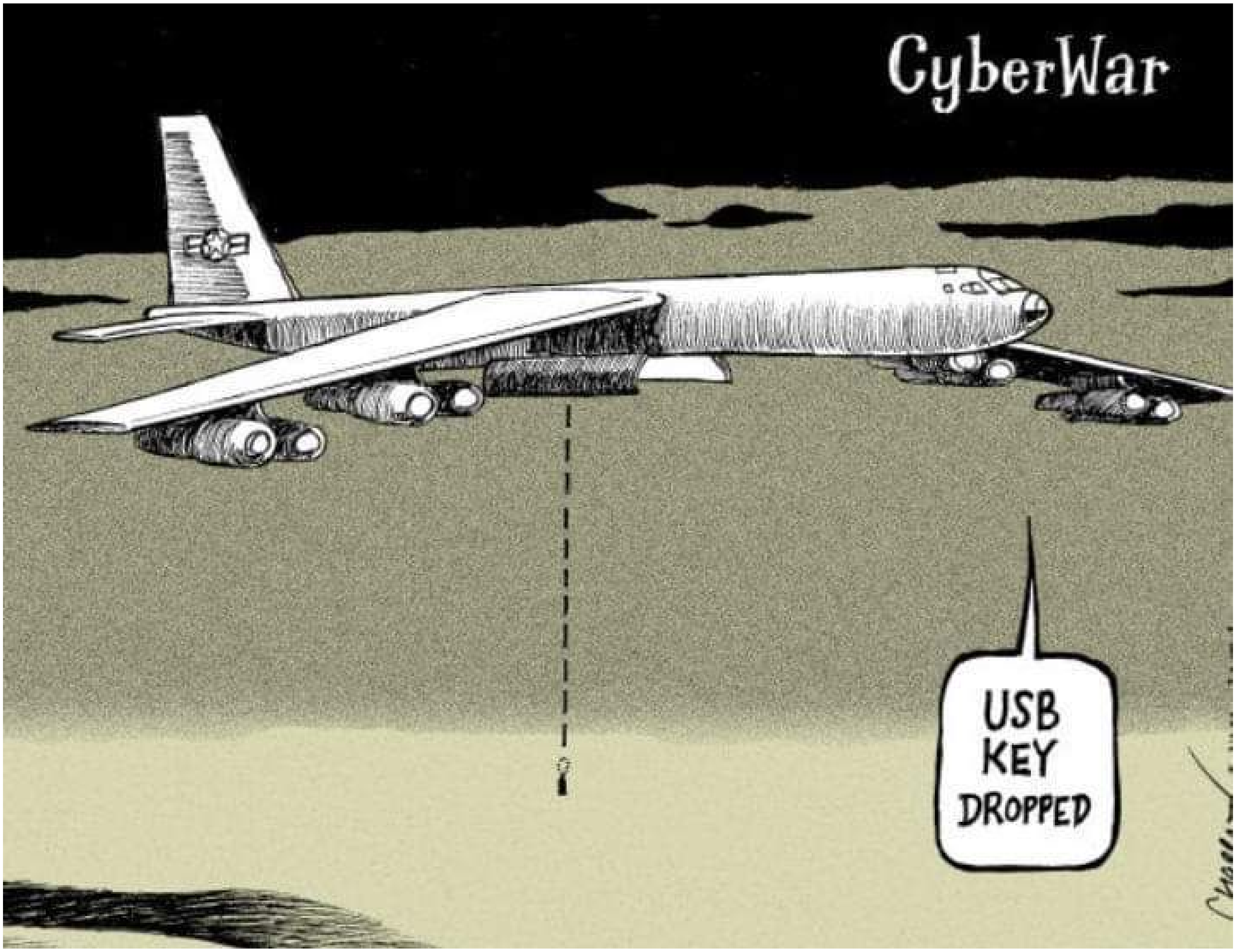
Security Now! #720 - 06-25-19

Bug Bounty Business

This week on Security Now!

This week we check in on the state of last week's Linux TCP SACK kernel panic, examine Mozilla's two 0-days which were being used against Coinbase and others, note that performing a full factory reset of an IoT device may not be sufficient, look at a very clever and elegant solution to OpenSSH key theft via RowHammer attacks, update on the BlueKeep RDP vulnerability, examine the cause of a three-hour widespread Internet outage yesterday morning, NASA's APT which crawled in via a Raspberry Pi, the cost of paying and not paying a ransomware ransom, an update on Microsoft's Chromium-based Edge browser, we handle a bit of listener feedback, then take a closer look at the state of the commercial Bug Bounty Business.



Security News

Checking in on the TCP SACK PANIC...

Everyone may have noted that the Internet is still here. And last week's revelation about the multi-decade Linux and FreeBSD TCP stack flaws that allowed any vulnerable and listening Linux machine to be remotely attacked and halted in a kernel panic has not, so far, resulted in any reports of widespread use and abuse. On the other hand, crashing Linux machines doesn't as obviously bring profit to hackers as using the Exim worm to find, compromise and setup cryptocurrency mining on millions of eMail servers across the Internet.

The guys at SentinelOne have produced a free script for Linux systems to both detect and protect the system from SACK PANICs. The script is up on Github from where it can be downloaded and applied:

https://github.com/Sentinel-One/sack-cve-fixer

Once applied, a system will no longer be vulnerable to any of the three SACK PANIC CVEs.

And, unlike the sysctrl commands we mentioned last week, the changes it makes =will= survive a reboot, making the fix more resilient. It also has a save and undo so that the changes it makes can be reverted if needed.

https://securityboulevard.com/2019/06/linux-admins-grab-our-free-tool-to-protect-against-netflix-sack-panic/

Mozilla patches a pair of Firefox 0-days.

The good news is that 0-days in Firefox have been extremely rare. The most recent previous Firefox 0-day was more than three and a half years ago, in December 2016, when they fixed a security flaw that was being abused to expose and de-anonymize users of the Firefox-derrived Tor Browser.

The bad news in this case was that this past week's pair of 0-days was found being actively exploited against employees of the cryptocurrency exchange Coinbase and presumably other exchanges.

Philip Martin, a member of the Coinbase security team which reported the attacks to Mozilla said: "On Monday (meaning last Monday, June 17th) Coinbase detected & blocked an attempt by an attacker to leverage the reported 0-day, along with a separate 0-day Firefox sandbox escape, to target Coinbase employees."

However, the aspect of this which is disturbing is that Mozilla had reportedly known of the first of the bugs since April 15th, so for a full two months. Samuel Groß, a security researcher with Google Project Zero said that he reported the first bug and while the Coinbase team reported the same bug being used in the attack against them, in their patch Mozilla also credited Samuel's discovery and report. So... apparently... for whatever reason Mozilla just hadn't gotten around to fixing the problem for two months.

ZDNet follow up with a request for additional details from Samuel, who told them: "the bug can be exploited for RCE [remote code execution] but would then need a separate sandbox escape" in order to run code on an underlying operating system. Perhaps Mozilla (wrongly) assumed that it wasn't an emergency so they'd fold its fix into a future security update. But here's another example of multiple problems being teamed up. And the bad guys happened to have a Firefox sandbox escape handy.

In their advisory, Mozilla engineers wrote: "A type confusion vulnerability can occur when manipulating JavaScript objects due to issues in Array.pop. This can allow for an exploitable crash and we are aware of targeted attacks in the wild abusing this flaw."

Fortunately, the pair of Firefox bugs, which were chained into a single exploit and deployed against Coinbase employees, was detected by Coinbase staffers.

If successful, the hackers could have gained access to the Coinbase backend network and used this access to steal funds from the exchange, which is an outcome of security breaches we have seen many times in the past and has led to quite significant losses at many cryptocurrency exchanges.

Philip Martin at Coinbase's security said: "We walked back the entire attack, recovered and reported the 0-day to Firefox, pulled apart the malware and the infrastructure it was using for the attack. We are working with various organizations continue burning down the attacker's infrastructure and digging into the attacker involved. We've seen no evidence of exploitation targeting customers" and Martin also added that other cryptocurrency-linked organizations have also been targeted by this group... And they are being notified.

According to indicators of compromise shared by Martin, attackers would send a spear-phishing email luring victims to a web page, where, if they used Firefox, the page would download and run an info-stealer on their systems that would collect and exfiltrate browser passwords, and other data. The attack was tailored for both Mac and Windows users, deploying different malware for each OS.

The first 0-day patch was pushed out as 67.0.3 and the second sandbox escape was fixed two days later with 67.0.4, which is where everyone should be with Firefox now.

Meanwhile... A full factory reset was not enough to safely resell a Nest cam

Before we go any further I'll make it clear that this was a mistake and that Google promptly fixed it, pushed out an update, and this problem was quickly resolved.

But... For some time before Google was made aware of the problem, it was the case that someone purchasing a previously owned and fully factory reset Nest Cam could still be watched by the camera's previous owners.

Last Wednesday, Wirecutter ran the story with the headline: "Buyer Beware: Used Nest Cams Can Let People Spy on You" -- which was true at the time.
https://thewirecutter.com/blog/used-nest-cams-can-let-people-spy-on-you/

> We’ve explained before that when you’re selling or giving away your old smart-home devices, it’s critical to do a factory reset on them first in order to protect your data and privacy. We’ve recently learned, however, that even performing a factory reset may not be enough to protect privacy for owners of the popular Nest Cam Indoor. And in a twist, this time the risk is on the side of the person receiving the device, not the person disposing of it.
>
> A member of the Facebook Wink Users Group discovered that after selling his Nest cam, he was still able to access images from his old camera—except it wasn’t a feed of his property. Instead, he was tapping into the feed of the new owner, via his Wink account. As the original owner, he had connected the Nest Cam to his Wink smart-home hub, and somehow, even after he reset it, the connection continued.
>
> We decided to test this ourselves and found that, as it happened for the person on Facebook, images from our decommissioned Nest Cam Indoor were still viewable via a previously linked Wink hub account—although instead of a video stream, it was a series of still images snapped every several seconds.
>
> If you buy and set up a used Nest indoor camera that has been paired with a Wink hub, the previous owner may have unfettered access to images from that camera. And we currently don’t know of any cure for this problem.
>
> We are unsure what further implications there may be regarding Nest’s video service, including whether it may be vulnerable to other methods or through other smart-home device integrations. We’re also unsure whether this problem affects the entire Nest lineup, including the Nest Cam Outdoor, Nest Cam IQ Outdoor, and Nest Hello video doorbell.
>
> According to Google, the issue has now been fixed. When we asked about how the company corrected the error, a representative said: “We usually don’t share how a fix was pushed out for various reasons—the statement is our update of record for this.”
>
> Although this particular bug has been fixed, we advise anyone interested in smart-home gear to be especially cautious when considering buying or selling used items, especially ones that have the potential to interfere with your intimate privacy and security, such as cameras, devices with microphones, and smart locks.

With enticing and exciting cloud-connected features exploding into the completely uncontrolled IoT market the caution "buyer beware" has never been more true.

OpenSSH to keep its private keys encrypted at rest

Last Thursday, June 20th, Google security researcher Damien Miller, one of the top OpenSSH and OpenBSD developers, added protection against any and all forms of side-channel attacks by leaving the OpenSSH private keys encrypted until they are needed.

What's weird is that last week, in the context of RAMBleed, I was just talking about this strategy. I noted that my own SQRL client never leaves its keys in the clear in RAM. The user's password or quickpass is used to transiently decrypt the keys briefly for the transaction, after which the plaintext versions are proactively zeroed in RAM. This is clearly the right way to design code for maximum safely in a hostile environment.

So the good news is, the OpenSSH project is getting this protection now, too.

According to Miller, OpenSSH will encrypt SSH (Secure SHell) private keys while they are at rest inside a computer's RAM. If an attacker manages to extract data from a computer or server's RAM, they will only obtain an encrypted version of a SSH private key, rather than the cleartext version. Miller indicated that this protection will be able to stop side-channel attacks like Spectre, Meltdown, Rowhammer, and Rambleed, dead in their tracks.

But... Miller had an additional challenge that I didn't have with my SQRL client. The advantage I had was that the knowledge of the key to decrypt the master key did not need to exist in the client... Since the USER was the one who would be providing it when needed. But that won't work for OpenSSH, since the system must be able to autonomously decrypt the SSH keys on-the-fly without user input whenever they are needed.

Miller's solution was clever and elegant. The comments in his code commit states: "this change encrypts private keys when they are not in use with a symmetic key that is derived from a relatively large 'prekey' consisting of random data (currently 16KB). Attackers must recover the entire prekey [perfectly] before they can attempt to decrypt the shielded private key, but the current generation of attacks have bit error rates that, when applied cumulatively to the entire prekey, make this unlikely."

I didn't dwell on it last week, but the successful OpenSSH key recovery that was used to demonstrate a successful RAMBleed attack utterly depended upon the use of some serious algorithmic post-processing to perform bit-guessing among the always-somewhat-uncertain recovered bits. RAMBleed returns bit probabilities, not certainties. So it turns out that there are some clever algorithms that can use known properties of the public and private key components of a private key to rule out bit many impossible bit combinations. NONE of that can be used against a purely random 16k bit pre-key... and every bit of the pre-key much be perfectly determined. I haven't looked, but I assume that Damien is hashing the pre-key when needed to produce a reduced key which then decrypts the statically-encrypted OpenSSH public key. Since every last bit of the pre-key must be perfectly determined, or (to coin a phrase) the hash will be trash, there is just NO chance that RAMBleed can succeed.

Damien modestly calls his elegant process "Shielding". He said: "Implementation-wise, keys are encrypted 'shielded' when loaded and then automatically and transparently unshielded when used for signatures or when being saved or exporteda."

Miller hopes they'll be able to remove this special protection against side-channel attacks in a few years time when computer architecture has become less unsafe.

OpenSSH is the default SSH client in most operating systems, from OpenBSD (for which it was initially developed) to Windows 10 (the latest OS to support it).

BlueKeep patching status update
As we have previously observed and reported, near the end of May, on Thursday May 30th, 16 days after Microsoft's significant May patch Tuesday, Raviv Tamir, Group Program Manager for Microsoft Threat Protection tweeted:

"My dashboard remains bleak, as only 57% of exposed machines I see worldwide have patched for CVE-2019-0708. GO PATCH!"

Then, the following Wednesday, June 5th, Raviv updated his numbers, tweeting:

> "Numbers are going up - now at 72.4% worldwide. That’s better but still not good enough. KEEP PATCHING! #BlueKeep #MDATP"

And this past Thursday, June 20th, Raviv updated again with a tweet:

> "Another update - worldwide update rate for CVE-2019-0708 numbers are up to 83.4%. KEEEEEEP PATCHING! #BlueKeep #MDATP https://t.co/0xyDebfRes
> — NotNinjaCat (@RavivTamir) June 20, 2019

When BleepingComputer asked Raviv about the number of computers that continue to be vulnerable to BlueKeep, as reported by Microsoft Defender ATP, Tamir said that there are still several million of them.

Verizon negligence caused a major Cloudflare and Amazon customer outage.
Yesterday (Monday) morning, from around 3 to 6 AM PDT, about 2% of the Internet was mistakenly routed through a Pittsburgh, Pennsylvania steel mill. That didn't turn out so well.

A reminder about Internet "Big Iron" routers and BGP: ...

- Routes that are "announced" to adjacent connected routers which, in turn, propagate the routing information.
- The whole "subnetting" thing allows "more specific" routes to be taken.

The Register:

> It all started when new internet routes for more than 20,000 IP address prefixes – roughly two per cent of the internet – were wrongly announced by regional US ISP DQE Communications: this announcement informed the sprawling internet's backbone equipment to thread netizens' traffic through DQE and one of its clients, steel giant Allegheny Technologies, a redirection that was then, mindbogglingly, accepted and passed on to the world by Verizon, a trusted major authority on the internet's highways and byways. This happened because Allegheny is also a customer of Verizon: it too announced the route changes to Verizon, which disseminated them further.
>
> And so, systems around the planet were automatically updated, and connections destined for Facebook, Cloudflare, and others, ended up going through DQE and Allegheny, which buckled under the strain, causing traffic to disappear into a black hole.

> Internet engineers blamed a piece of automated networking software – a BGP optimizer built by Noction – that was used by DQE to improve its connectivity. And even though these kinds of misconfigurations happen every day, there is significant frustration and even disbelief that a US telco as large as Verizon would pass on this amount of incorrect routing information.
>
> Routes across the internet change pretty much constantly, rapidly, and automatically 24 hours a day as the internet continuously reshapes itself as links open and close. A lot breaks and is repaired without any human intervention. However, a sudden large erroneous change like today's route change should have been caught by filters within Verizon and never accepted and disseminated.
>
> "While it is easy to point at the alleged BGP optimizer as the root cause, I do think we now have observed a cascading catastrophic failure both in process and technologies," complained Job Snijders, an internet architect for NTT Communications, in a memo today on a network operators' mailing list.

NANOG mailing list:

This is what looked happened:

There was a large scale BGP 'leak' incident causing about 20k prefixes for 2400 network (ASNs) to be rerouted through AS396531 (a steel plant) and then on to its transit provider: Verizon (AS701)

(ASN: Autonomous System Number)

Start time: 10:34:21 (UTC) End time: 12:37 (UTC)

All ASpaths had the following in common: 701 396531 33154

33154 (DQECOM) is an ISP providing transit to 396531.
396531 is by the looks of it a steel plant. dual homed to 701 and 33154.
701 is verizon and accepted by the looks of it all BGP announcementsfrom 396531

What appears to have happened is that 33154 those routes were propagated to 396531, which then send them to Verizon and voila... there is the full leak at work.

(DQECOM runs a BGP optimizer (https://www.noction.com/clients/dqe, thanks Job for pointing that out, more below)

As a result traffic for 20k prefixes or so was now rerouted through verizon and 396531 (the steel plant)

We've seen numerous incidents like this in the past lessons learned:
1) if you do use a BGP optimizer, please FILTER!
2) Verizon... filter your customers, please!

Since the BGP optimizer introduces new more specific routes, a lot of traffic for high traffic destinations would have been rerouted through that path, which would have been congested, causing the outages. There were many cloudflare prefixes affected, but also folks like Amazon, Akamai, Facebook, Apple, Linode etc.

here's one example for Amazon - CloudFront : 52.84.32.0/22. Normally announced as a 52.84.32.0/21 but during the incident as a /22 (remember more specifics always win)

https://stat.ripe.net/52.84.32.0%2F22#tabId=routing&routing_bgplay.ignoreReannouncements=false&routing_bgplay.resource=52.84.32.0/22&routing_bgplay.starttime=1561337999&routing_bgplay.endtime=1561377599&routing_bgplay.rrcs=0,1,2,5,6,7,10,11,13,14,15,16,18,20&routing_bgplay.instant=null&routing_bgplay.type=bgp

RPKI would have worked here (assuming you're strict with the max length)!

Cheers, Andree

We all know that mistakes can happen, but the fact that Verizon wasn't performing any sanity-checking is unconscionable. There's no way that one of their customers, whose networking services they provide and thus they know which IPs they have, can "announce" that they control IPs that was FAR outside of their control.

But the biggest problem is that NO ONE WAS AVAILABLE at Verizon's NOC -- Network Operations Center. Matthew Prince, Cloudflare's CEO was livid and tweeted:

> *"It's networking malpractice that the NOC at @verizon has still not replied to messages from other networking teams they impacted, including ours, hours after they mistakenly leaked a large chunk of the Internet's routing table."*
> — Matthew Prince (@eastdakota) June 24, 2019

NASA was infected by an APT for more than a year
... caused by an unauthorized Raspberry PI connected to the network.

In a report published last week by the NASA Office of Inspector General (OIG) revealed that in April 2018 hackers breached the agency's network and stole approximately 500 MB of data related to Mars missions.

The point of entry... was a Raspberry Pi connected to the network at NASA's Jet Propulsion Laboratory (JPL) without authorization or going through the proper security review. According to the 49-page OIG report, the hackers used this point of entry to move deeper inside the JPL network by hacking a shared network gateway.

The hackers used this network gateway to pivot inside JPL's infrastructure, and gained access to the network that was storing information about NASA JPL-managed Mars missions, from where they exfiltrated information.

<quote> "The attacker exfiltrated approximately 500 megabytes of data from 23 files, 2 of which contained International Traffic in Arms Regulations information related to the Mars Science Laboratory mission." The Mars Science Laboratory is the JPL program that manages the Curiosity rover on Mars and other projects.

The JPL division's primary role is to build and operate planetary robotic spacecraft such as the Curiosity rover and the various satellites that orbit planets in the solar system. JPL also manages NASA's Deep Space Network worldwide network of satellite dishes used to communicate with NASA spacecraft during missions.

Investigators said that besides accessing the JPL's mission network, the April 2018 intruder also accessed the Deep Space Network's IT network. Upon discovery of the intrusion, several NASA facilities disconnected from the JPL and DSN networks, fearing the attacker might pivot to their systems as well.

NASA's OIG said: "Classified as an advanced persistent threat, the attack went undetected for nearly a year, and the the investigation into this incident is ongoing."

The report blamed the JPL's failure to segment its internal network into smaller segments.

As our long time listeners know, we've been talking about the need for strong network segmentation for years, even at home or in small offices. The reason we liked the amazing little Ubiquity five-port routers so much was that for $49 they contained five entirely separate NIC adapters allowing strong network segmentation.

But the reason JPL probably didn't bother with network segmentation is the same reason most home and small office users don't bother... it's a pain in the butt to maintain separate networks. It's SO MUCH EASIER to have everything able to see and talk to each other. But that's also exactly the problem, since anything malicious that might get in anywhere then also has access to everything else.

Riviera Beach, Florida... Ouch!!

...pays $600,000, in 65 Bitcoin, ransom.

Riviera Beach, a Florida city, is coughing up $600,000 to hackers after a ransomware attack brought down its computer systems.

A Florida city, hit by a ransomware attack that crippled its computer systems for three weeks, voted this week to pay the attackers the requested ransom of $600,000.

Riviera Beach, a city in Florida populated by 35,000, was hit by the ransomware attack May 29 after a city employee clicked on a malicious link in an email, according to local reports. Attackers behind the malware, which spread throughout the city’s network and shut down its computer systems, asked for a ransom of 65 Bitcoin (worth around $600,000) in exchange for unlocking the computers.

In Riviera Beach, systems controlling the water utility were offline, government email and phone systems were non-functional, and 911 calls couldn’t enter into computer databases. According to local reports, the computer systems controlling city finances and water utility pump stations are partially back online.

So, last Monday meeting, the city council voted unanimously to authorize its insurer to pay the $600,000 ransom.

The security community, for its part, has argued that the city is taking a “big gamble” in paying the ransom.

(And that's a useful thing to remember: there's no guarantee that the bad guys will pay. But they should, since being paid by other future victims does depend upon the ability to recover the encrypted data after paying.)

Shlomie Liberow, technical program manager at HackerOne, said: “The Riviera Beach City Council has taken a big gamble by paying the ransom as there are no guarantees the attackers will return any of the data, which could leave the city in an even worse situation. By paying the ransom, the council also encourages more of these types of attacks as it makes it more profitable for attackers.”

Nobody wants to payoff criminals, but sometimes saying no is much more costly. Here's two recent examples recently cited by ThreatPost in their coverage of ransomware:

Last year, several Atlanta, Georgia city systems were crippled by a ransomware attack. A ransom of $51,000 was demanded for recovery. Atlanta said "no thanks" and ended up spending $2.6 million in recovery costs, including incident response and digital forensics, additional staffing and Microsoft Cloud infrastructure expertise.

The city of Baltimore, Maryland is another recent victim of ransomware, which hit in May and halted some city services like water bills, permits and more. The ransom demanded was $76,000 ransom. Baltimore also said "no" and ended up spending $18.2 million in restoration costs and lost revenue.

So while Riviera Florida's $600,000 is a lot of bitcoin, it appears that these cities are really not in a position to "restore from backups" and get on about their day.

Microsoft's latest Edge browser for Win7

https://www.microsoftedgeinsider.com/en-us/

As we've mentioned, Edge is being released in "Beta", "Dev" and "Canary" channels. And as has been the case with their previous releases, only the "Canary" channel is initially available for Windows 7, but as it solidifies the Dev and Beta will be added.

Closing The Loop

Hafthor @hafthor
I think you might be wrong about worming the RDP bug -- the advantage would be to gain access to machines on the LAN to have them mine as well.

Bug Bounty Update

Bug bounties have become a permanent feature of today's software and system security ecosystem. As we know, there are white hat hacker competitions such as Pwn2Own sponsored by the large sponsoring companies, some of whom have their products hacked and their defects responsibly disclosed for pre-disclosure repair.

Then there are somewhat sketchy outfits such as Zerodium whose tag line reads: "The leading exploit acquisition platform for premium zero-days and advanced cybersecurity capabilities." They pay big bucks for big exploits which they presumably resell to major player state-level actors and foreign and domestic law enforcement and intelligence services.

And, finally, we have White Hat Bug Bounty clearinghouse middlemen, like HackerOne, who serve as matchmakers between those who put up bounties for the responsible discovery and reporting of bugs in their own products. In contrast to Zerodium's pith, Hacker One states: "More Fortune 500 and Forbes Global 1,000 companies trust HackerOne to test and secure the applications they depend on to run their business."

Hacker One reports that for organizations that found vulnerabilities before they were exploited using HackerOne, Forrester found benefits of up to $1.6 million and an ROI of up to 646%.

Hacker One touts "From implementing the basics of a vulnerability disclosure process, to supercharging your existing security programs via a bug bounty program, HackerOne has you covered." -and- "More security teams use HackerOne to manage vulnerability disclosure and bug bounty programs than any other platform."

As for the benefits to using HackerOne:

- Establish a compliant process for receiving and acting on vulnerabilities discovered by third-parties
- Ensure bugs found by security researchers, ethical hackers, or other external parties reach the right people in your organization.

Improve your Pen Test results with a project-based vulnerability assessment program
Capture the intelligence of our trusted community in a time-bound program that consistently outperforms traditional penetration testing.

Launch a private, fully-managed bug bounty program for continuous coverage
Take a proactive approach to finding critical vulnerabilities across your critical surfaces with the full support of HackerOne's security experts.

So, essentially, HackerOne provides a means for companies to outsource the establishment and management of what would otherwise have to be an internal effort to locate bugs. And as we've observed when we've talked about this previously, nothing is more difficult -- or really, impossible -- than for a coder to locate their own bugs. So much assumption and context and ego is involved that it really takes someone who can truly adopt an adversarial posture to attack someone else's work with gusto to find and illuminate their faulty code.

As General Motors' Vice Prsident of Global Cybersecurity, Jeffrey Massimilla was quoted:
"Hackers have become an essential part of our security ecosystem."

So... How is HackerOne doing with this? Pwn2Own is great, but it's a contest format which has its pluses and minuses. And Zerodium is... well... Zerodium. My feeling is that the HackerOne model is the most practical one if the goal is to allow companies to end up with the most robust products and systems possible. ZDNet recently pulled some numbers together about HackerOne and that triggered my thought that it would be interesting to take a look at the overall success by client.

But let's first take a look at the disaggregated bug-by-bug and bounty-by-bounty transaction log, which HackerOne posts on their website: Let's scroll through the "Hacktivity" page...

https://hackerone.com/hacktivity?order_direction=DESC&order_field=latest_disclosable_activity_at&filter=type%3Abounty-awarded

At the head of the list of the top 20 biggest, fastest and most lucrative bounty programs on HackerOne, in first place is Verizon, whoSe program has been in place for nearly five and a half years, having been initiated in February of 2014. During this time Verizon has paid out more than $4 million in bounties, with a top award of $6,000 and a total of 5,269 bug reports resolved since their program was launched.

Taking the #2 position on the list, having launched two years later, in March of 2016, is Uber. Since launch Ubeer has shelled out more than $1.8 million in bounties, paying a top bounty of $15,000 and resolving 1,172 problems.

Behind Uber in 3rd place is PayPal. Since just September of 2018 they have paid out more than $1,170,000 for a total of 430 reports resolved and a top bounty of $30,000 which was twice Uber's maximum and 5 times Verizon's maximum.

Taking #4 is Shopify, since April 2015, paying more than $1.1 million across 996 reports with a top payout of $25,000.

Then we have Twitter in 5th place. Since May of 2014, paying out the same as Shopify -- $1.1 million -- across 995 reports and a top bounty of $15,120.

Intel is #6 with $800,000 since February 2018 but no additional details.

Airbnb takes 7th place, paying out more than $600,000 since joining in February 2015. A respectable top bounty of $15,000 and a total of 508 reports resolved.

Even though I've become a big fan of the newly released NetGate SG-1100 with FreeBSD and its pfSense router/firewall, I was glad to see that our previous power-router friends at Ubiquity have been participating at HackerOne since March 2015, during which time they have paid out more than $600,000 across 765 reports.

The gaming powerhouse "Valve" is there too, 9th in line for the past year since May of 2018. Valve has paid out $20,000 across 470 reports.

GitLab is in the middle of the top 20, having paid out more than $570,000 since February 2018 with a top bounty of $12,000 over 318 reports.

And right behind GitLab we have GitHub at spot 11. GitHub joined in April of 2016, paid a top bounty to $20,000 and a total of $520,000 for 348 reports.

Slack is next down with $420,000 in total payout over 838 reports and a maximum single bounty of $10,000.

Starbucks is in 13th place with more than $300,000 since November 2016

Continuing to the end we have Mail.ru, Grab, Coinbase, Snapchat (max bounty of $25,000), HackerOne themselves, DropBox (with a nice bounty maximum of $23,058), and finally "VK" Europe's largest social network.

I wanted to share these numbers and specific details to drive home the reality of the clear fact that many major corporations like General Motors, Starbucks, Spotify, and the EU, see today's 3rd-party outsourced Bug Bounty programs as a serious and important aspect of their overall software and online security strategy. There's no doubt that the operating budgets of these companies now has a line item labelled "Bug Bounty Program" and that there is some serious scratch with a bunch of zeroes on that line of the ledger.

And our individual listeners should consider that those bounties are being paid out to hackers just like them, and that the result, unlike the situation with SandboxEscaper or Zerodium, is a demonstrably safer and more secure Internet for everyone else to use and enjoy.

~30~