Security Now! #885 - 08-23-22
The Bumblebee Loader

Starting into our 18th year!

This week on Security Now!

This week we'll start off with a bit of fun over the most tweeted by far wacky tech news item. We then get serious with a very worrisome flaw which very likely exists in the WAN interface of the routers that many of us probably own. DDoS attacks have broken another record by a large margin, and both Chrome and Apple deal with, if not emergency then at least high priority software updates. We also have another major software repository tightening up its security against supply chain attacks. Then after sharing just a few, but powerful, bits of feedback, we're going to step through the blow-by-blow operation and actions of the newest and meanest kid on the block with the emergence of a powerful malware loader that gets its name from the DLL it first loads: Bumblebee.

What The Heck??

(Here's a YouTube Video of the event…)

A train of 53 Starlink satellites launched from Florida on Friday.
https://findstarlink.com/

Security News

I've got the music in me…
I needed to respond to what was, by far, the most Tweeted-to-me news item in a **long** time. And our listeners, who are naturally on top of their game, felt about this pretty much as I do.

For more than two and a half decades the highly respected Microsoft engineer, Raymond Chen, has been blogging. Last Tuesday he posted a blog entry that was just so weird that everyone picked up on it. Raymond's posting was titled "Janet Jackson had the power to crash laptop computers". The fact that this has been assigned a CVE number (CVE-2022-38392) has apparently lent it more credibility, or at least more notoriety, than I think it deserves. And the fact that the CVE refers to Raymond's blog as its sole reference seems to be somewhat self-referential: Raymond cites the CVE which cites Raymond.

I'm actually wondering whether it was a slow Blog week and Raymond may have needed a bit of filler. His blog post opens with two lines: *"A colleague of mine shared a story from Windows XP product support.* [Uh. Okay. Well, that wasn't recent!] *A major computer manufacturer discovered that playing the music video for Janet Jackson's "Rhythm Nation" would crash certain models of laptops."* ... Click bait, anyone?

From the CVE we learn that this was *"certain models of laptops"* circa 2005, so seventeen years ago. The CVE's formal description says: *"A certain 5400 RPM OEM hard drive, as shipped with laptop PCs in approximately 2005, allows physically proximate attackers to cause a denial of service (device malfunction and system crash) via a resonant-frequency attack with the audio signal from the Rhythm Nation music video."* So we can see why the tech press thought that this was just too wonderful to pass up. On the other hand, we have a **C.V.E.** that was apparently issued based upon what amounts to *"a friend told me"* rumor — no mention of the make or model of 5400 RPM OEM hard drive that should be kept away from discos.

So this begs the question of just how low the bar has been set for issuing CVE's? This is not an attack — although there are a vocal group of people who feel that any playing of Janet Jackson's Rhythm Nation should qualify as a form of terrorism. And neither is it a bug that needs to be fixed, nor malware that needs to be expunged. There's no action that can or should be taken today — it's from 17 years ago. So why give this old "heard it from an XP support guy" a CVE in 2022? I have no idea.

Those who have been following this podcast will recall that video, which was also cited in some of the coverage of this Rhythm Nation Hard drive DDoS Attack, where someone was monitoring the dynamic throughput of an array of spinning hard disk drives while screaming at the array at the top of his lungs. And sure enough, the throughput dropped during the screaming. As we noted at the time, the throughput dropped because modern mechanical hard drives have crammed their tracks so closely together that they have become quite sensitive to any exogenous vibration. In fact, the way they are mounted in server chassis can be critical.

Raymond, of course, also referred to the famous video showing the 1940 collapse of the Tacoma Narrows Bridge. In the same way that Rhythm Nation was able to rub some hard drives the wrong way in 2005, the coincidentally-timed gusts of wind through the Tacoma Narrows rubbed the Bridge the wrong way, until it disintegrated.

Anyway, I felt that this podcast needed to acknowledge this story that everyone Tweeted to me, and that most of the tech press had fun covering. I did too.

RealTek SoC flaw affects many millions of IoT devices
During one of the recent DEFCON presentations in Las Vegas, a team of four Argentinian researchers from the cybersecurity company Faraday Security detailed their discovery of what was subsequently classified as a CVSS 9.8 0-click Remote Code Execution vulnerability in the interface stack which Realtek provides in the SDK for their hardware's use with the popular open-source eCos operating system. Since they had previously responsibly disclosed their discovery, and RealTek had patched the flaw back in March, their presentation provided full disclosure of the technical details needed to replicate the attack.

Consequently, there is now exploit code released publicly for this critical vulnerability affecting networking devices which use Realtek's RTL819x system on a chip (SoC)... And those devices number in the tens of millions. Being of the turnkey, consumer plug it in and forget it variety, there's little chance that most of these tens of millions of devices are ever going to be updated. Many will have long since gone out of warranty. Since this RealTek System on a Chip RTL819x is incredibly popular, we're talking about devices that many of us probably already have since the chips are used by more than 60 vendors including ASUSTek, Belkin, Buffalo, D-Link, Edimax, TRENDnet, and Zyxel.

The vulnerability presents on the WAN interface and because it's a stack-based buffer overflow, it allows for zero-click instant compromise of the host upon receiving a packet from the public Internet. The DEFCON presentation left nothing to the imagination. The SIP ALG (application layer gateway) function that rewrites SDP data has a stack-based buffer overflow. This allows an attacker to remotely execute code without authentication via a crafted SIP packet that contains malicious SDP data.

We've spoken about the abuse of application layer gateways in the past. ALG's are essentially enhancements to baseline NAT routing to handle what would otherwise be NAT's interference with the details of specific NAT-unfriendly protocols. The simplest example is the original FTP protocol where the client instructs the server which port it has opened to receive the reverse connection. The router's application layer gateway code monitors the outgoing data, sees the port being specified by the FTP client, and either opens that port in its WAN side interface, or modifies the outbound port specification to a port it wishes to open. The point being that it allows A NAT router to become transparent to the NAT-hostile FTP protocol. In the case of this vulnerability, the trouble exists in the application layer gateway's logic for handling SIP (session initiation protocol) used in VoIP (voice over IP) systems. It's not clear whether disabling SIP's ALG, it that's an option, would help.

Johannes Ullrich, Dean of Research at SANS says that a remote attacker could exploit the vulnerability for the following actions. They could: crash the device, execute arbitrary code, establish backdoors for persistence, reroute or intercept network traffic. Basically, take over any vulnerable router. And he warned that if an exploit for CVE-2022-27255 turns into a worm, it could spread throughout the internet in minutes.

While Johannes is technically correct about a worm, as I've been saying recently, a massive worm attack no longer makes any sense. They made sense back when eMail viruses just existed to see whether they could. In today's world, anyone who's capable of writing a working worm is also capable of using the router as a proxy to bounce malicious traffic, to quietly mine cryptocurrency, to enslave the router into the service of one of today's massive BotNets (as we'll see in the next story) or to pivot into the network behind the router to see whether there might be something juicy worth attacking somewhere on that router's LAN.

For their part, the four security researchers said that:

- Devices using firmware built around the Realtek eCOS SDK before March 2022 are vulnerable
- Users are vulnerable even if they do not expose **any** admin interface functionality
- Attackers may use **a single UDP packet to an arbitrary port** to exploit the vulnerability
- This vulnerability will likely affect routers the most, but some IoT devices built around Realtek's SDK may also be affected

https://www.realtek.com/images/safe-report/Realtek_APRouter_SDK_Advisory-CVE-2022-27255.pdf

RealTek's vulnerability report is two pages and provides zero guidance. There's no list of manufacturers or makes and models of affected products. There's no action that any responsible end user can take. There's no obvious way to know whether any particular router or IoT device might be affected. The only recourse would be to proactively verify that your router is running the latest firmware available for it from its vendor, and to hope that they care enough to update their firmware for the model of router you have. If your system can run one of the alternative router firmware systems such as DD-WRT, that would be one way to move it to safety.

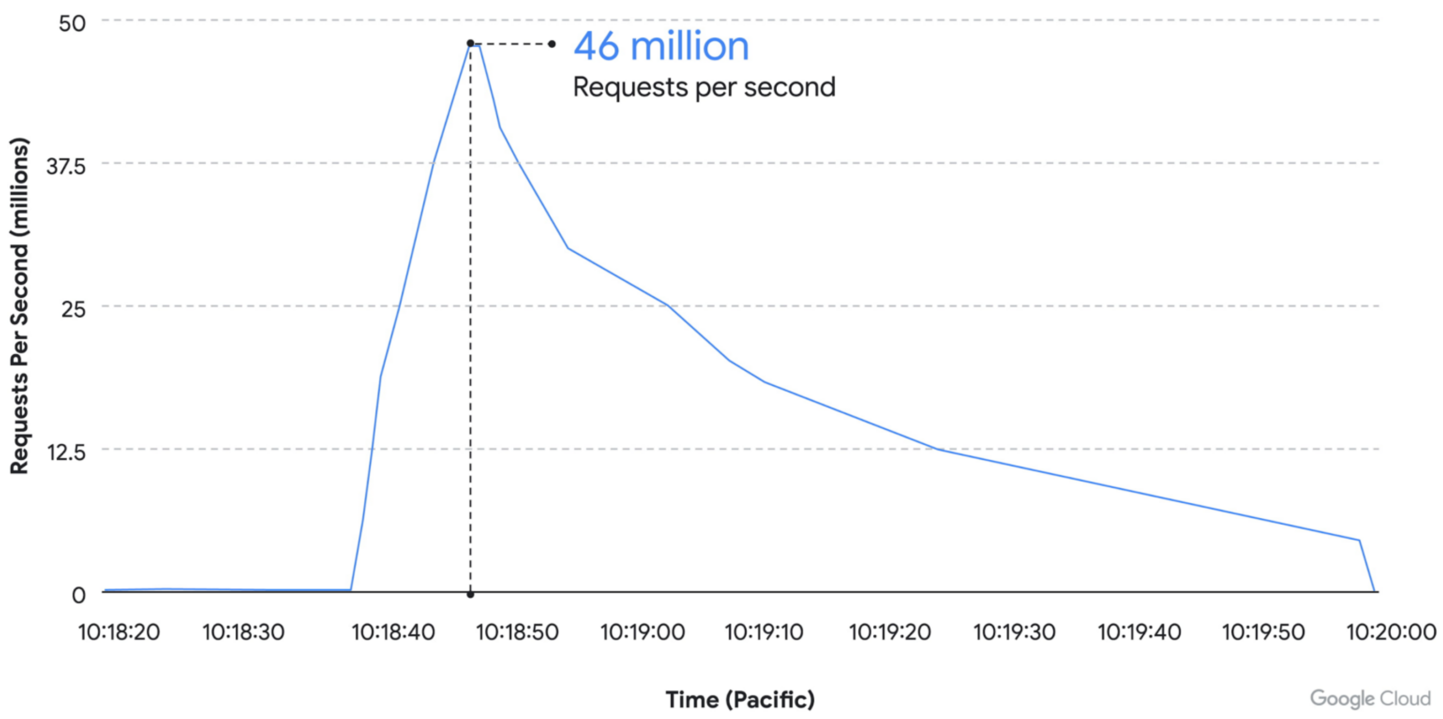
So many of these sorts of things have come out through the years; problems that are unlikely to ever be fixed, that it's possible to imagine the sort of massive known-vulnerability database that both nation states and sufficiently large criminal enterprises must now be maintaining. You want to get into which organization? What equipment do they have on their border? Look up all of the known exploits against it and start working down the list until you get in.

17 years ago, back when we were launching this podcast, that scenario would have seemed like pure speculative fiction. Today I'd lay money down that such databases must now exist all over the world. And it's difficult to see how this changes. Those who can afford to be truly aware of and concerned about security could choose to use a non-consumer router on their borders, such as something running pfSense. But that will still leave the vast majority of end user consumers potentially vulnerable to decades of previous vulnerabilities.

46 Million RPS - requests per second

After standing up to the largest-ever DDoS attack on behalf of one of its "Cloud Armor Adaptive Protection" customers, Google said it had blocked a record-breaking HTTPS-based DDoS attack that hit, at its peak, a whopping 46 million requests per second. That puts it 76% higher than the 26 million RPS attack which was previously mitigated by Cloudflare in June. Google chose not to disclose the target of this attack, but said that it believes the attack was carried out with the help of the Meris botnet.

To put the scale of this attack in perspective, it's an HTTPS request rate equivalent to receiving all of the requests to the Wikipedia domain — one of the top 10 traffic domains in the world — received by Wikipedia during one 24-hour period, compressed into just 10 seconds.



https://cloud.google.com/blog/products/identity-security/how-google-cloud-blocked-largest-layer-7-ddos-attack-at-46-million-rps

Google's report of the attack contains lots of interesting details. Here's what they shared:

Starting around 9:45 a.m. PT on June 1, 2022, an attack of more than 10,000 requests per second (RPS) began targeting our customer's HTTP/S Load Balancer. Eight minutes later, the attack grew to 100,000 requests per second. Cloud Armor Adaptive Protection detected the attack and generated an alert containing the attack signature by assessing the traffic across several dozen features and attributes. The alert included a recommended rule to block on the malicious signature.

Our customer's network security team deployed the Cloud Armor-recommended rule into their security policy, and it immediately started blocking the attack traffic. In the two minutes that followed, the attack began to ramp up, growing from 100,000 RPS to a peak of 46 million RPS. Since Cloud Armor was already blocking the attack traffic, the target workload continued to operate normally. Over the next few minutes, the attack started to decrease in size, ultimately ending 69 minutes later at 10:54 a.m. Presumably, the attacker determined they were not having the desired impact while incurring significant expenses to execute the attack.

I would argue that point. I suspect that the attack cost the attackers nothing whatsoever other than the exposure of the IP addresses of their fleet of infected consumer routers hosting the Meris botnet.

In any event, Google then proceeded to analyze the attack. They wrote:

In addition to its unexpectedly high volume of traffic, the attack had other noteworthy characteristics. There were 5,256 source IPs from 132 countries contributing to the attack. The top 4 countries [Brazil, India, Russia and Indonesia] contributed approximately 31% of the total attack traffic. The attack leveraged encrypted requests (HTTPS) which would have taken added computing resources to generate.

Again, Google appears to be deliberately missing the whole point: If there were 5,256 observed source IPs, then that crypto burden will have been well distributed across the globe. And then we learn that HTTPS pipelining was also in use, further limiting the crypto overhead:

Although terminating the encryption was necessary to inspect the traffic and effectively mitigate the attack, the use of HTTP Pipelining required Google to complete relatively few TLS handshakes.

Right, and thus also much less burden on the attackers. The attackers were establishing a single TLS secure connection, then attempting to flood that connection with pipelined HTTPS requests. There's no reason to believe that any IP that's flooding HTTPS requests down the pipe will ever generate a valid request. So those IPs should have and hopefully were dynamically blacklisted.

Approximately 22% (1,169) of the source IPs corresponded to Tor exit nodes, although the request volume coming from those nodes represented just 3% of the attack traffic. While we believe Tor participation in the attack was incidental due to the nature of the vulnerable services, even at 3% of the peak (greater than 1.3 million RPS) our analysis shows that Tor exit-nodes can send a significant amount of unwelcome traffic to web applications and services.

The geographic distribution and types of unsecured services leveraged to generate the attack matches the Mēris family of attacks. Known for its massive attacks that have broken DDoS records, the Mēris method abuses unsecured proxies to obfuscate the true origin of the attacks.

Ah... so attackers were bouncing their Botnet's traffic through intermediate proxies in order to protect the IPs of their actual Bot agents.

I couldn't help but note that in their redacted report, Google used text blurring to obscure the identity of their customer. And in some beautiful work that we covered a while back, what did we learn about the security of blurred text? We learned that it doesn't work. If someone is interested in learning what original text lies behind the blurred instance, they can identify the details of the typography from all of the samples of non-blurred text, then iteratively guess the text that's behind, employ the same blurring, and compare the result of the two blurrings, one they control and one they do not.

Anyway, Google's report then switches into marketing mode, bragging about their technology, which anyone would have to agree, definitely works. So we're now living in a world where those whose Internet web services **must** remain online in the face of attacks will need to bear the added cost of the privilege.

Chrome's 5th 0-Day of 2022
Last Tuesday, Google updated our Chrome browser for desktops to squash an actively exploited high-severity 0-day flaw in the wild.

Tracked as CVE-2022-2856, the issue is a case of insufficient validation of untrusted input in Intents. Security researchers Ashley Shen and Christian Resell of Google's TAG team (their Threat Analysis Group) are credited with reporting the flaw last month on July 19th.

As usual, there's no upside for Google to share anything more with us beyond "please be sure that your version of Chrome now ends in a .102" They did add that "Google is aware that an exploit for CVE-2022-2856 exists in the wild."

In addition to stomping on that 5th of the year actively exploited flaw, that update to .102 addressed 10 other security flaws, most of which relate to use-after-free bugs in various Chrome components. They also fixed a heap buffer overflow vulnerability in Downloads.

So, this is #5 this year. Previously, we had a use-after-free in Animation, two type confusion bugs in V8 and a heap buffer overflow in WebRTC. And now #5 is a rather vague "insufficient validation of untrusted input." Okay. If you're using one of the non-Chrome chromium siblings, Edge, Brave, Opera, or Vivaldi, be sure to update there, too.

Apple: Not to be left behind...
Last Wednesday, Apple released high-priority security updates for iOS, iPadOS, and macOS platforms to remediate a pair of 0-day vulnerabilities which were being exploited by threat actors to compromise Apple's users' devices.

CVE-2022-32893 - An out-of-bounds bug in WebKit which could lead to the execution of arbitrary code by processing a specially crafted web content.

CVE-2022-32894 - An out-of-bounds bug in the operating system's Kernel that could be abused by a malicious application to execute arbitrary code with the highest privileges.

Apple said that it had addressed both the issues with improved bounds checking, adding it's aware the vulnerabilities "may have been actively exploited." Uh huh. And please update immediately before you do anything else.

As usual, Apple didn't disclose any additional information regarding these attacks or the identities of the bad guys who have been seen using them, though, as usual, it's almost certain that they were abused as part of highly-targeted intrusions.

And since we're counting 0-days so far this year, this latest update brings Apple's total of actively exploited 0-days to 6 so far during 2022. As with Chrome, we had four before now:

The first was in IOMobileFrameBuffer, where a malicious application could execute arbitrary code with kernel privileges. The second was in WebKit, where processing maliciously crafted web content could lead to arbitrary code execution. The third was in the Intel Graphics Driver, where an application could read kernel memory (whoops!). And the fourth was in AppleAVD, where an application could execute arbitrary code with kernel privileges.

Now we have two more to add to that list and all still-supported devices should be updated. Both the vulnerabilities have been fixed in iOS 15.6.1, iPadOS 15.6.1, and macOS Monterey 12.5.1.

RubyGems to require MFA
As we know, RubyGems is the official package manager for the Ruby programming language. And in a welcome response to the increasing threat and prevalence of supply chain attacks, the RubyGems repository has become the latest platform to require multi-factor authentication (MFA) for its more popular package maintainers, In this it's following the footsteps of NPM and PyPI.

Specifically, owners of "gems" having more than 180 million total downloads are now, as of last Monday, August 15th, required to enable MFA. The RubyGems management said:

"Users in this category who do not have MFA enabled on the UI and API or UI and gem sign-in level will not be able to edit their profile on the web, perform privileged actions (i.e. push and yank gems, or add and remove gem owners), or sign in on the command line until they configure MFA."

Also, as gem downloads approach that magic mandatory 180 million count, once downloads surpass 165 million cumulative downloads, their maintainers will receive reminders to turn on MFA before the download count hits 180 million, at which point it will be made mandatory.

This is a further welcome attempt by package ecosystems to improve the past's casual approach to software supply chain security by working to prevent account takeover attacks, which could then enable malicious actors to leverage their access to push rogue packages to downstream customers.

As we know, adversaries are increasingly setting their sights on open source code repositories, with attacks on NPM and PyPI snowballing by a combined 289% since 2018. Researchers from Checkmarx, Kaspersky, and Snyk have all uncovered a huge number of malicious packages in PyPI that could be abused to conduct DDoS attacks and harvest browser passwords as well as Discord and Roblox credential and payment information.

So now RubyGems joins the ranks of NPM and PyPI which are all tightening up their security.

Closing The Loop

Thomas Tomchak @tomchak

Hey Steve. When you register a domain you do have the option to register a technical contact as well as the owner. When I have registered domains in the past for friends, I always make sure they are listed as the domain owner, and myself only as the technical contact. Yes, it's still under my registrar account, but at least that shows proof of ownership and they could probably transfer it to a new account at the same/different registrar should I get hit by a bus unexpectedly.

I'm so glad that Thomas thought to remind me and us of that. (And I acknowledge that several other listeners sent notes to the same effect.) Domain registration records provide for completely separate "Owner", "Administrative" and "Technical" contacts. I'm so used to always pointing all three of those at myself that I completely forgot the power of the flexibility that they could provide. This of course begs the question: What would a domain registrar do if the assigned owner of a domain — which is, after all, just a name and eMail address — wish to take control of the domain in the event that the Admin or Technical contact was unresponsive? Would that provide the degree of safety we're looking for?

In an attempt to answer that question definitively, I found an ICANN FAQ:
https://www.icann.org/resources/pages/faqs-f0-2012-02-25-en
Question #12 is: *"I can't "access" my domain name or my domain name management account because the domain name was registered by someone else, such as my web developer/administrative contact - what now?"*

And their answer is: *"You may not be able to access the domain name if you are not the administrative contact/registrant of record of the domain name. You should contact the individual or entity who registered the domain name to obtain access credentials/details or update the domain name's administrative contact/registrant of record.*

You should contact your registrar right away if your domain name manager/administrative contact is unreachable, has gone out of business, etc. to update your information. Once you are able to become the administrative contact/registrant of record, this will ensure that you have full control of managing your domain name and allow you to find someone else to help you manage your domain name, if you so choose. It's a good idea to keep a record of your domain name management credentials at all times, even if you choose to outsource some administrative/ management duties to a third party."

This still leaves "Owner" and "Administrative" in competition and seems to be a bit nebulous.

dagan @TechDagan *(pronounced Day' Ghen)*
I am truly flattered, honored and humbled that this podcast has had as much impact on people's lives through the years as it has. I suppose that my true love for technology and computing can be a bit infectious. So in that sense I'm gratified to have had the opportunity to infect so many of our listeners with this bug that's had me in its grip throughout my entire life. This Tweet really hit home, so I asked its author, dagan, if I could share it and I received his permission:

Steve, I wanted to privately drop you a brief personal note of thanks. I have been listening to Security Now religiously since 2011, and I truly believe your weekly show has had a significant impact on my career and, as such, my life. When I started listening, I was a happy nerd that loved the idea of security. 9 years in, I discovered and was credited for identifying 2 CVEs in a Red Hat product. Last year, I was credited for two more CVEs in a Suse product, and this year yet another vuln in a second Suse product. And finally, just last weekend, I was fortunate enough to have the opportunity to present at DEF CON, where I demonstrated chaining 3 of the 5 vulns together to fully compromise a Kubernetes cluster from the outside. I will also be speaking at KubeCon this year, where I will talk about securing clusters from risks introduced by third-party applications. I'd happily buy you a nice bottle of wine to express my most sincere gratitude, but I'll settle for a SpinRite license instead. Your show has changed my life, Steve. Thanks.

Wow. Like I said... it has been and is truly my privilege. And as I told Leo last week, doing this podcast with him is one of the best things I've ever done.

Then just this morning, Leo forwarded a very heart-warming story from someone who said his career was catalyzed by this podcast. As we start into year 18 I want to share what he wrote as an example of what is possible if anyone else out there might be in need of a bit of a nudge. This person wrote:

Leo,

I must say I'm quite honored to have received a reply directly from you! I would like to share a brief story you and Steve might like to hear. In 2015 I recognized Steve's name on a list of podcasts. I knew it from back in the mid/late '90s when I used shields-up frequently. I left IT after the dot com bubble around 2002-ish. I followed a path in science, but ultimately I couldn't find a good job in that field. I was in poverty, on Medicaid, and in significant debt. It wasn't just me — I met my wife and our son was just born during this time. It was rough. I listened to Security Now and the ad for IT Pro TV. I signed up, studied, and passed the CCNA.

In a matter of days I had a good paying job as a network engineer. I have since moved on into cybersecurity, getting a CISSP. I now work at a job I love, and my family is able to live debt-free with good healthcare and everything else that goes with this terrific career.

I have you and Steve to thank for this. Your show made this field approachable and fun. Words cannot begin to express my appreciation and gratitude. You guys change lives. Please keep up the great work!

Michael

Thank you, Michael, for sharing your story with us. The field of cybersecurity truly is a growth industry. There is a crying need for more trained cybersecurity professionals. And boy is it interesting!!

The Bumblebee Loader

https://www.cybereason.com/blog/threat-analysis-report-bumblebee-loader-the-high-road-to-enterprise-domain-control

The Bumblebee Loader has recently become a big deal on the malware front.

At the end of June, Symantec wrote: *"Bumblebee, a recently developed malware loader, has quickly become a key component in a wide range of cyber-crime attacks and appears to have replaced a number of older loaders, which suggests that it is the work of established actors and that the transition to Bumblebee was pre-planned. By analysis of three other tools used in recent attacks involving Bumblebee, Symantec's Threat Hunter team, has linked this tool to a number of ransomware operations including Conti, Quantum, and Mountlocker. The tactics, techniques, and procedures (TTPs) used in these older attacks support the hypothesis that Bumblebee may have been introduced as a replacement loader for Trickbot and BazarLoader, since there is some overlap between recent activity involving Bumblebee and older attacks linked to these loaders."*

Earlier, on June 7th, "Cyble", the security firm we were just talking about last week, wrote: *"In March 2022, a new malware named "Bumblebee" was discovered and reportedly distributed via spam campaigns. Researchers identified that Bumblebee is a replacement for BazarLoader malware, which has delivered Conti Ransomware in the past. Bumblebee acts as a downloader and delivers known attack frameworks and open-source tools such as Cobalt Strike, Shellcode, Sliver, Meterpreter, etc. It also downloads other types of malware such as ransomware, trojans, etc."*

And last Wednesday, the global security firm CyberReason, based in Boston Massachusetts with offices in London, Tel Aviv, Tokyo, France, Germany, South Africa and Singapore, published in their "Malicious Life" Blog a detailed technical description of the operation of this extremely dangerous new entry onto the malware scene. They titled their report: *"Bumblebee Loader - The High Road to Enterprise Domain Control."*

This podcast would be remiss if we didn't take some time to bring everyone up to speed about this emergent threat.

CyberReason's report explains that they analyzed a case that involved a Bumblebee Loader infection which allows them to describe in detail the attack chain from the initial Bumblebee infection to the compromise of the entire enterprise network.

Let's begin with a couple of bullet points to set the stage:

- The majority of the infections with Bumblebee we have observed started by end-users executing LNK files which use a system binary to load the malware. Distribution of the malware is done by phishing emails with an attachment or a link to the malicious archive containing Bumblebee.

- Bumblebee operators conduct intensive reconnaissance activities and redirect the output of executed commands to files for exfiltration.

- The attackers compromised Active Directory and leveraged confidential data such as users' logins and passwords for lateral movement. The time it took between initial access and Active Directory compromise was less than two days. (I share a timeline breakdown in a minute.)

- Cybereason GSOC (that's their Global Security Operations Center) has observed threat actors transitioning from BazarLoader, Trickbot, and IcedID to Bumblebee, which seems to be in active development and generally the loader of choice for many threat actors.

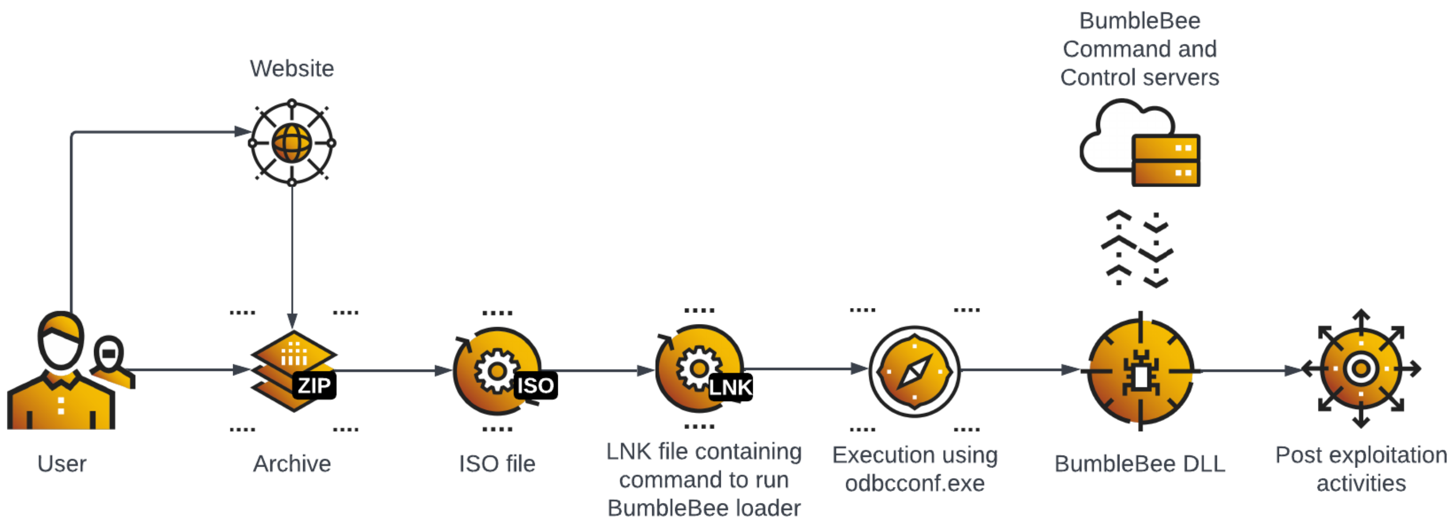
- Attacks involving Bumblebee must be treated as critical. Based on GSOC findings, the next step for the threat actors is ransomware deployment, and this loader is known for ransomware delivery.

So, let's take this step-by-step...

- A spear phishing email is received containing an archive or a link to an external source to download the archive. As we know, the malware is encapsulated in an archive to prevent the archive's contents from being tagged with the Mark of the Web (MOTW) which would complicate its execution.

- The user extracts the archive and mounts the resulting ISO image. Newer releases of Windows will happily mount .ISO images, thus exposing the ISO's file system files.

- The content of the mounted ISO image is a .LNK file executing the Bumblebee payload upon user interaction:


Website
BumbleBee Command and Control servers
ZIP
ISO
LNK
User
Archive
ISO file
LNK file containing command to run BumbleBee loader
Execution using odbcconf.exe
BumbleBee DLL
Post exploitation activities

So, the operators behind an instance of Bumblebee host malicious websites that implement a drive-by download. To infect the system, an end-user has to first manually decompress the archive containing the ISO file, mount the file and then execute the Windows shortcut (LNK). This is all done as part of a phishing eMail where the user fully believes that they are doing the right thing, installing this or that that's needed, or updating something that's needed before they can proceed. So the user is without question unwittingly complicit in the success of this penetration and intrusion. All of the other mechanics is about avoiding everything the user's enterprise security people have done to keep bad stuff out, despite what dumb stuff their users might do.

So... the LNK file has an embedded command to load and execute the Bumblebee Dynamic-link library (.DLL) using the already and always present odbcconf.exe in what has become the increasingly popular "Living Off the Land" approach of using what's already available (and plenty is) on modern systems. So in this context, odbcconf.exe is called a LOLBin. A response (.rsp) file is also used where the [Bumblebee specific name].rsp has the reference to the Bumblebee DLL. Specifically, the LNK file's "Target" property contains the string "odbcconf.exe -f [Bumblebee specific name].rsp" and the .rsp file contains a reference to [Bumblebee specific name].dll which is the Bumblebee payload.

If anyone's curious, you can see this for yourself. In any version of Windows, open a command prompt and type "odbcconf /?" and you'll receive a pop-up from ODBCCONF showing a list of its command-line options. Among them is /F which takes a response file – basically a command stream – as its argument.

In this case, this loads and runs the Bumblebee .DLL at which point all is lost because a hostile executable has made it into the user's system and has been started.

The Bumblebee DLL injects code into multiple running processes in order to establish a strong foothold on infected endpoints and the newly launched odbcconf.exe process creates Windows Management Instrumentation calls to spawn two new processes from the wmiprivse.exe (Windows Management Instrumentation Provider Service). Once again, both of these newly spawned processes are existing Windows executables where malicious code is dynamically injected into them once they have been started:

wabmig.exe is the Microsoft Contacts import tool. It is injected with Meterpreter agent code. Meterpreter is a Metasploit attack payload that provides an interactive shell from which an attacker can explore the target machine and execute code. It's deployed using in-memory DLL injection. As a result, Meterpreter resides entirely in memory and writes nothing to disk.

wab.exe is the Microsoft address book app. After being launched, it receives an injection of the Cobalt Strike beacon.

Bumblebee performs privilege escalation by loading an exploit for CVE-2020-1472 (Zerologon) into rundll32.exe. Bumblebee uses a User Account Control (UAC) bypass technique to deploy post exploitation tools with elevated privileges on infected machines. Specifically, it uses an existing trusted binary **fodhelper.exe**. This prevents Windows from showing a UAC window when it's launched. **fodhelper.exe** is the executable used by Windows to manage features in Windows settings. Again, living off the land.

So, **fodhelper.exe** is exploited to run *"cmd.exe" /c rundll32.exe C:\ProgramData\Cisco\[Cobalt strike].dll", MainProc* where [Cobalt strike].dll is a Cobalt Strike framework beacon and MainProc is the exported function to run. As we know, Cobalt Strike is an adversary simulation framework used to assist in red team attack operations. Bad guys use it to conduct actual post-intrusion malicious activities. Unfortunately, it's a powerful modular framework with an extensive set of features that are useful to malicious actors, such as command execution, process injection, and credential theft. And speaking of credential theft...

After obtaining its foothold and elevating itself to system privilege without any further user interaction or UAC permission, Bumblebee performs credential theft using two methods:

The first method is to trigger a memory dump of Windows' LSASS process. LSASS is Windows Local Security Authority Subsystem Service. Within its memory footprint are the keys to the kingdom including both domain and local usernames and passwords. They are all sitting in the memory space of the LSASS process. So Bumblebee dumps the memory of this process using **procdump64.exe** to obtain access to the sensitive information.

The second method of credential theft used by Bumblebee is registry hive extraction using **reg.exe**:

- HKLM SAM, with SAM being the Security Account Manager database is where Windows stores information about user accounts.
- HKLM Security: Local Security Authority (LSA) stores user logins and their LSA secrets.
- HKLM System: Contains keys that could be used to decrypt/encrypt the LSA secret and SAM database

Bumblebee issues the three commands of the form:

"reg.exe save hklm\sam c:\ProgramData\sam.save" and then the same for **\system** and **\security**. That creates a trio of files containing the dump of those three system-critical registry hives. Then the LSASS dump and those three registry hives are compressed using 7z and exfiltrated.

At that point the human operators behind Bumblebee process the retrieved credentials offline, attempting to extract cleartext passwords. The observed time between credential theft and the next activity is approximately 3 hours.

After the attackers have gained a foothold within the organization's network, they gather information using tools such as nltest, ping, netview, tasklist and ADFind to collect a wide range of information related to the organization. They collect information such as the domain names, users, hosts and domain controllers. We talked about ADFind in episodes 789 and '90 back in October of 2020. It's a powerful Active Directory exploration tool meant to aid in the administration of Active Directory systems. Unfortunately, it's been turned against those administrators.

Bumblebee uses Cobalt Strike agent for lateral movement. Their analysis observed multiple connections from the Cobalt Strike process to internal addresses on RDP — Remote Desktop Protocol on TCP port 3389. After lateral movement, the attacker persists on the organization network using the commercial remote management software "AnyDesk."

After the attacker has obtained a highly privileged username and password, they access the Volume Shadow Service Shadow Copy, which is Windows' built-in facility to create backup copies and snapshots of computer files or volumes while they are in use. So Bumblebee accesses the remote Active Directory machines using the Windows Management Instrumentation command-line utility (WMIC) and creates a shadow copy using the vssadmin command. In addition, the attacker steals the **ntds.dit** file from the domain controller. **ntds.dit** is a database that stores Active Directory data, including information about user objects, groups and group membership. And the file also stores the password hashes for all users in the domain.

In order to obtain maximum privileges on the Active Directory domain, the threat actor executes the following steps:

- Creates a shadow copy of the machine file's volume
- Lists all available shadow copies, storing the result in a file.
- Copies the Active Directory database (ntds.dit) as well as registry hives containing credentials and sensitive data from the shadow copy.
- Compress the output directory for exfiltration.

And, finally, the threat actor uses a domain administrator account obtained previously to move laterally on multiple systems. After initial connection, they create a local user and exfiltrate data using the open source RCLONE software. Wikipedia describes Rclone:

Rclone is an open source, multi threaded, command line computer program to manage or migrate content on cloud and other high latency storage. Its capabilities include sync, transfer, crypt, cache, union, compress and mount. The rclone website lists supported backends including S3, and Google Drive.

In the instance observed and monitored by Cybereason, the rclone.exe process transferred approximately 50 GB of data to an endpoint with an IP address over TCP port 22 (SSH), located in the United States.

So what does all this tell us?

The first and most obvious thing we've learned is that you do NOT want to have your enterprise stung by The Bumblebee. It'll definitely ruin your whole day. But speaking of "day", Cybereason compiled the entire event into a timeline.

Taking everything we've just stepped through, here's how it stretches out in time...

Activities	Time
Initial access	T0
Reconnaissance / nltest, net, whoami	T0 + 30 minutes
Command and Control / Loading Meterpreter agent	T0 + 4 hours
Privilege Escalation / Zerologon exploitation	T0 + 4 hours
Command and Control / Cobalt Strike beacon execution	T0 + 6 hours
Credential Theft / registry hive	T0 + 6 hours
Reconnaissance / adfind, ping, curl	T0 + 6 hours and 30 minutes
Credential Theft and Privilege Escalation / LSASS memory dump with procdump64.exe	T0 + 19 hours
Credential Theft / NTDS.dit exfiltration with Active Directory full privilege	T0 + 22 hours
Lateral Movement / Cobalt Strike socks-tunnel (RDP)	T0 + 24 hours
Data Exfiltration / Rclone	T0 + 3 days

One lesson we learn is why **local privilege escalation** vulnerabilities form such a crucial part of this attack chain. Remote Code Execution seems like the worst possible nightmare. And indeed, it's not good. But none of that was needed here. If malware were truly constrained within a user's low-privilege account, much less damage could be done. But the old saying "If wishes were horses, beggars would ride" reminds us that wishing won't make it so. With everything we've seen of the continuing and apparently worsening trouble Microsoft is having securing Windows, which they refuse to just leave as is and instead fix — because it's obvious they can't do both — there is virtually zero chance that once a single piece of malicious software gets loose inside of a user's machine that the rest of the organization will not fall.

And this brings us back to that first fatal click. That innocent action taken by a user on the inside, that initiated the collapse of even the most carefully constructed enterprise security. Those in charge of their organization's security must be living in a state of constant terror over what any one of their employees might do next.