohol, travel (from, to, departure, return), car_hire_driver_age, hotel_stars

127.0.0.1 static.audienceinsights.net
127.0.0.1 api.behavioralengine.com

<quote> We found two scripts using this technique to extract email addresses from login managers on the websites which embed them. These addresses are then hashed and sent to one or more third-party servers. These scripts were present on 1110 of the Alexa top 1 million sites. The process of detecting these scripts is described in our measurement methodology in the Appendix 1. We provide a brief analysis of each script in the sections below.
https://webtransparency.cs.princeton.edu/no_boundaries/autofill_sites.html
https://gist.github.com/BBcan177/b6df57cef74e28d90acf1eec93d62d3b

Demo page: https://senglehardt.com/demo/no_boundaries/loginmanager/index.html

What can we (browsers) do about this?

Kernel Page Table Isolation (KPTI) -- Page Tables MAY be in trouble
What are page tables?

Kernel page-table isolation (KPTI, previously called KAISER) is a hardening technique in the Linux kernel to improve security by better isolating user space and kernel space memory. KPTI was merged into Linux kernel version 4.15, to be released in early 2018, and backported into Linux Kernel 4.14.10. Windows implemented an identical feature in version 17035 (RS4).

Prior to KPTI, whenever executing user space code (applications), Linux would also keep its entire kernel memory mapped in page tables, although protected from access. The advantage is that when the application makes a system call into the kernel or an interrupt is received, [the] kernel's page tables are always present, so most context switching-related overheads (TLB flush, page table swapping, etc) can be avoided.

In 2005, the Linux kernel adopted address space layout randomization (KASLR), which makes it more difficult to exploit kernel vulnerabilities, which relies on kernel addresses remaining hidden from user space. Despite prohibiting access to these kernel mappings, it turns out there are several side-channel attacks in current Intel x86 (not AMD) processors (as of December 2017) that can leak the location of this memory, making it possible to work around KASLR. AMD processors are not affected by these attacks and don't need KPTI to mitigate them.

KPTI fixes these leaks by separating user space and kernel space page tables entirely. On recent x86 processors, a TLB flush can be avoided using the process context identifiers (PCID) feature, but even then it comes at a significant performance cost particularly in syscall-heavy and interrupt-heavy workloads. The overhead was measured to be 0.28% according to KAISER's original authors, but roughly 5% for most workloads by a Linux developer.

KPTI can be disabled with the "nopti" (No Page Table Isolation) kernel boot option. Also provisions were created to disable KPTI if newer processors fix the information leaks.

Kernel developer, Tom Lendacky: AMD processors are not subject to the types of attacks that the kernel page table isolation feature protects against. The AMD microarchitecture does not allow memory references, including speculative references, that access higher privileged data when running in a lesser privileged mode when that access would result in a page fault.

Disable page table isolation by default on AMD processors by not setting the X86_BUG_CPU_INSECURE feature, which controls whether X86_FEATURE_PTI is set.

One possible vector suggested by Matt Tait (pwnallthethings) on Twitter: if speculative operations can influence what the processor does with the cache, the results can be observed with cache timing attacks. If the branch predictor reads the results of speculative operations, it's real easy, as he suggests here:

tl;dr: there is presently an embargoed security bug impacting apparently all contemporary CPU architectures that implement virtual memory [which, yes, is all of them], requiring hardware changes to fully resolve. Urgent development of a software mitigation is being done in the open and recently landed in the Linux kernel, and a similar mitigation began appearing in NT kernels in November. In the worst case the software fix causes huge slowdowns in typical workloads. There are hints the attack impacts common virtualization environments including Amazon EC2 and Google Compute Engine, and additional hints the exact attack may involve a new variant of Rowhammer.
http://pythonsweetness.tumblr.com/post/169166980422/the-mysterious-case-of-the-linux-page-table

KASLR: Kernel Address Space Layout Randomization.

Page Tables:

The critical need for page table entry caching.

If this is a RowHammer-like means to probe page tables, it would be a means of defeating KASLR

In the Linux work, comments in the code have been redacted, and additionally the main documentation file describing the work is presently missing entirely from the Linux source tree.

From a little digging through the FreeBSD source tree, it seems that so far other free operating systems are not implementing page table splitting, however as noted by Alex Ioniscu on Twitter, the work already is not limited to Linux: public NT kernels from as early as November have begun to implement the same technique

https://lwn.net/Articles/742404/

IOHIDeous
On New Years Eve, hacker "Siguza", who describes himself as a hobbyist developer and hacker from Switzerland who goes by the name Siguza, dropped the tweet:

> "Fuck it, dropping a macOS 0day. Happy New Year, everyone. https://t.co/oG2nOlUOjk
> — Siguza (@s1guza) December 31, 2017"

The link in his tweet points to a page on his Github subdomain where, we promised in the tweet, he provides an extensive and extremely well written walk through of this apparently 15+ year old bug.

https://siguza.github.io/IOHIDeous/

<quote> The exploit accompanying this write-up consists of three parts:

- poc (make poc)
  Targets all macOS versions, crashes the kernel to prove the existence of a memory corruption.

- leak (make leak)
  Targets High Sierra, just to prove that no separate KASLR leak is needed.

- hid (make hid)
  Targets Sierra and High Sierra (up to 10.13.1, see README), achieves full kernel r/w and disables SIP [System Integrity Protection] to prove that the vulnerability can be exploited by any unprivileged user on all recent versions of macOS.

Siguza jumped into a dialog about this over on Ycombinator and wrote:

I had actually submitted to the ZDI, but had written the exploit & write-up in the first place mainly because I like hacking rather than for money. I figured I'd see what offers I'd get anyway, but once I had spent all the time on the write-up, I mainly wanted people to see that, and the amount offered wasn't enough to convince me otherwise. I might've published this earlier even, but my December was kinda busy, first with the v0rtex exploit and then with 34C3.

And an engineer from Apple's security team contacted me a bit after releasing - they had found

the bug a while ago, but hadn't verified the subsequent patch which actually didn't fix it. And a while ago I tweeted this https://twitter.com/s1guza/status/921889566549831680 (try diff'ing sources to find it :P). So they do have people on it. I also told that person to extend my condolences to whoever has to come in and fix that now, but they basically said that there's nothing to apologise for and that they (the team) really like such write-ups. So... I guess I'm not that evil?

And I neither wanna watch the world burn nor did anyone brush me the wrong way - I didn't publish this out of hate, but out of love for hacking. If you're concerned about skids hacking you now, they need to get code execution first on your machine. If you're concerned about people who can do that, then those can also get kernel r/w without me, so... nothing really changed for the average user.

PS: Yes, it's really me. Will add keybase proof if my karma gets >= 2. Edit: done, see my profile.

Code Used in Zero Day Huawei Router Attack Made Public
https://threatpost.com/code-used-in-zero-day-huawei-router-attack-made-public/129260/

The exploit code used by one of the several Mirai botnet variants -- called Satori -- which was used to compromise hundreds of thousands of Huawei routers over the past several weeks, was posted to PasteBin and is now public. Researchers expect that the code will quickly become a commodity and be leveraged in DDoS attacks via other existing botnets such as Reaper or IOTrooper.

Two week ago, Check Point Security identified the vulnerability (CVE-2017-17215) in a Huawei home router model HG532 that was being exploited to spread the Mirai variant know either as Mirai Okiru or Satori. Since then, Huawei issued an updated security notice to customers warning that the flaw allows a remote adversary to send malicious packets to port 37215 to execute remote code on vulnerable routers.

The CheckPoint researchers said that zero day exploits how the Huawei router uses of the Universal Plug and Play (UPnP) protocol.

They wrote: "In this case the implementation in the Huawei devices was exposed to WAN through port 37215 (UPnP).” ... and the UPnP framework supports a “DeviceUpgrade” that can carry out a firmware upgrade action. The vulnerability allows remote attackers to execute arbitrary commands by injecting shell meta-characters into the DeviceUpgrade process. After these have been executed, the exploit returns the default HUAWEIUPNP message, and the ‘upgrade’ is initiated.”

Sonos Oh No's
https://t.co/YDT6gbouwH
https://documents.trendmicro.com/assets/pdf/The-Sound-of-a-Targeted-Attack.pdf

Sonos users may have noticed a bit more updating activity from their systems than usual in the late summer and fall of 2017.

Researchers at Trend Micro became curious about the state of security in IoT speaker systems. So they poked at the Sono and Bose offerings... and found many rather glaring information leakage problems.

~4,000 to ~5,000 SONOs systems are publicly exposed to the Internet.

What's exposed is an internal web server which was providing a plethora of information.

The most damaging was probably the account logon information for third-party music services the user had configured.

But also a full scan of the user's internal network. It was possible to issue commands, including a broadcast PING to obtain the private network IPs of every other device on the user's internal LAN... as well as MAC addresses which had been collected in the SONOs ARP table.

It was also possible to peruse the user's music, monitor what was going on at the moment -- what was playing and when, and lots more.

The biggest concern is that these devices were exposed to the public Internet.

A SHODAN search found between 4,000 and 5,000 SONO devices whose quite powerful internal webserver was open and listening for incoming connections... and required NO authentication of any kind. It was all open.

SONOs responded quickly and closed the majority of the leaks... but this is another reason why it is truly crucial for IoT devices to be sequestered on their own WiFi segment which is separate from the household's or office's non IoT devices.

Thousands of major sites are taking silent anti-ad-blocking measures
https://techcrunch.com/2017/12/27/thousands-of-major-sites-are-taking-silent-anti-ad-blocking-measures/

About 1/3rd of the top 10,000 websites are showing some awareness, concern, or powerful anti-adblocking effort.

For a paper to be presented at the Network and Distributed Systems Security Symposium in February of 2018, researchers University of Iowa and UC Riverside carefully profiled the top 10,000 sites, examining the HTML page content returned across multiple visits both with and without overt adblocking installed.

38.2% of the top 1K sites were altering their behavior based upon the blocking behavior of the browser.

And they found evidence of the next escalation of this battle: Anti-adblocker capabilities which arrange to circumvent the blocking behavior of adblockers.

Using Thunderbird? Update if you haven't already
https://arstechnica.com/information-technology/2017/12/mozilla-squashes-critical-thunderbird-bug/

If you're using Thunderbird for email needs make sure you're on version 52.5.2. Mozilla recently released the new version, which has patches that squash a handful of bugs. The bug, rated critical by the Mozilla Foundation, is CVE-2017-7845, which is a buffer overflow vulnerability affecting only Windows users. "A buffer overflow occurs when drawing and validating elements using Direct 3D 9 with the ANGLE graphics library, used for WebGL content," Mozilla said in its security advisory. "This is due to an incorrect value being passed within the library during checks and results in a potentially exploitable crash."

Miscellany

Apple finally admits to what we all knew: The newer iOS releases were DELIBERATELY underclocking the older phones.
https://www.theverge.com/2017/12/28/16827248/apple-iphone-battery-replacement-price-slow-down-apology

- CPU DasherX

Humble Bundle Python Book Bundle
https://www.humblebundle.com/books/python-by-packt-book-bundle

Boston Dynamics' "Atlas" human robot...
https://www.youtube.com/watch?v=fRj34o4hN4I&feature=youtu.be

SpinRite

Adriaan Stander / Location: Le Havre, France
Subject: SpinRite levels
Date: 29 Dec 2017 03:51:01

Dear Steve
I'm longtime SpinRite owner, which I bought not because I had a problem with a drive, but as reward for your work on Security Now. I have never had to use it in anger (i.e. for data recovery), but I do use it from time to time for disk maintenance.

Can you perhaps sometime in the future, on a Security Now podcast, explain the different levels of SpinRite, in a bit more detail that what comes up when you run SpinRite?
In particular, can you please give some practical examples of when levels 1, 3, and 5 may be useful? / Kind regards Adriaan Stander

Closing The Loop

Dallas Haselhorst / @oneoffdallas

Steve, on episode 642 in relation to SpinRite, you mentioned you should not use swap with SSD. I read about this quite extensively some time ago and I was unable to come up with a solid answer one way or another so I was surprised to hear you give it a definitive thumbs down. Does it matter which OS you are using, e.g. Windows vs. FreeBSD? If the system has plenty of RAM, would adding swap cause any harm? Thanks!

The strange story of “Extended Random”

https://blog.cryptographyengineering.com/2017/12/19/the-strange-story-of-extended-random/

Matthew Green (Johns Hopkins University cryptographer and professor)

There is additional circumstantial evidence to strongly suggest that the United States NSA was attempting to compromise encrypted communications.

DUAL_EC_DRBG / RSA's BSAFE library / the default choice.
(For which RSA, as reported by Reuters, received a payment of $10 million from the NSA.)

The trouble is that a bunch of DUAL_EC_DRBG bits must be obtained in order to obtain sufficient "internal state" of the generator, and normal TLS connections fall just shy of providing this much "nonce" information in their handshake.

So... now enter: The "Extended Random" extension to TLS! Matthew writes that: "The Extended Random extension is an IETF Draft proposed by an NSA employee named Margaret Salter (at some point the head of NSA’s Information Assurance Directorate, which worked on “defensive” crypto for DoD) along with Eric Rescorla as a contractor.

It’s important to note that Extended Random by itself does not introduce any cryptographic vulnerabilities. All it does is increase the amount of random data (“nonces”) used in a TLS protocol connection. This shouldn’t hurt TLS at all, and besides it was largely intended for U.S. government machines.

The only thing that’s interesting about Extended Random is what happens when that random data is generated using the Dual EC algorithm. Specifically, this extra data acts as “rocket fuel”, significantly increasing the efficiency of exploiting the Dual EC backdoor to decrypt TLS connections."

However... this was all thought to be rather academic until some older Canon printers were discovered to have this "Extended Random" extension in actual use. This was discovered because the emerging TLS v1.3 specification officially uses the same "Extension Identifier" as the proposed but never ratified "Extended Random" TLS extension.

So... it turns out that not only did RSA's BSAFE library default to the use of DUAL_EC_DRBG, but it *ALSO* implemented the never officially ratified TLS "Extended Random" extension for TLS connections which would have, as Matthew wrote, acted like "rocket fuel" to significantly increase the ease of exploiting the DUAL_EC_DRBG backdoor.

Of course... none of this is proof. But it is the nature of these things that there will likely never be any absolute proof found. However, if this is NOT the story it surely is an interesting collection of synergistic coincidences.