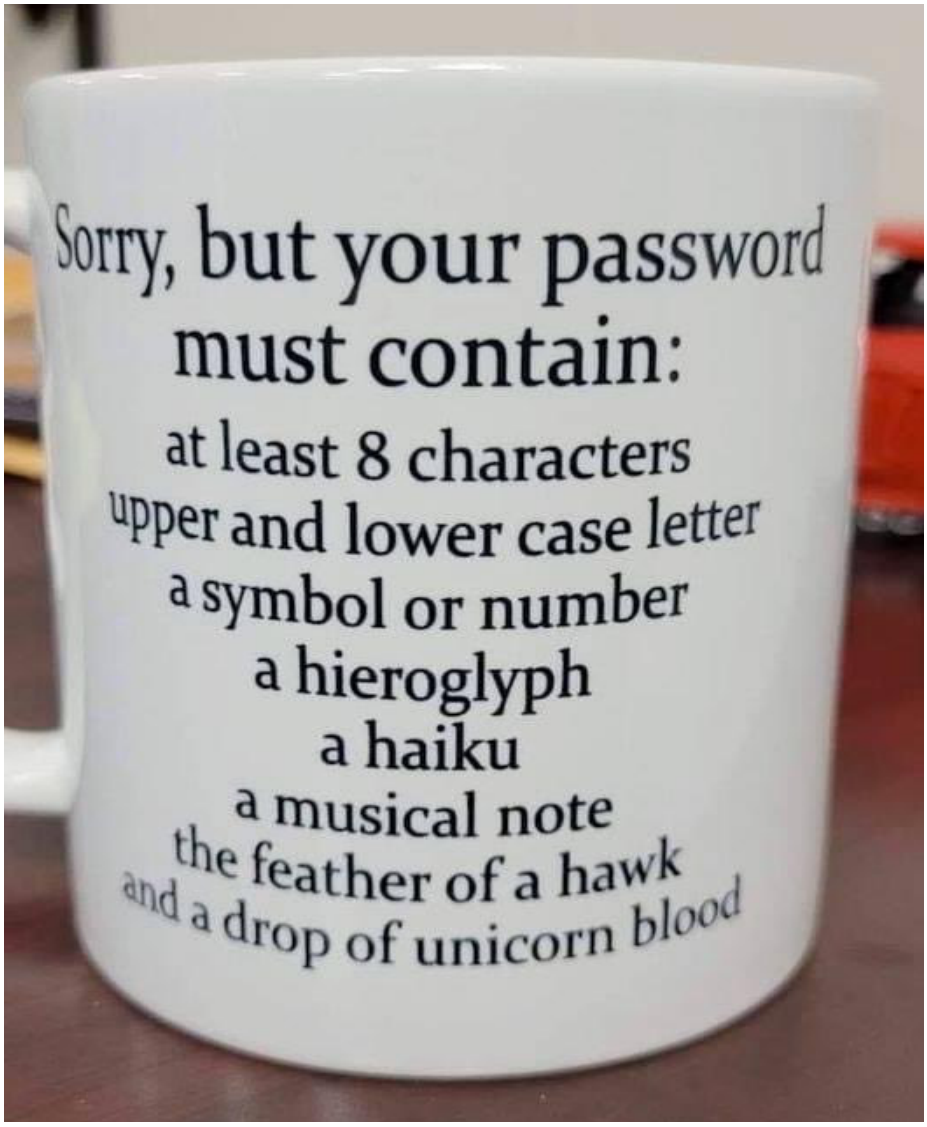
Security Now! #836 - 09-14-21

The Mēris Botnet

This week on Security Now!

This week we're going to note the apparent return of REvil—not nearly as dead and gone as many had hoped. We're going to look at a new and quite worrisome 0-day exploitation of an old Windows IE MHTML component. Even though IE is gone, it's guts live on in Windows. We're going to share the not surprising but still interesting results of a pair of security impact surveys taken of IT professionals and the home workers they're trying to protect. After that, we examine a fully practical JavaScript based Spectre attack on Chrome. Then I have a bit of closing the loop feedback to share, and a surprisingly serious question about the true nature of reality for us to consider. Finally, we'll finish out today's podcast by looking back at the evolution of Internet DoS attacks through the years which have recently culminated in the largest ever seen, most problematic to block and contain, RPS DDoS attack where RPS stands for Requests Per Second.

What do you mean you've been unable to create an account there? What's the problem?



Security News

A new worrisome 0-day attack against Office documents.
Microsoft is warning of a newly discovered IE 0-day being exploited in targeted Office attacks.

While the danger might not be extreme, especially if the use of this exploit remains targeted, This should remind us of our picture of the week two weeks ago titled "Pandora's Inbox" where Pandora is depicted thinking to herself... "It can't hurt to open one little attachment, can it?" And while I agree that it's unlikely to hurt, we know that once a 0-day that has only been observed being used in highly targeted attacks is discovered and publicized, its secret is out and a patch will be forthcoming. Which means that the optimal strategy for those who wish to exploit their time-limited advantage becomes "spray it far and wide to collect all of the curious and incautious Pandoras possible." Don't be a Pandora.

When we hear that it's an IE 0-day, that's really a misnomer because the vulnerability, now being tracked as CVE-2021-40444, was found in Microsoft's MHTML component, also known as Trident, which is the Internet Explorer browser engine. And while IE itself is no longer showing up to the party, that component remains in active use by Office applications to render any web-hosted content inside Word, Excel and PowerPoint documents.

Microsoft has said that they are aware of targeted attacks exploiting this vulnerability and that "An attacker could craft a malicious ActiveX control to be used by a Microsoft Office document that hosts the browser rendering engine." They explained that the attacks and the underlying 0-day were discovered by security researchers from Mandiant and EXPMON.

Last Tuesday, EXPMON tweeted: *"EXPMON system detected a highly sophisticated #ZERO-DAY ATTACK in the wild (ITW) targeting #Microsoft #Office users! At this moment, since there's no patch, we strongly recommend that Office users be extremely cautious about Office files – DO NOT OPEN if not fully trust the source!"*

And since Microsoft's disclosure the hacking attacking community has been on overdrive. Posts, guides, demo code, working proofs of concept and everything anyone would need to design and launch their own attacks, have been freely available since late last week. Even Github has been hosting complete exploit documentation. The good news is, Windows Defender's knowledge base has been updated to recognize known instances of the attacks. So some protection is presently available to everyone using some form of Defender.

There are guides to implementing various mitigations, but attackers keep finding ways around those mitigations. Today is Patch Tuesday so maybe this one is being fixed as we speak? But, again, now that the cat's out of the bag and a fix will almost certainly be forthcoming, the optimal exploitation strategy may be changing from a sniper rifle to a shotgun. So even if you're not being targeted you'll still want to keep your head down.

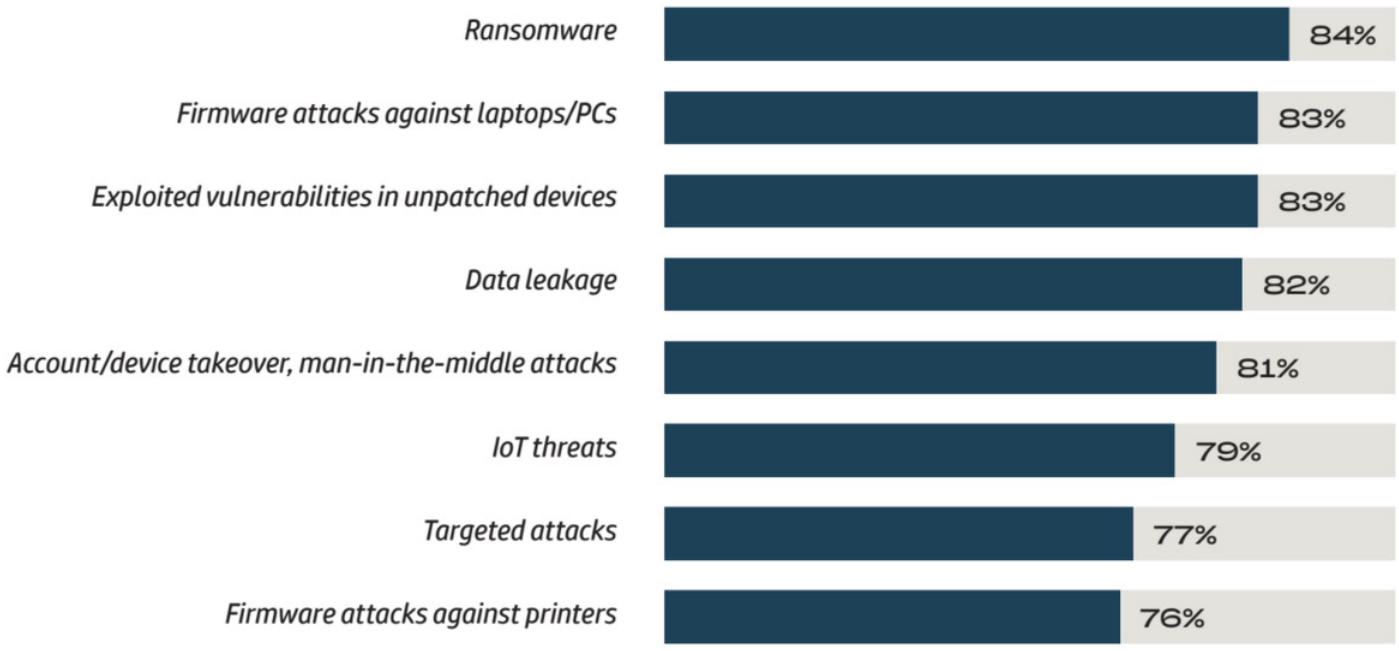
Work From Home (WFH) — No problem?
Last Thursday, Hp's Wolf Security group published a new study which they titled the "Security Rebellions & Rejections report." It is a compilation of data from an online YouGov survey aimed at office workers who adopted Work From Home, and other global research conducted with IT

decision-makers. Thus, the report presents a view from IT professionals who are responsible for keeping their company secure amid the pandemic-fueled rush to shelter in place and work from home.

In total, 91% of those surveyed said that they have felt "pressured" to compromise on security due to the need for business continuity during the COVID-19 pandemic.

76% of respondents said that security had taken a backseat, with 83% believing that working from home has created a "ticking time bomb" for corporate security incidents. The following chart shows the relative level of threat IT teams feel as a result of their office working employees increasingly working from home:

Figure 3– Level of threat IT teams believe the following attack methods pose with people increasingly working from home on potentially insecure networks



And the view from the suddenly-remoted office worker substantiates the worries that those who are responsible for security are evidencing:

Specifically, according to the survey, younger office workers, in particular, are more likely to circumvent existing security controls and safeguards in order to manage their workloads, with 48% of this group saying that security tools, such as website restrictions or VPN requirements, are a hindrance -- with at least 31% of them having at least attempted to bypass them.

Overall, 48% of office workers said that security measures waste time and 54% in the 18-24-year-old bracket were more concerned with meeting deadlines than potential security breaches. And within that same group, 39% stated that they were unsure or unaware of their employer's security policies.

Three other bullet points which the report highlighted, were:

- 37% of office workers believe security policies are often too restrictive
- 80% of IT teams experienced backlash from home users because of security policies
- 83% of IT teams said the blurred lines between home and work life made enforcement "impossible."

Joanna Burkey, HP's Chief Information Security Officer stated that: "CISOs are dealing with increasing volume, velocity, and severity of attacks. Their teams are having to work around the clock to keep the business safe while facilitating mass digital transformation with reduced visibility. Cybersecurity teams should no longer be burdened with the weight of securing the business solely on their shoulders; cybersecurity is an end-to-end discipline in which everyone needs to engage."

Right. But tell that to someone who's having trouble authenticating to their remote employer's VPN and who has NO appreciation for the dangers that are lurking.

Browser News

"Attacks only ever get better"

As we know, the beginning of 2018 was dominated by the concept of Spectre attacks. Spectre, which was assigned CVEs 2017-5715 and 53, refers to a class of CPU performance optimizations which turned out to create previously unsuspected hardware-based vulnerabilities in many modern CPUs that break the isolation separating applications to — theoretically — permit attackers to trick a program into accessing arbitrary locations associated with its memory space, thus abusing it to read the content of accessed memory, and to thereby potentially obtain sensitive data.

Well, guess what. Practical attacks are no longer theoretical. Academic researchers at the University of Michigan, University of Adelaide, Georgia Institute of Technology, and Tel Aviv University have designed a Spectre-based side-channel attack which can be weaponized to successfully overcome the Site Isolation protections Google added to Chrome and to the Chromium browsers to leak sensitive data in Spectre-style speculative execution attacks.

They call it "Spook.js" and, as its name suggests, it's a JavaScript-based attack. The researchers said: "An attacker-controlled webpage can know which other pages from the same websites a user is currently browsing, retrieve sensitive information from these pages, and even recover the user's username and password login credentials when they're autofilled." they added that an attacker could retrieve data from Chrome extensions as well. Any data stored in the memory of a website being rendered, or a Chrome extension, can be extracted, including personally identifiable information displayed on the website, and auto-filled usernames, passwords, and credit card numbers.

Responding to this, Google said: "These attacks use the speculative execution features of most CPUs to access parts of memory that should be off-limits to a piece of code, and then use timing attacks to discover the values stored in that memory. Effectively, this means that untrustworthy code may be able to read any memory in its process's address space."

Chrome's "Site Isolation" rolled out in July 2018, being fully enabled since Chrome 67. Site Isolation were a series of software countermeasures designed to make these processor-based side-channel attacks more difficult to exploit — things like reducing timer granularity. When Site Isolation is enabled Chrome 67 and beyond will load each website into its own process. This would thwart attacks between OS isolated processes, and thus, between sites.

But the researchers found scenarios where the site isolation safeguards do not separate websites and they were able to get around the anti-Spectre protections. Spook.js exploits this design quirk to result in information leakage from Chrome and Chromium-based browsers running on Intel, AMD, and Apple M1 processors.

So I dug into this a bit, and upon closer inspection it turns out that what Google did was to only isolate at the second level domain level. For example, Chrome will separate 'example.com' and 'example.net' due to different top-level domains, and Chrome will also separate 'example.com' and 'attacker.com—those two having differing second-level domains. But `attacker.example.com' and 'corporate.example.com' are still allowed to share the same process, and it's this lack of absolute process isolation by domain that allows pages hosted under 'attacker.example.com' the opportunity to extract information from pages under 'corporate.example.com.'"

So this was a very useful discovery since the researchers were able to demonstrate, through practical JavaScript-based attacks, that the existing countermeasures are insufficient to protect users from browser-based speculative execution attacks.

The Chrome Security Team's immediate response to this (they received early access to the research last July) was to extend Chrome's existing Site Isolation to ensure that browser extensions would no longer share a common process with each other. That will curtail the cross-extension leakage that the researchers demonstrated. This feature is controlled by a setting called "Strict Extension Isolation" which is enabled as of Chrome versions 92... and why would anyone **not** want that to be turned on?

At this point it's unclear whether Google will take further action. They might feel that isolating at the second-level domain is sufficient. So the researchers explained that as an immediate workaround: "Web developers can immediately separate untrusted, user-supplied JavaScript code from all other content for their website, hosting all user-supplied JavaScript code at a domain that has a different eTLD+1. This way, Strict Site Isolation will not consolidate attacker-supplied code with potentially sensitive data into the same process, and that will place that data out of reach even for Spook.js since it's unable to reach across process boundaries."

Ransomware News

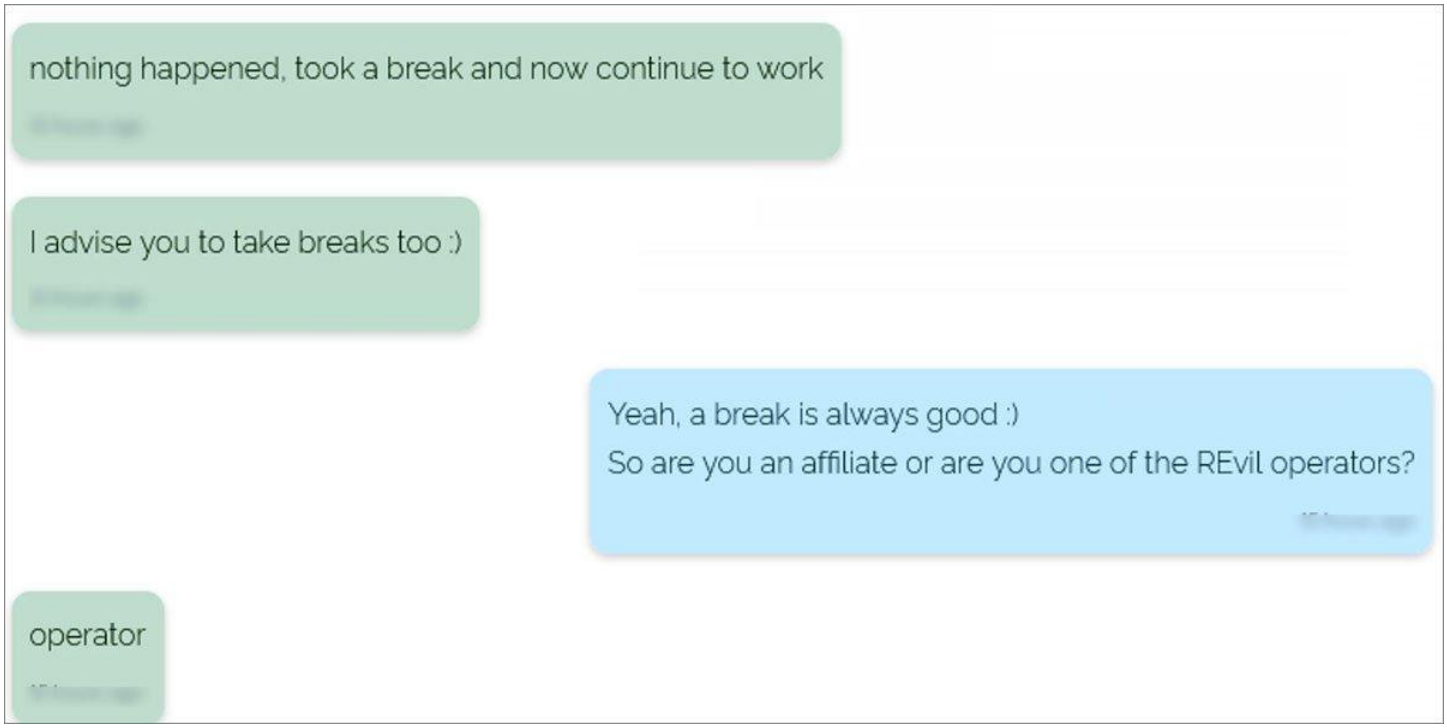
The return of REvil — Apparently, vacation's over.
EMSIsoft may have been the first to observe that REvil's "Happy Blog" data leaking site on the dark web had returned last week. At that stage, their ransomware negotiation site had not yet returned. But it has now. Bleeping computer arranged to get some Russian postings translated into English. Remember that REvil's primary operator is known as UNKN, "unknown"...

"As Unknown (aka 8800) disappeared, we (the coders) backed up and turned off all the servers. Thought that he was arrested. We tried to search, but to no avail. We waited - he did not show up and we restored everything from backups.

After UNKWN disappeared, the hoster informed us that the Clearnet servers were compromised and they deleted them at once. We shut down the main server with the keys right afterward.

Kaseya decryptor, which was allegedly leaked by the law enforcement, in fact, was leaked by one of our operators during the generation of the decryptor." - REvil

In a dialog apparently with a REvil operator that Bleeping Computer was able to obtain:



If we're to believe them, the REvil operation just decided to take a break. I suspect that there's likely much more to it, but I'm pretty sure we'll never know. What is worth noting, however, and the reason I've mentioned this news, is that, for what it's worth, the distressingly successful ransomware gang behind the attacks on JBS, Coop, Travelex, GSMLaw, Kenneth Cole, Grupo Fleury, and of course the worst by far which leveraged the Kaseya vulnerabilities, is back in operation. Only time will tell what that means.

Closing the Loop

David Cortesi / @davecortesi

@SGgrc "Zen of Assembly Language" is hard to find on the used market but it can be read free online at https://github.com/jagregory/abrash-zen-of-asm

This is the source for an ebook version of Michael Abrash's Zen of Assembly Language: Volume I, Knowledge, originally published in 1990 and.

Reproduced with the blessing of Michael Abrash, converted and maintained by James Gregory.

The Github releases list, has Epub and Mobi versions available for download, and you can find a mirror of the HTML version at www.jagregory.com/abrash-zen-of-asm.

That James Gregory page is the entire book and all code listing examples on one very long HTML page. But it's the quickest way to get a sense for this topic, presented by Michael Abrash.

MrJohnDoe / @MrJohnDoe9
@SGgrc my Verizon G3100 router recently updated and now has both a "Guest" network, and a separate "IoT" network. I've heard you mention this type of network separation several times since I started listening to Security Now. Thanks!

[Verizon FiOS G3100 Wireless router 4-port switch GigE 802.11ax MoCA 2.5]

Sci-Fi

I have this next piece under "Science Fiction" — but is it fiction???
The other day I googled "are we living in" — that's all I typed: "are we living in" and I was frankly quite surprised when the suggested phrase completion offered what I was planning to type, which was: "are we living in a computer simulation."

As much as we all know I'm a science fiction enthusiast, it turns out that this is actually a question. Back in 2003, an Oxford University philosopher named Nick Bostrom published, in the Philosophical Quarterly, his paper titled "Are You Living in a Computer Simulation?" In that paper, Bostrom establishes a serious philosophical framework for considering the question. And while Elon Musk made news when he brought up his support for what's become known as "the simulation hypothesis", the well known astrophysicist Neil deGrasse Tyson had the same question posed to him by NBC News, with Tyson giving it a "better than 50-50 odds" that the simulation hypothesis is correct. Tyson said: "I wish I could summon a strong argument against it, but I can find none." And by the end of his carefully constructed and considered paper, which I won't go into here but I have a link to it in the show notes, Nick Bostrom concludes that if computer-loving aliens exist, meaning, if there are intelligent entities having the power to create simulations, then Bostrum argued, "we are almost certainly living in a computer simulation."

I don't want to spend too much more time on this, but I did want to note that the question is truly being taken very seriously. On his podcast StarTalk, when discussing this question, which was the topic of the entire podcast, Neil deGrasse Tyson observed that such a simulation would most likely create perceptions of reality on demand rather than simulate all of reality all the time, in the same way that a video game is optimized to render only the parts of a scene visible to a player.

Last year, in the "Space & Physics" section of "Scientific American" dated October 13, 2020, an article posed the question "Do We Live in a Simulation? Chances Are about 50–50":
https://www.scientificamerican.com/article/do-we-live-in-a-simulation-chances-are-about-50-50/

And Wikipedia has a page titled:
https://en.wikipedia.org/wiki/Simulation_hypothesis

For any of our listeners who are curious, I have a link to Nick Bostrom's paper in the show notes.
https://www.simulation-argument.com/simulation.pdf

And his paper is being hosted by a very intriguing website:
https://www.simulation-argument.com/

Which hosts all of the repercussions—on both sides of the argument—which have been catalyzed by Nick's paper. If you think that the whole thing's just a big joke, you might want to spend a little time over at https://www.simulation-argument.com where you will find serious people giving this question some serious consideration.

Now, what got me onto this whole thing? It was the recently released trailer for the fourth movie in "The Matrix" series. One of the Wachowski's (Lana) is bringing back Neo and Trinity for another special effects extravaganza opening on Wednesday, December 22nd... and I can't wait! That'll be the day after our final podcast of 2021 and the trailer looks like it's going to be a lot of fun.

And who knows? We all know how often science fiction predicts reality. So perhaps this is actually recursive? We are truly living in a Matrix-like pseudo-reality and we're making movies about Matrix-like pseudo-realities!

But I am sure of one thing that is real, Leo, at least in our present reality, which is the sponsor you're about to tell us about...

The Mēris Botnet

(Mēris is Latvian for 'plague')

We haven't talked about DDoS attacks for a while. But the recent series of headline grabbing mega-attacks on the Russian multinational, Yandex, have recently been breaking all records to culminate in the largest single attack ever seen. And since the attacks originated from a new and frighteningly large Botnet, which is apparently hosted by unpatched and compromised routers and which could be aimed at any target on the Internet, I thought we ought to check-in this week and catch up on what's going on in Big Bot-land.

Yandex is large enough to probably be at least somewhat familiar to this podcast's followers; though perhaps more so outside the US than in. It's one of the largest Internet companies in Europe, operating Russia's most popular web search engine and Yandex is Russia's most visited website. It's revenue last year was around $3 billion USD. Or, since Russian Rubles, currently being worth about 1.4 cents are fun to convert to, that would be annual revenue of 214 billion rubles.

Anyway, it seems that apparently somebody has a grudge against Yandex, or perhaps they're just wanting to flex their new mega Botnet's muscles to see what it can do. When you think about it, I'm sure that someone running a huge Botnet wants to know just how huge it is. But it's not really possible to know how powerful a Botnet is unless it's turned loose against someone. And since the typical result of a large and powerful Botnet being unleashed is the immediate meltdown of its target, it's necessary to choose a sufficiently large and/or well protected target so that some of the attack's metrics can be obtained and reported.

But before we dig into the details, let's review a bit about the history and the various forms of denial of service (DoS) and distributed denial of service (DDoS) attacks:

Few should first note that the term "denial of service" is itself extremely broad and generic. For example, when we've been talking about newly discovered vulnerabilities which have not yet been weaponized into full attacks, the easier sort of "junior attack" is to just crash the service that receives a malformed packet. And that's referred to in the industry as a "denial of service" attack because a crashed service will, indeed, be denying its clients its service. So I just wanted to give that other form DoS attack some air, and then put it aside, since that's not the form of DoS attacks we're going to be talking about.

DoS and DDoS attacks have evolved tremendously through the years. In every case we'll see that attacks can always be reduced to the consumption or overloading of some resource. Before we get specific, we should briefly refresh our memory of the Open Systems Interconnection (OSI) model which describes and standardizes the roles of seven layers that computer systems use to communicate over a network. The OSI's meaning of "layer" will become clear when the layers are described. The model was the first standard model for network communications, and it was adopted by all major computer and telecommunication companies in the early 1980s who were hungry to adopt something, anything. There were no standards at the time and we had chaos.

As it turned out, the Internet is only loosely based on the OSI model because it had no need for two of the upper layers, 5 and 6. But the others are all well identifiable in today's networking...

- Layer 1 is the Physical Layer. It literally describes and specifies the voltages, currents, and bit rates that will the used to form the bits for every layer above.

- Layer 2 is the Data Link Layer. For most networks that means Ethernet protocol. This is where 48-bit MAC addresses appear.

- Layer 3 is the Network Layer. This is where the IP protocol resides. ARP, the Address Resolution Protocol is used to associate layer 3 IP addresses to the underlying layer 2 MAC addresses so that Ethernet switches and routers are able to route Ethernet packets by IP.

- Layer 4 is the Transport Layer. This is where TCP, UDP and ICMP live. Just as layer 3 is the payload for layer 2, layer 4 is the payload for layer 3. They're like nested envelopes.

- Layers 5 and 6, the OSI Session and Presentation layers that had no role in the Internet.

- Which brings us to layer 7, the Application layer. This is where protocols that run over TCP or UDP are found. So, HTTP, FTP. SMTP, DNS, RDP... and all the rest. All of those layer 7 protocols are the payload carried by UDP or TCP on layer 4.

The first attacks on Internet servers were low-bandwidth "SYN Trickle" attacks which could be generated by a single malicious client on the Internet by simply sending TCP SYN packets to a remote server. This would cause early TCP/IP protocol stacks to rapidly exhaust their pending-connection resource pools. In this way, a server offering any form of TCP connection-accepting service would become unable to accept legitimate connections once all of its local connection-accepting resources had been tied up by a client that never had any intention of honoring its initial TCP connection requests. The OSI model would place that attack at layer 4 since it was specifically an attack against the server's implementation of its TCP protocol.

So, if that was the SYN Trickle attack, the next thing that happened was the SYN Flood:

As we know, the Internet is essentially a data communications fabric knit together by routers. Packets come in, their IP addressing headers are examined, and they are routed toward their destination by referencing a routing table. One of the parameters of routers is their maximum packet handling rate. Packets on the Internet are variable length, but since packets have a fixed overhead for addressing, maximum overall efficiency will be obtained using longer packets to yield the greatest payload to header ratio. Since it's unusual for routers to encounter a high percentage of small packets, the performance of a router's routing switch fabric will be scaled to handle the average packet rate that might be presented to it through all of its incoming interfaces. And this explains the success of the SYN flood attacks:

TCP SYN packets are among the tiniest packets possible. Whereas a typical Internet packet will be 1514 bytes, TCP SYN packets weigh in at just 60 bytes. This means that 25 TCP SYN packets can fit into the space occupied by one normal-size IP packet. It also means that if nothing but TCP SYN packets are sent to a router, they can arrive at a rate that's 25 times higher than typical Internet packets. Since every one of those SYNs are valid IP packets, each one must be

examined and routed to the proper interface. But few routers are able to handle 25 times the routing rate demand that they were designed to. Routers have short buffers which are used to even out the incoming flow, queuing packets briefly, as needed. But when a router's switch fabric is overloaded, those buffers will quickly overflow with the majority of packets being dropped and lost forever. The original senders of legitimate packets will then retransmit the dropped packets, which just makes things worse. And as a consequence, the packets of valid traffic will be very likely to be dropped. So once again we have a denial of service. In this instance, the resource being consumed by the TCP SYN flood is the network's routing capacity near the targeted web server. It's there where traffic from across the Internet becomes concentrated enough to begin collapsing nearby routers. The targeted web server never even experiences most of the attacking traffic, because it can't even reach the server, and neither can legitimate traffic. This type of SYN Flood is an attack that leverages the IP protocol, so it would be an attack on layer 3.

TCP SYN flooding attacks are typically generated by a fleet of Bots widely spread across the Internet. They each aim at the same server, and as their traffic is passed from router to router, it is successively aggregated and concentrated into ever more dense streams of TCP SYN packets until it reaches a router that cannot manage to handle the packet rate and that router begins dropping everyone's packets — both legitimate and malicious. Since the attackers are widely distributed across the Internet, this is a classic Distributed Denial of Service (DDoS) attack.

Over time, TCP/IP stacks were redesigned to tolerate the layer 4 SYN trickle attacks and Internet routers were upgraded to handle the high rates of small packets at layer 3 without collapsing.

So more brute force methods were devised.

Someone noticed that a very small query to a DNS server could result in a very large reply. Since DNS servers reply to the IP that queried, the UNIX raw socket API was used to again "spoof" the source IP address. And thus were born the so-called reflection/amplification attacks. Even one determined attacker with a high-speed connection might bring down a large site. The attacker would send tiny DNS queries as fast as their connection would allow, spraying them out across the Internet's publicly available DNS servers. And in each case the attacker would "spoof" their IP to be that of the target server. This would cause all of those DNS servers across the Internet to direct their large reply packets to that single spoofed IP.

In this case, it wasn't a router's switching engine fabric that would be overwhelmed, it was the total bandwidth of the site's connections to the Internet. Most web surfing traffic is very bursty. We look at a web page for a while before we click another link. Many times the site doesn't have what we're looking for, so we'll go somewhere else by hitting our browser's [BACK] button. And even when we do load a page, these days a huge percentage of a website's content is delivered by other Internet web servers. The result of this natural burstiness and source distribution of web traffic is that sites are not scaled to handle massive amounts of continuous traffic... because that never happens... unless the site is being subjected to a significant bandwidth attack. All that's needed is to overwhelm a website's scaling for normal daily traffic, and it turns out that's not too difficult. When aided by raw socket IP address spoofing and servers like DNS that can be induced to amplify traffic, it's often feasible to readily overwhelm a site's connection bandwidth.

But over time, this too was overcome by aggregating many sites to share much larger Internet connection bandwidth. One hundred websites might share 100 times the bandwidth. Individually,

the economics still make sense and they're still getting the same amount of bandwidth. But by pooling their individual bandwidths to create a much larger shared bandwidth pool, they're now able to transiently "borrow" from that pool, as needed, to thwart large bandwidth attacks while remaining on the air. Cloudflare and other large providers have been quite successful in offering anti-DDoS protection by "fronting" for their website clients.

But today's modern websites are dynamic. They do not deliver static HTML pages which sit as text files on non-volatile disk storage. When you see a site whose page URLs end in .php, .asp or .jsp you're seeing a page which doesn't actually exist anywhere as an HTML web page. Instead, it was created by invoking a script. Many sites like those created by Wordpress, any modern shopping or catalogue site, eBay, Craigslist or web forum only exist as collections of virtual pages, created on-the-fly, on demand, from a collection of SQL database queries emitted by PHP, Active Server Pages or Java Server Pages and then knit together to produce finished HTML.

But it turns out that running that scripting, serving those SQL database queries, and assembling those finished pages can be quite compute intensive. It's neat and flexible to be assembling pages on the fly from templated style sheets, but this approach inherently introduces a new bottleneck which can be attacked and which may prove significantly more difficult to filter and block. Whereas yesterday's attacks, being measured in packets per second or bits per second, were "transport layer" capacity attacks These are also commonly referred to as OSI layer 3 or 4 attacks — with layer 3 being IP and layer 4 being TCP. This newest form of attack is an "application layer" capacity attack where the attack strength is measured in requests made per second (RPS). Similarly, these are often referred to as OSI layer 7 — application layer — attacks.

And this brings us to the topic of today's podcast: "The Mēris Botnet."

We started this discussion by talking about Yandex. Like many large firms, Yandex has their own internal network security people while also employing the talents of external specialty firms who have an inherently broader scope and are able to provide more experience and specialization as well as network services to their clients. Yandex's external partner is Qrator Labs.

Qrator Labs is a large European Internet security and attack mitigation company. They maintain offices in Prague in the Czech Republic, in Dubai in the UAE, and in Moscow in Russia. Their page on DDoS attack prevention explains:

https://qrator.net/en/qrator-technologies/nejtralizaciya-ddos-atak

> Automatic DDoS mitigation at all OSI levels up to L7 (application level) inclusive, on all the tariff plans, with no exception.
>
> All layers of protection solutions by Qrator Labs solution work together in a connected complex. This approach serves as base for neutralization of even the most complex attacks, which sometimes combine DDoS attacks on channel capacity with the attacks on the layer of web applications (L7), frequently accompanied by hacking.
>
> The system blocks not only high-speed network layer attacks (L3 and L4), but also low-frequency destructive L7 attacks which can only be identified through behavioral or correlational analysis and continuous traffic monitoring. The speed of reaction to DDoS attacks is reduced to the absolute minimum as the system works in fully automatic mode, which also

ensures the uptime of clients' web resources 24/7.

Last Thursday, Qrator published a blog posting that surprised and worried many in the Internet community and generated many headlines. It's title was: "Mēris botnet, climbing to the record"

They wrote:

For the last five years, there have virtually been almost no global-scale application-layer attacks. During this period, the industry has learned how to cope with the high bandwidth network layer attacks, including amplification-based ones. It does not mean that botnets are now harmless.

End of June 2021, Qrator Labs started to see signs of a new assaulting force on the Internet – a botnet of a new kind. That is a joint research we conducted together with Yandex to elaborate on the specifics of the DDoS attacks enabler emerging in almost real-time.

We see here a pretty substantial attacking force – dozens of thousands of host devices, growing. Separately, Qrator Labs saw the 30,000 host devices in actual numbers through several attacks, and Yandex collected the data [of] about 56,000 attacking hosts.

[I need to pause here for a moment to talk about identifying the attacking devices. In DNS, for example, reflection attacks it's the DNS servers themselves that appear to be swamping the target with unasked for and unwanted DNS replies. And anytime IP addresses are being spoofed there's no ready way to identify the attacker from the traffic. But this is NOT the case whenever an full TCP connection is completed, as with any form of level 7 attack. In order for a malicious client to attack a website by issuing a web query, the site MUST have a valid IP for the other endpoint. In other words, layer 7 attacks cannot be "blind" attacks.

So, when they say "*We see here a pretty substantial attacking force – dozens of thousands of host devices, growing. Separately, Qrator Labs saw the 30,000 host devices in actual numbers through several attacks, and Yandex collected the data [of] about 56,000 attacking hosts.*" — they actually have those device's IP addresses.]

However, we suppose the number to be higher – probably more than 200 000 devices, due to the rotation and absence of will to show the "full force" attacking at once. Moreover, all those being highly capable devices, not your typical IoT blinker connected to WiFi – here we speak of a botnet consisting of, with the highest probability, devices connected through the Ethernet connection – network devices, primarily.

Some people and organizations already called the botnet "a return of Mirai", which we do not think to be accurate. Mirai possessed a higher number of compromised devices united under C2C, and it attacked mainly with volumetric traffic.

We have not seen the malicious code, and we are not ready to tell yet if it is somehow related to the Mirai family or not. We tend to think that it is not, since the devices it unites under one umbrella seems to be related to only one manufacturer – ***Mikrotik.***

Another reason we wanted to name this particular botnet, operating under elusive C2C, with a different name – Mēris, which means "Plague" in the Latvian language. It seems appropriate and relatively close to Mirai in terms of pronunciation.

Specific features of Mēris botnet:

- *Socks4 proxy at the affected device (unconfirmed, although Mikrotik devices use socks4)*
- *Use of HTTP pipelining (http/1.1) technique for DDoS attacks (confirmed)*
- *Making the DDoS attacks themselves RPS-based (confirmed)*
- *Open port 5678 (confirmed)*

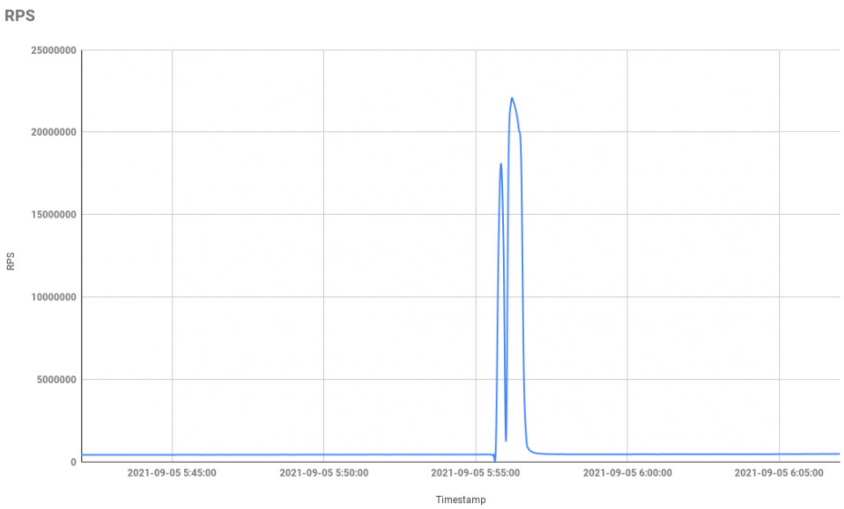
We do not know precisely what particular vulnerabilities lead to the situation where Mikrotik devices are being compromised on such a large scale. Several records at the Mikrotik forum indicate that its customers experienced hacking attempts on older versions of RouterOS, particularly 6.40.1 from 2017. If this is correct and we see that old vulnerability still being active on thousands of devices being unpatched and unupgraded, this is horrible news. However, our data with Yandex indicates that this is not true – because the spectrum of RouterOS versions we see across this botnet varies from years old to recent. The largest share belongs to the version of firmware previous to the current Stable one.

It is also clear that this particular botnet is still growing. There is a suggestion that the botnet could grow in force through password brute-forcing, although we tend to neglect that as a slight possibility. That looks like some vulnerability that was either kept secret before the massive campaign's start or sold on the black market.

It is not our job to investigate the origins, so we must move on with our observations.

In the last couple of weeks, we have seen devastating attacks towards New Zealand, United States and Russia, which we all attribute to this botnet species. Now it can overwhelm almost any infrastructure, including some highly robust networks. All this is due to the enormous RPS power that it brings.

It's been in the news lately about "largest DDoS attack on Russian internet and Yandex", but we at Yandex saw a picture much bigger than that. Cloudflare recorded the first attacks of this type. Their blog post of August 19, 2021, mentioned the attack reaching 17M requests per second. We observed similar durations and distributions across countries and reported this information to Cloudflare.



Requests per Second graph of a DDoS attack on Yandex, September 5, 2021

Here is the history of attacks from the same botnet we recorded at Yandex:

- *2021-08-07 - 5.2 M rps*

- *2021-08-09 - 6.5 M rps*
- *2021-08-29 - 9.6 M rps*
- *2021-08-31 - 10.9 M rps*
- *2021-09-05 - 21.8 M rps*

Yandex' security team members managed to establish a clear view of the botnet's internal structure. L2TP tunnels are used for internetwork communications. [L2TP is Layer 2 Tunneling Protocol. It's often used to encapsulate VPN traffic.] *The number of infected devices, according to the botnet internals we've seen, reaches 250,000.*

The attacking sources (IP addresses), which are not spoofed, seem to predominantly have the same trait – open ports 2000 and 5678. Of course, many vendors put their services onto those particular ports. However, the specific combination of port 2000 "Bandwidth test server" and port 5678 "Mikrotik Neighbor Discovery Protocol" makes it almost impossible to ignore.

Although Mikrotik uses UDP for its standard service on port 5678, an open TCP port is detected on compromised devices. This kind of disguise might be one of the reasons devices got hacked unnoticed by their owners.

Based on this intel, we decided to probe the TCP port 5678 with the help of Qrator.Radar. The results we have gathered were surprising and frightening at the same time.

It turns out that there are 328,723 active hosts on the Internet replying to the TCP probe on port 5678. Of course, not necessarily each of those is a vulnerable Mikrotik – there is evidence that Linksys devices also use TCP service on port 5678. There may be more, but at the same time, we have to assume that this number might represent the entire active botnet.

United States of America	*139930*	*42.6%*
China	*61994*	*18.9%*
Brazil	*9244*	*2.8%*
Indonesia	*7359*	*2.2%*
India	*6767*	*2.1%*
Hong Kong	*5225*	*1.6%*
Japan	*4928*	*1.5%*
Sweden	*4750*	*1.4%*
South Africa	*4729*	*1.4%*

The HTTP pipelining they mentioned is another concern. This indicates that these evil bots were well designed for this application. We've talked about the advances in HTTP that allow for multiple outstanding requests to be pipelined — meaning that multiple queries can be sent to the server for it to handle and answer as it sees fit, even if out of order.

By using pipelined requests we have something that's vaguely reminiscent of the earlier level 3 and level 4 flooding attacks, but far more devastating and difficult to block. The only upside is that the attacking IP addresses cannot be spoofed, so they could be blocked. But blocking a list of 328,723 remote IPs comes with its own challenges. How exactly does one do that. That's a LOT of individual discrete IPs.
And these are also HTTPS requests, meaning that no external agency is able to see into the queries until the targeted website provides their TLS certificate so that the filtering agency can

accept and terminate the incoming TLS connection.

The other clever thing that HTTP pipelining brings is a reduced need for TLS connection setup. If multiple requests are sent down a single TLS connection, less time is spend establishing a connection.

So we have a bot fleet that was purpose-designed for RPS attacks which is squeezing every last bit of air out of the TLS-encrypted TCP connection it has opened with the remote web service by shoveling-in HTTP web requests. And remember that these requests can **all** be valid HTTP web requests. The only thing malicious about them is that too many of them are being sent in from repeat web clients.

In order for a service to remain in the air, non-attacking HTTP queries must all be answered.

It means that front-end optimization could hurt even more by answering every particular network request, as the pipelining is basically about sending requests in batches to the aimed server, making him answer those trash batches of requests. Furthermore, we are speaking of HTTP requests, which already constitute the most computing power of a server, more so if we are talking about a secured connection, bearing the cryptographic load on top of usual requests.

Requests pipelining (in HTTP 1.1) is the primary source of trouble for anyone who meets that particular botnet. Although browsers do not usually use pipelining (except Pale Moon web browser, which does that), bots do that. Moreover, it is easy to recognize and mitigate since the universally accepted TCP sequence is request-response and not the request-request-request-response.

What we see here is a "quality vs quantity" scenario. Because of the request pipelining technique, attackers could squeeze much more RPS than botnets usually do. It happened because traditional mitigation measures would, of course, block the source IP. However, some requests (about 10-20) left in the buffers are processed even after the IP is blocked.

What to do in such a situation?

Blacklists are still a thing. Since those attacks are not spoofed, every victim sees the attack origin as it is. Blocking it for a while should be enough to thwart the attack and not disturb the possible end-user.

Although it is, of course, unclear how the C2C owners for the Mēris botnet would act in the future – they could be taking advantage of the compromised devices, making 100% of its capacity (both bandwidth and processor wise) into their hands.

In this case, there is no other way other than blocking every consecutive request after the first one, preventing answering the pipelined requests.

If there is no DDoS attack mitigation at the targeted server whatsoever, request pipelining could turn into a disaster, as the attacker needs much less workforce to fill the RPS threshold for the victim. And it turns out that many were not ready for such a scenario.
We have contacted Mikrotik with the data we have gathered, trying to find a solution, or notify them of our observations, minimally, which we were able to do yesterday.

Last but not least – please, keep your network devices updated with the latest firmware possible all the time. It is about each router and modem, every internet-connected device and the future of the Internet as a whole. And change your passwords. Nobody knows yet if that's not the brute force against administration passwords – better be safe, than sorry.

So what we've watched over the years is a gradual evolution in attack technology.

In those quaint old days source IPs were being spoofed to hide the attacker's identity when trickling TCP SYN packets into a server was sufficient to bring it down. Then, later, source IPs were being spoofed to bounce reflection/amplification attacks off of other machines on the Internet, causing them to indirectly attack the target with a pure bandwidth flood.

But today, we have such a large volume of vulnerable Internet appliances deployed globally that it's not necessary to hide. And these RPS — requests per second — attacks cannot be spoofed. They are also dauntingly difficult to block because each individual request is valid, and it's only when a given IP makes too many requests within a short time that the client begins to look like a bot.

And having just written that, although Qrator's write-up didn't mention it, this could be the main reason for employing HTTP pipelining: With HTTPS and TLS, the only thing that can be seen by any edge-filtering anti-DoS service is the connection, not the contents. So the use of HTTP pipelining serves as a means for a bot to hide that it's actually attacking the server, since all of its multiple attack queries are slipped down a single persistent connection. It'll just remain connected, pummeling its target with time consuming and expensive-to-answer web queries for as long as it's able.

Someone, somewhere, has built, assembled and is in control of a horrifically powerful botnet unlike anything seen before. It consists of upwards of a quarter of a million MicroTik routers, nearly half of them with IP addresses in the United States. (Perhaps "mi-crot'-ick" was the proper pronunciation after all!)

Those brief 60-second attacks against Yandex weren't meant to hurt them, they were meant to give the attackers some sense for the scale of this new offensive weapon they have created. So they must be feeling quite pleased with themselves now. NOTHING can withstand 21 million web requests per second. Defensive DDoS measures have been improving over time. Unfortunately, so have the attacks. It is a true shame that so much complexity and cost is being added to a beautifully simple system — the IP-based Internet — just to protect it from abuse.