

Security Now! #1030 - 06-17-25

Internet Foreground Radiation

This week on Security Now!

- An exploited iOS iMessage vulnerability Apple denies?
- The NPM repository is under siege with no end in sight.
- Were Comcast and Digital Realty compromised? Don't ask them.
- Matthew Green agrees: XChat does not offer true security.
- We may know how Russia is convicting Telegram users.
- Microsoft finally decides to block two insane Outlook file types.
- 40,000 openly available video camera are online. Who owns them?
- Running SpinRite on encrypted drives.
- An LLM describes Steve's (my) evolution on Microsoft security.
- What do we know about the bots that are scanning the Internet?

A perfect summation of where we are today with AI:



vas  @vasumanmoza · 9h



Claude 4 just refactored my entire codebase in one call.

25 tool invocations. 3,000+ new lines. 12 brand new files.

It modularized everything. Broke up monoliths. Cleaned up spaghetti.

None of it worked.
But boy was it beautiful.

Security News

An iOS iMessage nickname vulnerability?

The mobile threat hunting security firm iVerify posted the news of their discovery with the headline: "*iVerify Uncovers Evidence of Zero-Click Mobile Exploitation in the U.S.*" They wrote:

Throughout late 2024 and early 2025, iVerify detected anomalous activity on iPhones belonging to individuals affiliated with political campaigns, media organizations, A.I. companies and governments operating in the United States and European Union.

Specifically, we detected exceedingly rare crashes typically associated with sophisticated zero-click attacks via iMessage – an exploitation technique previously unobserved in any systematic way in the United States. Subsequent forensic examination of several of these devices ultimately revealed a previously unknown vulnerability in the "imagent" process which, owing to its relative position in the operating system and functionality, would provide attackers a primitive for further exploitation. This vulnerability was patched by Apple in iOS 18.3. We've dubbed this vulnerability NICKNAME.

In the course of our investigation, we discovered evidence suggesting – but not definitively proving – this vulnerability was exploited in targeted attacks as recently as March of this year. Specifically, we learned that Apple sent Threat Notifications to at least one device belonging to a senior government official in the EU on which we saw the highly anomalous crashes. Likewise, one device demonstrated behavior frequently associated with successful exploitation, specifically the creation and deletion of iMessage attachments in bulk within a matter of seconds on several occasions after an anomalous crash. We only observed these crashes on devices belonging to extremely high value targets. And these crashes constituted only .0001% of the crash log telemetry taken from a sample of 50,000 iPhones.

While this evidence does not definitively prove exploitation, it is nonetheless difficult to ignore and merits a public discussion, particularly in light of SignalGate. Our findings suggest it doesn't matter what channel is being used to communicate if the device itself is compromised; attackers have access to all conversations, regardless of whether those happen over Signal, Gmail, or any secure application. This is why it's crucial that organizations on the front lines of digital conflict – including the US government – adapt their mobile security models to face modern threats.

Our findings have been vetted by multiple, independent third parties, including iOS security experts, such as Patrick Wardle from the Objective-By-The-Sea foundation who have confidence in our conclusion that mobile compromise is real, not academic or hypothetical, and that it's happening here in the United States.

So what exactly are those findings?

So far, we've observed six devices total that we believe were targeted for exploitation by this threat actor, four of which demonstrated clear signatures associated with NICKNAME, and two which demonstrated clear signs of successful exploitation. Interestingly, all of the victims had either previously been targeted by the Chinese Communist Party (CCP), meaning they were confirmed to have also been targeted by Salt Typhoon; they were engaging in business pursuits counter to or of particular interest to the CCP; or they had engaged in some sort of activism against the CCP.

We don't have enough evidence to make clear attribution or a full view of an exploit chain, but the circumstantial evidence could indicate CCP.

How does it work?

iPhones allow you to set a nickname or avatar for numbers in your contact list. The vulnerability is likely triggered by sending repeated, rapid-fire nickname updates to iMessage, which results in a use-after-free memory corruption. This makes NICKNAME a good candidate for a primitive to pivot off as part of a longer exploit chain. We believe this vulnerability correlates with successful iPhone exploitation, due to four concurrent factors:

- *The extreme rarity of these specific crash patterns (<0.001% of all crash logs)*
- *Their exclusive appearance on devices belonging to high-value targets*
- *Similarity to crash patterns seen in known spyware attacks*
- *Evidence of successful exploitation, including the receipt of at least one Apple Threat Notification proximal to the observed behavior and evidence of 'cleaning' behavior*

Is it Still Active?

Differential analysis reveals the vulnerability was patched in the iOS 18.3.1 release; however, NICKNAME could be one link in a larger exploit chain. It is possible that there are other elements of the exploit chain that are still active, which is why we're only speaking about the link in the chain that has definitively been patched.

We provide a full technical analysis and look forward to sharing any additional material findings when our investigation concludes:

<https://welcome.iverify.io/hubfs/iVerify-Nickname-Vulnerability-Report.pdf>

I've included a link to their full technical report in the show notes. It's important to also disclose that Apple is actively contesting this. Axios reported:

Apple has fixed the flaw — which was present in iOS versions through 18.1.1 — but disputes that it was ever used to hack devices.

- *Ivan Krstić, head of Apple Security Engineering and Architecture, said in a statement: "We've thoroughly analyzed the information provided by iVerify, and strongly disagree with the claims of a targeted attack against our users."*
- *Krstić added: Apple confirmed the underlying Nickname bug, but said its own field data from iPhones points to it being a "conventional software bug that we identified and fixed in iOS 18.3."*
- *He said: "iVerify has not responded with meaningful technical evidence supporting their claims, and we are not currently aware of any credible indication that the bug points to an exploitation attempt or active attack. We are constantly working to stay ahead of new and emerging threats, and will continue to work tirelessly to protect our users."*

So the results are, at best, ambiguous. On the iVerify side, *"if it walks like a duck and quacks like a duck"* – which this flaw did – then it would be reasonable to conclude that it's probably a duck.

But as we observed and talked about long ago back when Kaspersky discovered some of their iPhones containing malware, the very fact that iPhones have been so tightly locked down actively thwarts the type of post-exploitation forensic analysis that would allow 3rd parties like iVerify to help put pressure on and keep Apple honest – if, indeed, they were not already being.

The trouble is that the stakes in all of this have been raised to such a high level. iVerify referred to SignalGate in their posting, reminding us of the threat that high level classified military operations planning was being conducted on non-secured civilian smartphone hardware.

The iVerify disclosure ended in an important reminder. They wrote:

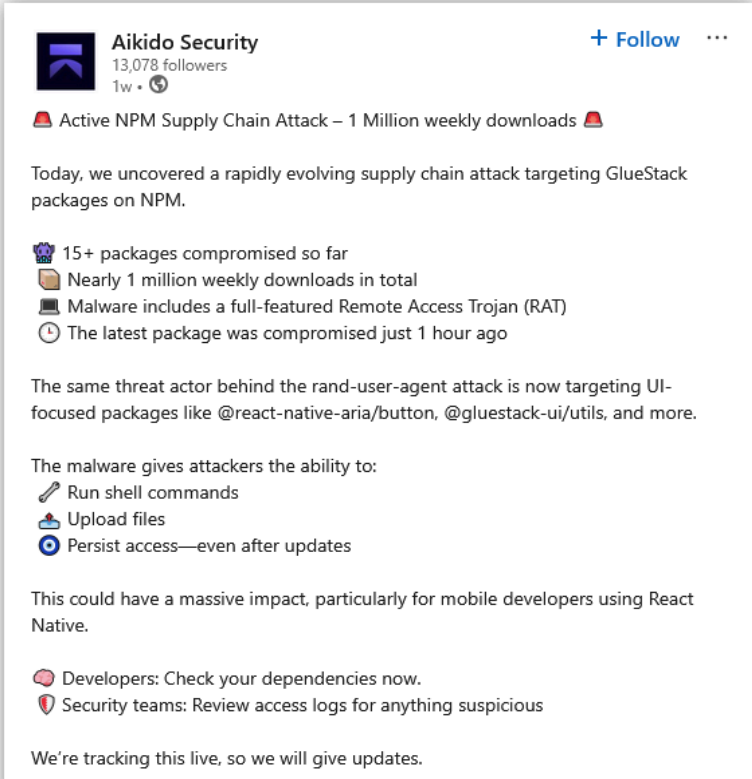
iVerify recommends that high-risk users keep their phones updated and turn on Apple's Lockdown Mode, which is designed to guard against spyware. iVerify's COO Rocky Cole said that it's likely that Lockdown Mode could have prevented these potential infections.

Given that all of the evidence continues to show – just consider last month's Pwn2Own competition against fully patched systems – that we do not currently have the technology or capability to perfectly secure our devices, having a bifurcated feature set where fewer features can be offered for greater security makes all kinds of sense. It's like Microsoft disabling their Edge browser's JIT – Just In Time – JavaScript compiler features after observing that 80% of their Chromium-based browser's security problems are being discovered there. If no amount of effort appears able to make it safe, then just turn it off when safety matters more than features.

NPM Under Siege

Two quick bits of news should serve to remind us that the open source library system is more or less under constant attack. The Node JavaScript Package Manager (NPM) facility description reads: *"Relied upon by more than 17 million developers worldwide, npm is committed to making JavaScript development elegant, productive, and safe. The free npm Registry has become the center of JavaScript code sharing, and with more than two million packages, the largest software registry in the world. Our other tools and services take the Registry, and the work you do around it, to the next level."* Unfortunately, its openness is also its challenge. The first piece of news that caused to to pause was:

Eighty-four malicious npm packages were discovered and taken down last week. Check out the GitHub security advisory portal for more details. This also includes two packages spotted by Socket that would wipe production systems.



The screenshot shows a tweet from the account 'Aikido Security' (13,078 followers). The tweet is titled 'Active NPM Supply Chain Attack – 1 Million weekly downloads' and contains the following text: 'Today, we uncovered a rapidly evolving supply chain attack targeting GlueStack packages on NPM.' It lists several statistics: '15+ packages compromised so far', 'Nearly 1 million weekly downloads in total', 'Malware includes a full-featured Remote Access Trojan (RAT)', and 'The latest package was compromised just 1 hour ago'. It also mentions that the same threat actor is targeting UI-focused packages like '@react-native-aria/button', '@gluestack-ui/utils', and more. The tweet lists the capabilities of the malware: 'Run shell commands', 'Upload files', and 'Persist access—even after updates'. It concludes with a warning to developers to check their dependencies and for security teams to review access logs. The tweet ends with 'We're tracking this live, so we will give updates.'

And the second was:

A threat actor has compromised 16 npm libraries from the Gluestack UI framework. The attacker compromised a Gluestack admin's account, added a remote access trojan to the libraries, and pushed updates on Friday. The affected packages are extremely popular and have almost one million weekly downloads. Aikido Security says the attacker is the same threat actor behind another supply chain attack on the rand-user-agent package last month.

So this is another PSA – Programmer Service Announcement – reminding and cautioning all of our listeners who may avail themselves of the true value of shared open libraries to always remain vigilant and aware that not everyone who places code there is motivated by altruism.

Were COMCAST and “Digital Realty” compromised?

I was scanning reports of a possible undisclosed breach of COMCAST and the major data center enterprise “Digital Realty” when I encountered this comment:

*“Inside two major U.S. telecom operators, incident response staff have been instructed by outside counsel **not to look for signs of Salt Typhoon**, said one of the people, declining to name the firms because the matter is sensitive.”*

Wow. So that’s what has evolved from the intersection of big business, cyber security and legal accountability. The reporting is from NextGov’s cybersecurity reporter. The headline of the story was *“US agencies assessed Chinese telecom hackers likely hit data center and residential internet providers”*. This headline which teases us with the phrase “US agencies” begs the question, which US agencies made this assessment. To that end, the reporting says:

Two U.S. security agencies listed mass media provider Comcast and data center giant Digital Realty among companies likely ensnared by a Chinese hacking group previously found inside major U.S. and global telecom operators, according to three people familiar with the matter.

Right. And guess who those two agencies were?

*The **National Security Agency** (our US NSA) made the determination that Comcast had likely been impacted by the group, known as Salt Typhoon, according to two of the people. The Cybersecurity and Infrastructure Security Agency (CISA) cataloged Digital Realty as being potentially compromised, the third person said. The people spoke on the condition of anonymity to discuss the matter’s sensitivity.*

Salt Typhoon breached major telecom carriers in a global, multi-year espionage campaign uncovered last year. Over time, news trickled out about the scope and scale of the incident, which was first reported by The Wall Street Journal.

The hacking unit is part of a broader syndicate of state-backed groups tied to different military and intelligence arms of China’s central government. The “Typhoon” moniker comes from a Microsoft naming convention for Beijing-linked cyber actors.

Such intrusions, especially into a data center environment, could give the hackers a potentially far deeper foothold into infrastructure supporting the world’s information service providers than previously known. The agencies’ assessments have not been previously reported.

There's uncertainty among officials about who was impacted by Salt Typhoon. Various agencies across the U.S. government are in possession of lists of confirmed or potential victims, but it's not clear if the tallies are consistent with each other, adding to confusion about who may have been accessed, targeted or marked for investigation. CISA, for instance, is in possession of a list of both telecom and information technology companies, but an FBI tabulation shows different entities.

And here it comes:

Making investigations into the breach more complicated is that multiple telecom providers have invoked legal strategies to protect themselves from disclosing compromise by the hackers. Inside two major U.S. telecom operators, incident response staff have been instructed by outside counsel not to look for signs of Salt Typhoon, said one of the people, declining to name the firms because the matter is sensitive.

So now we have deliberate and internally formalized "heads buried in the sand" strategies in place because employees may be deposed under oath. I hope any cross-examining counsel has the presence of mind to ask whether they were ever instructed by anyone to avoid looking for signs of external intrusion. The article continues:

One of the sources said that having been assessed as likely victims, CISA representatives should have contacted Digital Realty and Comcast multiple times since December. It's not clear whether consistent back-and-forth communications were established. CISA tends to initiate outreach to potential victims when it's believed their networks are compromised, according to another person familiar with the cyber defense agency's notification process.

A new concern this year is that CISA has recently suffered a significant and controversial reduction in personnel as a result of the job cuts enacted by DOGE. In the same way that it's impossible to prove a negative, it can be challenging to justify the presence of staff whose job is to prevent trouble. This is the problem many corporate CISOs face, but on a government agency scale. Someone challenges *"What are all those people over there doing?"* to which the reply is *"They're keeping an eye on things."*, which is then followed by the difficult to defend challenge *"So, why do we need so many of them?"*

As all of our listeners will appreciate, CISA has been doing a surprisingly tremendous job since it really got rolling. I have often commented that I've been surprised by how proactive and effective it has been – especially considering that it's a government agency. I hope these cutbacks will not compromise that. It's probably impossible to accurately gauge since we cannot know how things would have been with a significantly smaller CISA. We'll just need to watch and see. The reporting continues:

A Comcast spokesperson told Nextgov: "We've worked with law enforcement and government agencies and have closely monitored our network. We have found no evidence that Salt Typhoon has impacted our enterprise network."

An intrusion into either provider could carry significant national security risks. Comcast facilitates internet access for millions of users and businesses, while Digital Realty hosts troves of physical infrastructure used by telecom operators, cloud providers and governments to route global web traffic.

A CISA spokesperson said: "As a policy, we do not comment on individual entities." The NSA declined to comment, and the FBI did not respond to a request for comment. Digital Realty did not return multiple requests for comment.

Nextgov reported in December that hundreds of organizations were notified of potential Salt Typhoon compromise. Last month, CyberScoop reported that CISA and the FBI devised a coordinated notification campaign to alert affected companies and help them deter the hacks, sometimes providing new data on an hourly basis.

The FBI concurred with other agency assessments that the Salt Typhoon attacks, broadly speaking, are the most egregious national security breach in U.S. history by a nation-state hacking group.

Marc Rogers, a seasoned telecommunications cybersecurity expert said: "[A breach of Comcast and Digital Realty] would confirm what many of us in the cybersecurity industry already suspected: That the Salt campaign was broader than just telcos and we have low confidence the attackers have been evicted."

Nextgov obtained an internal CISA list of communications sector hardware and software products found to have been exploited by China-linked hacking groups. Of several listed, one of those vulnerabilities, first discovered in 2018, was found in MikroTik routers and some of the software flaws exploited by Salt Typhoon were first disclosed in 2018.

Marc Rogers said: "Something that isn't being talked about enough is that the initial way in which these attackers used were simple flaws like 8-year-old vulnerabilities and credential theft. Instead of talking about 'ripping and replacing' we should be looking at why we aren't simply patching and maintaining our existing critical infrastructure."

Eric Hanselman, the chief technology, media and telecommunications research analyst at S&P Global Market Intelligence explained that: "Chinese access into datacenter and colocation firms would provide the hackers with a different target set compared to messaging services operated by traditional carriers. The additional risk created would be their gaining the ability to monitor intra-service and intra-application communications traffic that doesn't normally traverse the internet backbone. That could include storage traffic moving from colocation environments into cloud or traffic moving from hosted environments into on-premises infrastructure. That traffic might have less robust protections, as it's not traversing the open internet."

Digital Realty has over 300 data centers across 25 countries and 50 metropolitan areas, according to a company marketing webpage. They list Amazon Web Services, Google Cloud, IBM, Microsoft and Nvidia among their clients. The company is considered one of the largest data center colocation providers in the world, housing the physical systems where cloud and telecom networks exchange data.

Eric Hanselman said: "We can reasonably assume that these attackers already have sufficient access into internet infrastructure and are looking to expand the depth with which they can monitor other activities that are taking place within data center environments."

Comcast's broadband and cable customer base is around 51 million, while its total wireless customer count totals about 8.1 million, according to recent earnings data.

It is widely believed that Salt Typhoon has still not been excised from telecom systems, despite public statements from companies saying otherwise. On Thursday, well known Republican Senator Josh Hawley said in a Senate Homeland Security Committee hearing that the hackers are still inside. He said: "If a foreign actor chose to concentrate on any member of the audience here — we were told behind closed doors, of course — but what we were told is that foreign actors basically have unlimited access to our voice messages and our telephone calls."

President Donald Trump, Vice President JD Vance and a range of U.S. officials had their calls and texts directly targeted in the Salt Typhoon hacks. The cyberspies accessed providers' "lawful intercept" systems, used to comply with government orders requiring access to communications metadata for law enforcement investigations.

Wow. And remember that as we previously saw, Salt Typhoon's apparent way into these major telecom backbone providers was not rocket science nor advanced Pwn2Own-style elite hacking. It was simply that somewhere within telecom sprawling and largely out-of-control infrastructure were older unpatched systems still online with known vulnerabilities. The reporting says:

A spokesperson for the House China Select Committee said in an email: "If these reports are accurate, they point to yet another serious and deeply concerning example of the Chinese Communist Party targeting America's digital infrastructure." and noted that: "the panel has repeatedly warned about the CCP's efforts to exploit access points into our communications networks, and this apparent breach reinforces the urgent need to harden our defenses."

In March, the House's Homeland Security Committee chair Republican Representative Mark Green of Tennessee sent a request to DHS asking the agency to transmit internal documents about Salt Typhoon and another Chinese hacking unit, Volt Typhoon. Green said in a statement to Nextgov: "Every new detail that emerges surrounding the Salt Typhoon intrusions teaches us the lengths China-backed hackers will go to undermine the integrity of our critical infrastructure, U.S. sovereignty and the privacy of Americans." Green said this in reference to recent testimony from DHS Secretary Kristi Noem saying CISA is lacking detailed information about the telecom hacks.

It's difficult not to wonder whether some additional manpower at CISA might help. Green added:

"My colleagues and I on the committee share this concern, which is why we sent a letter in March to examine the previous administration's response to the Volt and Salt Typhoon intrusions."

I was about to comment on that when I saw that Nextgov's reporting had already done so. They wrote:

The Cyber Safety Review Board — a DHS body that was dismissed at the start of the Trump administration — was in the middle of investigating the Chinese telecom hacks. Lawmakers have called for it to be reinstated. CISA has also been mired in budget plans to slash significant parts of its workforce and operations.

I hope that CISA will be able to recover and rebuild whatever effectiveness it may have lost. It seems pretty clear that private industry is unwilling to expend the cost and effort required to fully secure its own business operations. But when the public depends upon the security of those operations there's a legitimate need for oversight, regulation and accountability.

Matthew Green agrees: "Don't rely upon XChat's privacy"

I mentioned briefly in passing last week that someone named Matthew Garrett had looked at the encryption mechanisms underlying X's supposedly new end-to-end encrypted XChat DM facility and had decided that it was no better than the old one. He shared Elon's declaration about how it was written in Rust when it was still C.

Since then, Matthew Garrett's posting came to the attention of another Matthew. This Matthew was none other than the renowned Johns Hopkins university Cryptographer, Matthew Green. This Matthew is well known to this podcast, so this Matthew's posting last week titled "[A bit more on Twitter/X's new encrypted messaging](#)" is of interest. Matthew's post is longer than we need and [I've included a link to the entire thing in the show notes](#). So I'm just going to share his short bullet-pointed introduction and summary:

Matthew Garrett has a nice post about Twitter (uh, X)'s new end-to-end encryption messaging protocol, which is now called XChat. The TL;DR of Matthew's post is that from a cryptographic perspective, XChat isn't great. The details are all contained within Matthew's post, but here's a quick TL;DR:

- There's no forward secrecy. Unlike Signal protocol, which uses a double-ratchet to continuously update the user's secret keys, the XChat cryptography just encrypts each message under a recipient's long-term public key. The actual encryption mechanism is based on an encryption scheme from libsodium.*
- User private keys are stored at X. XChat stores user private keys at its own servers. To obtain your private keys, you first log into X's key-storage system using a password such as PIN. This is needed to support stateless clients like web browsers, and in fairness it's not dissimilar to what Meta has done with its encryption for Facebook Messenger and Instagram. Of course, those services use Hardware Security Modules (HSMs.)*
- X's key storage is based on "Juicebox." To implement their secret-storage system, XChat uses a protocol called Juicebox. Juicebox "shards" your key material across three servers, so that in principle the loss or compromise of one server won't hurt you.*

Matthew's post correctly identifies that the major vulnerability in X's system is this key storage approach. If decryption keys live in three non-HSM servers that are all under X's control, then X could probably obtain anyone's key and decrypt their messages. X could do this for their own internal purposes: for example because their famously chill owner got angry at some user. Or they could do it because a warrant or subpoena compels them to. If we judge XChat as an end-to-end encryption scheme, this seems like a pretty game-over type of vulnerability.

So in a sense, everything comes down to the security of Juicebox and the specific deployment choices that X made. Since Matthew wrote his post, I've learned a bit more about both of these things. In this post I'd like to go on a slightly deeper dive into the Juicebox portion of X's system. This will hopefully shed some light on what X is up to, and why you should not use XChat.

So Matthew Green concurs with Matthew Garrett which is to say that no one should consider any

encrypted messaging system to be securely end-to-end encrypted when such a system externally stores on its user's behalf their private keys.

A perfect example is Apple's currently controversial Advanced Data Protection. What it explicitly does is give its user's discretionary control over whether or not a copy of their private key is also retained by Apple. Allowing that enables additional features, but also enables Apple to similarly respond to court order subpoenas. In the case of ADP, if that's not what you want, if you're not in the United Kingdom, and all of your devices are running Apple OSes that support ADP – iOS or iPadOS v16.2 or later in the case of the iPhone and iPad – then you can turn that on and a new private key Apple has never seen will be created and shared only among those iDevices.

No one should confuse Apple's state-of-the-art encryption technology with what Elon is peddling. I'm not suggesting that anyone necessarily needs end-to-end encrypted DM's on X, but everyone should be aware that they're not really available there nor on Facebook Messenger or Instagram.

Telegram?

I also recently mentioned that Telegram's encrypted privacy had recently been called into question when Russian citizens who were supporting Ukraine were being arrested and convicted by Russia's FSB. It turns out that the culprit might not be any weakness in Telegram's questionable encryption, but rather a compromise of its network infrastructure. In other words, there may be some leakage of messaging metadata – which we know can be notoriously difficult to prevent – it's why we've gone to the lengths of needing TOR – coupled with the fact that Telegram's network infrastructure appears to be directly under Russia's control. This could explain how people are getting in trouble for who they contact without needing to see inside their messages.

I'm not going to spend any more time on this because this brings us to another of those "don't use it if you really care about your privacy" conclusions. But I'll leave a link to extremely detailed coverage in the show notes: <https://istories.media/en/stories/2025/06/10/telegram-fsb/> With reporting titled: *"Telegram, the FSB, and the Man in the Middle: The technical infrastructure that underpins Telegram is controlled by a man whose companies have collaborated with Russian intelligence services."*

"Microsoft Outlook to block more risky attachments used in attacks"

BleepingComputer brings the news that starting in July, Microsoft Outlook will be blocking two additional file types. They reported:

Microsoft announced it will expand the list of blocked attachments in Outlook Web and the new Outlook for Windows starting next month. The company said in a Microsoft 365 Message Center update that Outlook will block .library-ms and .search-ms file types beginning in July.

Microsoft said: "As part of our ongoing efforts to enhance security in Outlook Web and the New Outlook for Windows, we're updating the default list of blocked file types in OwaMailboxPolicy. Starting in early July 2025, the [.library-ms and .search-ms] file types will be added to the BlockedFileTypes list."

Windows Library files (.library-ms), which define virtual collections of folders and files in the Windows file system, were used earlier this year in phishing attacks targeting government entities and private companies to exploit a Windows vulnerability (CVE-2025-24054) that exposes NT Lan Manager hashes.

Okay. Now let just pause here for a moment to say that if I didn't know that we have many young people listening to this podcast with their parents while they're on their way to school in the morning, as well as in many other settings, and that those parents have grown to trust me to keep the colorfulness of my language under control for those young ears, at this point I would loudly expand upon the well known abbreviation WTF!

Why in the world, Microsoft, would you have EVER, by default, EVER considered allowing ANY email client – which inherently presents as large an attack surface as any web browser, and which is being continually bombarded with unwanted and potentially malicious content – to handle .library-ms files which we are now told, define virtual collections of folders and files?

I've been in this business since long before it was a business and I've never seen a .library-ms file. How is it that this is a file type that all Outlook users' clients should have ever been able to open? And how can that possibly be addressing some need that anyone has?

This is utterly unbelievable to me, as it should be equally unbelievable to anyone trained in the practice of cyber security. How many times have we talked about the security benefit that flows from first blocking everything by default and then only allowing selected known safe and **needed** content through any security perimeter? Email is a security perimeter. This is unbelievable. I'm so surprised by this because any rational security-aware design would never be permitting the reception and handling of – by default – any whacky file type someone at Microsoft might come up with in the future. Okay. So what about this other file type? BleepComputer writes:

The .search-ms URI protocol handler has also been exploited in phishing and malware attacks since at least June 2022, when Hacker House co-founder and security researcher Matthew Hickey found that it could be used to automatically launch Windows Search windows on recipients' devices to trick them into launching malware when chained with a Windows Support Diagnostic Tool (MSDT) remote code execution vulnerability (CVE-2022-30190).

Well, isn't that just peachy? What year was that? Oh yeah, 2022. So it only took Microsoft what? Three years to finally announce that **next month** they plan to start blocking this other unneeded and clearly abuse-prone file extension. Wow. BleepingComputer reports that in Microsoft's announcement, Microsoft wrote:

"The newly blocked file types are rarely used, so most organizations will not be affected by the change. However, if your users are sending and receiving affected attachments, they will report that they are no longer able to open or download them in Outlook Web or the New Outlook for Windows. No action is required if your organization does not rely on these file types. The update will automatically apply to all OWA Mailbox policies in your organization. If your organization needs to allow these file types, you can add them to the AllowedFileTypes property of your users' OwaMailboxPolicy objects before the rollout."

BleepingComputer then explains:

You can find the complete list of blocked Outlook attachments on Microsoft's documentation website. Enterprise users with a Microsoft Exchange Server account can ask Exchange Server administrators to adjust security settings for their mailboxes to accept attachments blocked by Outlook if they can't be shared as an archive, using a different extension, or using OneDrive or SharePoint.

This move is part of a much broader effort to remove or turn off Office and Windows features that have been abused and exploited to infect Microsoft customers with malware. It started in 2018 when Microsoft expanded support for its Antimalware Scan Interface (AMSI) to Office 365 client apps to block attacks using Office VBA macros. Since then, the company began blocking VBA Office macros by default, disabled Excel 4.0 (XLM) macros, introduced XLM macro protection, and started blocking untrusted XLL add-ins by default across Microsoft 365 tenants.

Microsoft also announced in May 2024 that it would kill off VBScript and disabled all ActiveX controls in Windows versions of Microsoft 365 and Office 2024 applications in April 2025.

Again, it's truly inexplicable that Microsoft has been so utterly lame about the security of their email clients on the desktop and in the cloud. The only rational explanation is that this was all originally put in place by engineers who had no training in security. Hubris is the only explanation for a policy of "allow everything to run by default." It is the exact equivalent of having an "allow all" firewall policy and believing that it could ever be secure to only block the dangerous ports.

40,000 openly available video cameras online

As we were recording last week's podcast, Leo encountered the news of 40,000 cameras having been found online. This raised a bunch of questions. The first of which was probably what sort of exploit might have been needed to hack into and compromise such a large inventory of Internet-connected cameras? And the answer is: "None!" All 40,000 of these video cameras are simply online and wide open.

The news of this came from Bitsight, an Internet scanning company that offers to keep an eye on the IPs and ports of its clients to let them know when anything like this might be happening to them. In their report they wrote:

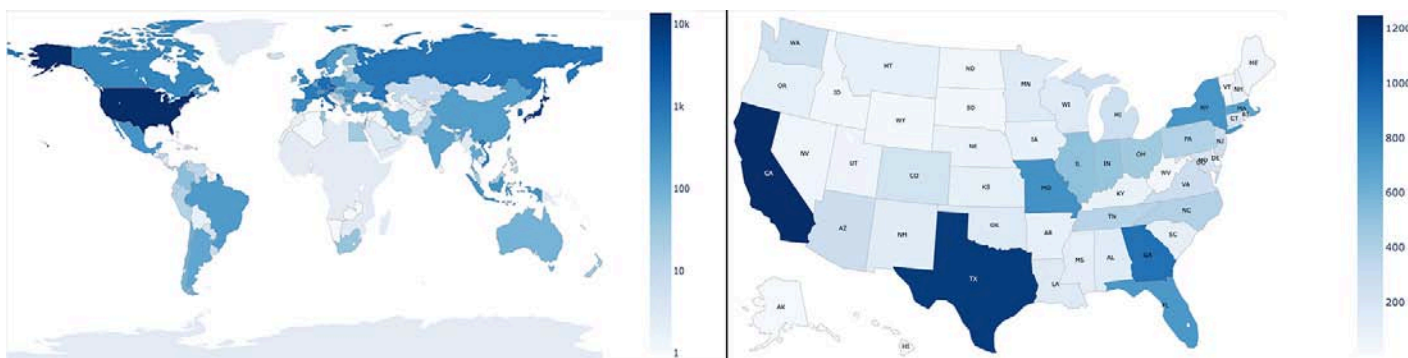
Welcome to 2025, where thousands of internet-connected cameras meant to protect us are actually putting us at risk. In our latest research at Bitsight TRACE, we found over 40,000 exposed cameras streaming live on the internet. No passwords. No protections. Just out there. We first raised the alarm in 2023, and based on this latest study, the situation hasn't gotten any better.

These cameras—intended for security or convenience—have inadvertently become public windows into sensitive spaces, often without their owners' knowledge. No matter the reason why one individual or organization needs this kind of device, the fact that anyone can buy one, plug it in, and start streaming with minimal setup is likely why this is still an ongoing threat.

And it doesn't take elite hacking to access these cameras; in most cases, a regular web browser and a curious mind are all it takes, meaning that 40,000 figure is probably just the tip of the iceberg.

For their key takeaways they wrote:

Bitsight TRACE has found more than 40,000 security cameras openly accessible on the internet, allowing anyone to view their live footage. The United States and Japan rank first and second for camera exposure. Most times, all that an attacker needs to spy on homes or even large organizations is just a web browser and the right IP address. We detected conversations on the dark web where bad actors were discussing exposed cameras. We scanned the entire internet for exposed HTTP-based and RTSP-based cameras. The United States leads the charge with roughly 14,000 exposed online cameras, followed by Japan, Austria, Czechia, and South Korea. Given the high prevalence of exposed cameras in the United States, we also analyzed their distribution across each state:



I grabbed a heat map of the world and of the U.S. What's curious is that the U.S. Map shows that the huge preponderance of open cameras are located in California and Texas. It would be interesting to determine why. The distribution is extremely non-uniform. They noted that:

Not all online cameras are bad: some people stream beaches or birdhouses on purpose. But here's where things get problematic:

- *Residential cameras watching front doors, backyards, and living rooms*
- *Office cameras disclosing whiteboards and screens full of confidential information*
- *Factory cameras showing manufacturing secrets*
- *Even public transportation cameras streaming passengers*

By leveraging the intelligence gathered by our awesome Cyber Threat Intelligence colleagues, we dug into dark web forums where people openly discuss tools and tactics to find and abuse these cameras (some even sell access). This isn't hypothetical: this is happening right now.

They then finish their synopsis with a section titled: *"What should I do to protect myself or my company?"* Their advice is what any long-time follower of this podcast would echo. They wrote:

If you have a security camera at home or manage surveillance cameras for your company, then taking the right precautions can make the difference between keeping your footage private and unintentionally broadcasting it to the world. Here are some simple but essential guidelines to ensure your cameras are secured:

- *Check if your camera is accessible from the internet. Try accessing it from a device outside your home network. If you can reach it remotely without logging in through a secure app or VPN, it might be exposed.*
- *Change default usernames and passwords. Many cameras come with weak or publicly known default credentials. Set a strong, unique password.*

- *Disable remote access if you do not need it. If you only use your camera on your home network, there is no reason to allow outside connections.*
- *Keep firmware up to date. Manufacturers often release security updates that fix vulnerabilities. Regularly check for updates and install them.*

Additionally, if you manage security cameras in your organizations:

- *Restrict access with firewalls and VPNs. Ensure only authorized personnel can access camera feeds using a VPN or firewall rules that block access from untrusted sources.*
- *Monitor for unusual activity. Set up alerts for unexpected login attempts.*

It would be really interesting to follow-up and track down a large set of those cameras to determine whether they are likely being deliberately shared publicly or may be inadvertently exposing parts of the physical world to a global audience that should not have access to it.

SpinRite

My tech support guy, Greg, forwarded an interesting question about SpinRite which bears sharing because its answer is not always intuitive and sometimes results in some confusion, as it did when it was recently posed over in GRC's web forums:

Lee Garrison

Hi Steve, I need to run SpinRite 6.1 on a 3 terabyte partition which is encrypted with VeraCrypt for the purpose of locating and fixing bad sectors on this encrypted partition. The drive is a Western Digital 4TB HDD which also has the rest of the drive space occupied with an unencrypted partition. My question is: "Should I first decrypt this partition with VeraCrypt before running SpinRite on it, or should I leave it encrypted while running SpinRite on it?" We have been discussing this problem over on your GRC Forums under the "Running SpinRite" topic, but with varying opinions persisting.

I went over to the forum to see what the dialog had been over there. It was as Lee had asked. The answer is that SpinRite has NO PROBLEM running on drives that have been encrypted with VeraCrypt, TrueCrypt, Bitlocker, or another other form of encryption. Since SpinRite 6.1 is seeing the drive as nothing more than opaque blocks of data, it doesn't care whether that data might be encrypted or not.

SpinRite's Real Time Monitor screen presents a cool "window" that allows its user to "see" the data that's passing by as SpinRite is working on it, and one of the cool things SpinRite's users notice is that they often see their own recognizable data flashing past that window while SpinRite is running. But that's an example of what encryption would change. When SpinRite is running on an encrypted drive or partition, that Real Time Monitor window will **never** reveal any of its user's recognizable data. Everything will just look like monitor static. It will be digital noise because, as we know, the result of high quality encryption is data that is indistinguishable from completely random noise.

So, by all means, Lee, run SpinRite over that encrypted partition and any damaged sectors that SpinRite is able to recover **will** result in recovered files once that encrypted partition is mounted and decrypted.

Listener Feedback

Jason Egan

Steve, I wanted to send along my thanks to you for reminding me of the Tower of Hanoi puzzle! I had forgotten how much I'd enjoyed it as a child. I picked one up for my children (who are 8 and 10) for Father's Day and they're hooked! It makes me proud! Thanks for not only bringing us timely and informative security news, but also for interesting and fun things like this! I appreciate what you do every week. / Jason

Jason, thanks for your note. I appreciate it, and all of the feedback our listeners take the time to send every week. A surprising number of our listeners mention that the podcast also makes them laugh. I assume that's mostly thanks to the picture of the week, though every so often we do encounter things in the security space that are so absurd as to be ridiculous and funny. When it's occasionally possible to inform and entertain at the same time, that's a win.

Brian Tillman

What I'm curious about is how a newspaper can claim that an LLM's users are reading data that's supposed to be behind a paywall. If there's a paywall in place, how are the LLMs gaining access to that material? Doesn't seem like a very good paywall to me.

I think that's a really good point. Many sites, like coding forums, will have huge historical depth of code that could be plumbed. So once an LLM got in there and sucked out its content it's game over. The information has escaped. But the point of a news site is that it's news. So to Brian's point, although it may not have been clear several years ago, today's LLMs have learned that they MUST legally abide by robots exclusion rules and not traverse into any sites that have explicitly banned their entry. And I'd be pretty sure that's no longer happening.

Ze'ev

Hi Steve and Leo, My name is Ze'ev, and I'm a longtime listener of the Security Now podcast. Your show is fantastic, and I'm glad to hear there's no definitive end date in sight! :-)

Recently, like many others, I've become interested in AI. As part of a hackathon at work, I explored the LlamaIndex Python library, which can be used (among other things) to easily create a Retrieval-Augmented Generation (RAG) system. That immediately made me think of the incredible podcast transcripts that Elaine Farris has created for your show. I realized it might be an interesting opportunity to use AI to gain insights from this amazing trove of data. I ended up writing a simple Python program (about 130 lines of code) that lets you run a query or prompt and get an answer based on the transcripts. After some experimentation, I found that indexing and querying the transcripts by year yields the best results. Below are a few examples of prompts I tried and the responses I got from the LLM.

The code is available on GitHub here: <https://github.com/zeev-b/sn> Thank you for all the work you put into the podcast — it's been an invaluable source of knowledge (and entertainment!) over the years. / Best regards, Ze'ev

Okay. Ze'ev shared his query and the LLM's reply about our most recommended science fiction reading material and our most recommended television shows. But his third query and the LLM's response was the most interesting to me and, I figured, would also be interesting to our listeners. What Ze'ev asked his transcript-trained LLM was: *"How has Steve's opinion on Microsoft security changed over the years?"* and the LLM's reply was:

Steve's opinion of Microsoft security has evolved significantly over the years, reflecting a complex interplay of recognition of improvements, growing skepticism, and critical concerns about the company's practices.

In the early years, particularly around 2015, Steve acknowledged Microsoft's efforts in managing security, noting that the company had developed defenses against constant attacks and was not frequently in the news for major breaches. He appreciated the improvements in security but remained cautious about privacy implications, especially with Windows 10's design.

By 2016, his perspective began to shift towards a more critical stance, particularly regarding Microsoft's handling of updates and the promotion of Windows 10. He expressed concerns about the alignment of Microsoft's interests with those of users and highlighted issues such as the lack of transparency in updates and potential security risks associated with third-party antivirus software.

In subsequent years, Steve recognized improvements in built-in security measures, particularly with Windows Defender, but he also emphasized the ongoing challenges in the cybersecurity landscape. By 2018, he began to advocate for the use of built-in security solutions over third-party options, acknowledging the complexities of maintaining security in a rapidly evolving environment.

However, as time progressed, his concerns deepened. By 2019 and into 2020, he expressed frustration with Microsoft's inconsistent handling of vulnerabilities and updates, particularly regarding support for older operating systems. He highlighted the need for users to be proactive about security and criticized Microsoft's approach to updates, which he felt often led to significant vulnerabilities.

By 2021 and 2022, Steve's skepticism intensified, as he noted a perceived negligence on Microsoft's part in addressing known security issues promptly. He pointed out that the company seemed to prioritize other aspects over security, leading to a growing concern about the effectiveness of their measures.

In 2023, his criticism became more pronounced, focusing on Microsoft's reliance on updates and the prevalence of known vulnerabilities in their products. He expressed frustration with the company's failure to address these issues proactively and questioned the ethics of their approach to forcing upgrades.

By 2024, Steve's opinion had crystallized into a broader skepticism about Microsoft's commitment to security, particularly in light of high-profile security failures and a corporate culture that seemed to deprioritize security investments. He criticized the company's reactive approach to vulnerabilities and argued for the inclusion of essential security features in base products rather than as paid add-ons.

As of 2025, Steve's view reflects a recognition of the inherent complexities in securing Microsoft's systems, acknowledging that while improvements have been made, the legacy nature of their software creates ongoing vulnerabilities. He advocates for additional protective measures when using their products, emphasizing the need for caution and accountability in Microsoft's security practices.

Overall, Steve's evolving opinion illustrates a journey from cautious optimism to critical skepticism, highlighting the need for significant changes in how Microsoft approaches security to better protect its users.

We are living through a truly astonishing revolution where we are witnessing the emergence of a technology that's able to take transcripts of 20 years of my rambling and turn them into that.

Internet Foreground Radiation

As I mentioned at the top of the show, today's podcast title "Internet Foreground Radiation" is a play on the term "Internet Background Radiation", which I coined 26 years ago back in 1999, while developing GRC's ShieldsUP! facility and was observing the random packet crap and noise that would occasionally flow to any Internet IP address.

Wikipedia reminds us that:

Cosmic background radiation is electromagnetic radiation that fills all space. The origin of this radiation depends on the region of the spectrum that is observed. One component is the cosmic microwave background. This component is redshifted photons that have freely streamed from an epoch when the Universe became transparent for the first time to radiation. Its discovery and detailed observations of its properties are considered one of the major confirmations of the Big Bang.

Fortunately, unlike the cosmic background radiation which will presumably never die, the original designers of the Internet had the foresight to place a "time to live" counter into every single packet that moves across the Internet. And the very first thing that every Internet router does, when it receives an incoming packet, is to decrement that packet's remaining "time to live". If that packet's time to live was '1' and is decremented to '0', that signals that the packet has been alive long enough, that if it was ever going to reach its destination it should have by now, and that for the sake of the greater good of the Internet, it must now be put to rest. Sorry, packet. When this occurs, well behaving Internet routers will see where that packet came from, from its source IP address and will send back an ICMP "time exceeded" message to inform its sender that the packet it sent to whatever destination IP never, for whatever reason, reached that destination.

My point here is that unlike true cosmic background radiation, Internet packets are strictly not allowed to wander around the Internet, forever aimlessly. And what this, in turn, means is that all Internet background radiation has a deliberate source somewhere and that any time a packet is received, someone somewhere deliberately formed it and dropped it onto the Internet.

Now, that said, that "someone somewhere" could be some cranky old and forgotten NT server in a locked and forgotten closet that became infected by the Code Red or Nimda worms of 2001. Those were the good old days, where Code Red, for example, was a flash worm that successfully infected more than 350,000 Microsoft IIS web servers within a few hours of its launch onto the Internet. So if any skeptics might be wondering whether things have gotten better through the intervening years, the answer is certainly yes. We do seem to be well past the point of flash worms taking down the Internet. But the presence of any mono-culture should always make any prudent person nervous, since mistakes can always happen.

My point is that even today, even though Internet packets will never persist on the Internet, true Internet Background Radiation, being emitted from dusty servers in lonely locked closets may still exist. So the reason I named today's podcast "Internet FOREGROUND Radiation" is that there's something else going on that an Internet security firm has been observing. The distinction I wanted to make is that whereas "Internet BACKGROUND Radiation" – much like the cosmic background radiation – lacks deliberate intention, there is now separate Internet Foreground Radiation, behind which a great deal of deliberate intention lies.

So who's generating THAT radiation, and why?

Last week, the firm "HUMAN Security" posted the results of their long-running research under the heading: *"Opportunity Makes the Thief: How Web Scanner Bots Target New Websites for Cyber Attacks"* ([Research Link](#)) Since what they found is not something that I think has ever been known or appreciated before, I thought it would be worth sharing and taking a closer look at. They introduce this subject by writing:

When a new website goes live, it's not people who visit first—it's bots. Automated tools probe new domains within minutes, long before any customer or legitimate user arrives. These bots vary widely in intent: some are benign (search engine crawlers indexing your pages, or commercial security scanners checking for vulnerabilities with permission), but many are malicious. Among the most pernicious are web scanner bots, which quickly examine websites for weaknesses and exploit them, turning reconnaissance and attack into a single automated sequence, carried out at scale.

HUMAN Security's Satori Threat Intelligence team monitors bot activity across our customer network and in dedicated research environments. One such environment is a honeypot: a web server we intentionally set up to attract only bot traffic. By observing the requests hitting this fresh, otherwise unpublicized website, we are able to gain insight into the types of bots that target a website from its inception and onwards. One early finding in this experiment was that web scanners consistently dominate early traffic to new sites, and continue to probe the sites day after day, long after other bot types began to appear. This persistence underscores why scanner activity is a security concern for the full life span of any web property.

This blog post examines the threat of these web scanner bots and shares our recent research findings on several active scanner campaigns (including the Mozi.a botnet, a Mirai-variant called Jaws, and the "Romanian Distillery" scanner).

Web scanner bots are an escalating cyber threat: These bots are often the very first visitors to any new website, probing for security weaknesses long before any human users arrive. On a newly launched site observed by HUMAN, scanners made up an average of ~70% of all bot traffic in the first days, and on some days 100% of detected bot visits were from scanners.

Botnet-driven scanning operations are growing more complex: The "Romanian Distillery" operation is a prime example: once focused solely on harvesting SMTP credentials, it now probes for PHP services, .env files, and misconfigurations across a distributed /24 subnet. Its scan patterns follow doubling revisit intervals and reveal a coordinated infrastructure designed for scale. In some cases, such scanners don't stop at discovery—they attempt SQL injection or deploy malware immediately after identifying a weakness.

Traditional defenses struggle to catch web scanners: Many of these scanner bots evade simple security measures. They may rotate through networks of infected devices using other botnets to distribute their scans, hide their true identity by omitting or faking their user-agent strings, and rapidly change tactics to avoid signature-based detection. Legacy security tools that rely on known malicious IP lists or obvious signatures often miss these stealthy probes.

Web scanners, also known as website vulnerability scanners, are automated tools designed to identify security weaknesses in web applications, websites, and APIs. They systematically probe and inspect websites for misconfigurations, exposed files, default credentials, or known vulnerable software. If a new website is still being configured or lacks proper protections, scanners will attempt to find and exploit any flaw they can, before the site's owners have a chance to secure it.

So the first interesting and crucial takeaway would be to never assume that any security can be added after any portion of a new site goes live. Never assume that since it hasn't been advertised or announced or in any way made public, that it might be at all safe to place anything online that isn't already fully hardened. Essentially, the entire Internet has become a loaded and cocked mouse trap ready to spring and capture at the slightest provocation.

This is where my favorite of all tricks – IP-based filtering – can come in handy. It's so simple and so absolutely robust as a security solution. If you wish to create some true external exposure, simply first block all access, then selectively allow the IPs through that you know you can trust. But never open the flood gates until you're fully prepared to be deeply attacked.

They wrote:

Not all scanners are bad, though. There are "good" scanners run by security companies or researchers to help site owners by identifying vulnerabilities so they can be fixed, before those "bad" scanners get to them. But both good and bad scanners impose load on your site, and the bad ones, if not blocked, will certainly attempt to leverage any weakness they find. Scanners don't just visit once; they constantly and persistently re-scan sites over time (since new vulnerabilities might appear with site updates or as new exploits are discovered).

It's always fun to run across something we've never touched on or talked about on this podcast. Given that this is our one thousand and thirtieth podcast, we've logged many thousands of hours discussing and covering pretty much anything and everything that has happened over the past 20 years. So it's not often that we encounter something we've never discussed before.

But today is one such rare day, because the ubiquitous shift to TLS HTTPS website connections, which increasingly require the use of SNI – server name indication – to be specified and provided by the connecting client in its first TLS handshake packet, means that knowing the IP address of a site, or having a site's IP address simply by scanning them all, is no longer sufficient to successfully establish a completed handshake to a site's servers.

Think about that for a second, because we never have on this podcast. From the birth of the Internet, it has been possible to simply scan the Internet's 32-bit IPv4 space for web servers and to establish connections to them. But that all changed once the likes of Cloudflare and other CDN's came along. As a result of IPv4 space depletion and the economic fact that IP sharing is inherently less expensive since it also allows for infrastructure sharing, more than 90% of today's websites are now sharing IP addresses, leaving fewer than 10% of all sites sitting on a dedicated IP address. This means that more than 90% of the Internet's websites have migrated behind proxies that are only able to disambiguate website destinations by examining the incoming client's SNI extension field.

This even applies to a modest facility like mine. GRC's .sc shortcut server, our mail, dev, sqlr, gitlab servers and others all share the same IP. So it's not possible to reach any of them simply by their IP. It's necessary for anyone who wishes to connect to any server behind that IP to somehow also and first know which domain name they expect to find at which IP address. So, as I said, that's not an observation that's ever come on on this podcast before.

However, I wish this presented a bigger problem for web scanners than it does. The guys at HUMAN Security explain what scanners' response to this has been, writing:

To ensure that their scanners manage to get first in line for any new website, threat actors

take advantage of feeds and services that announce new websites or domains coming online.

For example, threat actors monitor Newly Registered Domain (NRD) feeds, which are lists of recently registered, updated, or dropped domains, such as WHOIS databases and domain drop lists. Such NRD feeds are re-purposed from policy feeds intended to increase corporate security to threat intelligence and monitoring feeds against the websites themselves.

Threat actors also monitor certificate transparency logs, such as CertStream, which publicly log new TLS certificates. For scanners, a new domain registration or certificate issuance indicates a new website that could be scanned. Once the large-scale SEO crawlers index the new website, scanners may also monitor the search engines' new listings, and the scale of scanners will increase even further.

For the purposes of this research, these guys are interested in identifying and, where possible, disambiguating and classifying the range of bots that are probing their honeypots. We know this doesn't matter for the sake of security, since for security it's necessary to simply be equally secure for anyone who might come knocking. But what these guys found was intriguing and revealing. Under the heading *"Identifying Web Scanner Bot Traffic"* they wrote:

Some scanners openly identify themselves in the user-agent string, which is the part of an HTTP request that might say, for example, "Mozilla/5.0 (compatible; ScannerXYZ/1.0; ...)". Security teams can easily filter or block those known scanners. But many malicious scanners deliberately mask their identity or use misleading user agents to obfuscate their true nature. In these cases, identifying them requires analyzing their behavior and deploying antibot mechanisms to intercept their activity.

Some user agents we observed suggest the presence of outdated or anachronistic systems including BeOS, legacy Linux kernels, and even Windows 3.1 Internet Explorer versions. Obviously, no real users are surfing the web on Windows 3.1 today, so this was a dead giveaway of automated activity. These impossibly old user agents likely came from a public user-agent database that the attackers grabbed for obfuscation purposes. A fun and benign find that should never hit any of your web servers if you have a decent bot mitigation solution deployed.

Mozilla/1.22 (compatible; MSIE 10.0; Windows 3.1)

Mozilla/1.22 (compatible; MSIE 5.01; PalmOS 3.0) EudoraWeb 2.1

Mozilla/2.0 (compatible; MSIE 3.03; Windows 3.1)

Mozilla/2.0 (compatible; MSIE 3.0B; Windows NT)

Mozilla/2.0 (compatible; MSIE 4.0; Windows 98)

Mozilla/3.0 (compatible; NetPositive/2.1.1; BeOS)

Mozilla/3.01Gold (Win95; i)

Mozilla/4.0 (PSP (PlayStation Portable); 2.00)

Mozilla/4.0 (compatible; Dillo 3.0)

Mozilla/4.0 (compatible; Linux 2.6.22) NetFront/3.4 Kindle/2.0 (screen 600x800)

Mozilla/4.0 (compatible; MSIE 4.01; Windows CE; PPC; MDA Pro/1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1)

I particularly like the *"Mozilla/1.22 (compatible; MSIE 5.01; PalmOS 3.0) EudoraWeb 2.1"* there's some blasts from the past! So I would be inclined to agree with their assumption about the likely source of those bogus bot User-Agent strings is likely correct.

Under the heading *"Reconnaissance and Probing the Target"* they explain what these scanners tend to do once they locate a new candidate target. They write:

Before scanners are even deployed, operators conduct manual reconnaissance to identify likely entry points—directories, configuration files, endpoints, and services that may exist at newly launched sites. They then craft scanners with predefined paths and exploitation logic, tuned to probe and attack if those elements are present and identified.

Once launched against a site, the scanner rapidly tests for these known targets. It attempts to enumerate directories, pages, API endpoints, and exposed resources, executing preset payloads or exploits where applicable.

One of the most common methods to launch scanners is using 'Dirbust', a dictionary-based attack against web servers that automates the process of discovering hidden files and directories on a website. This tool scans through predefined lists of potential directory and file names (e.g. /admin/, /config.php, /backup.zip, etc.) in hopes of getting lucky in finding unprotected sensitive files or admin interfaces.

Here again, having been a website admin myself for the past 25 years, it can sometimes be tempting to imagine that it might be possible to just briefly do something that's not entirely secure under the assumption that just for a few minutes no one will notice. (You know, don't worry, I'm just going to put the tip in!) All I can say is that on today's Internet, doing anything like that is risky at best.

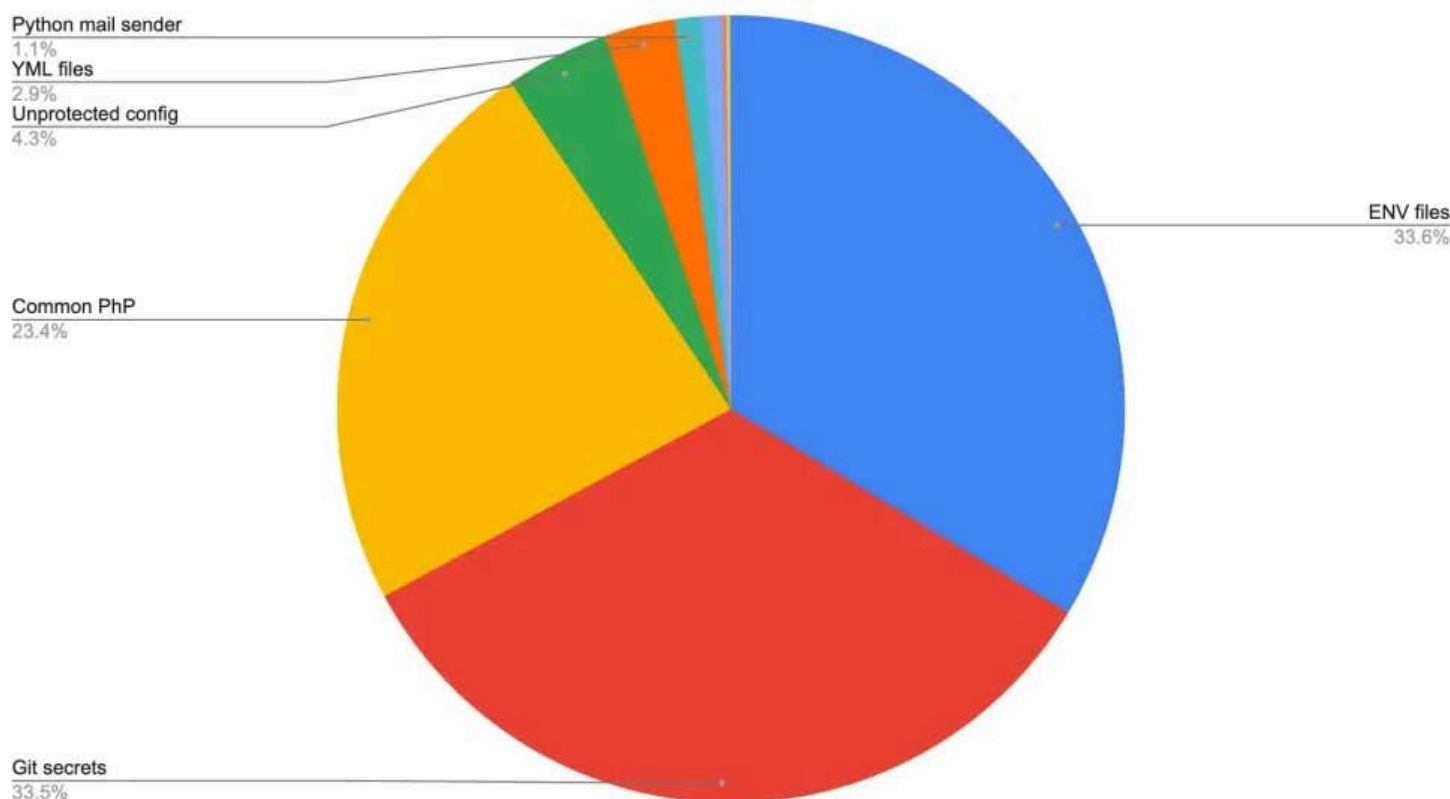
I've always had better things to do than wonder and worry about what percentage of GRC's inbound connection traffic has malicious intent, but, for example, I've seen ample evidence of tools like the *"Dirbust"* they mention in the logs that I sometimes briefly enable when I'm trying to track down some specific behavior. The only way to be safe is to assume that it's all malicious and be prepared for that. GRC's servers do not log website activity specifically because the signal-to-noise ratio is so low that there's virtually no signal among the noise. That's today's reality. It really has become a nasty jungle out there. It's sad that's happened, but it has.

They wrote:

*Using the scanner traffic from our research, we mapped the most targeted path types targeted by scanners. This mapping shows that scanners have particular favorites when it comes to these initial probes. The two most targeted types of files **by far** were environment configuration (.env) files and repositories for code secrets. In fact, about one-third of all scanning attempts in our study were after .env files, and another one-third were looking for Git repository data (such as .git folders or leftover export files).*

This is no surprise, environment (.env) files often contain API keys, database passwords, and other secrets that would be a jackpot for an attacker, and Git-related files might expose source code or credentials that enable a deeper compromise. The potential exploitation from each of these path types is listed in the table below and the most targeted paths are shown in the chart.

Scanner Activity by Path Type Count



The pie chart I've copied from their report into the show notes shows that fully 33.6% of webserver file type requests were for .ENV files. And the effectively equal 33.5% were for GIT secrets. Next up at 23.4% were common PHP files. Collectively, just those three account for a whopping 90.5% of all website probes. This leaves 4.3% for unprotected config files, 2.9% for YML files, and 1.1% for Python mail sender files.

The table that they mentioned contained a few interesting comments:

Path Type	Explanation
.env files	Stores API secrets, tokens, and other sensitive information that can be used by attackers to pivot and attack.
Git secrets	Used to gain access to victims' repositories, leading to cross-organisation compromise.
Common PHP	Useful when doing recon, identifying the versions of PHP's or other installed frameworks.
Unprotected config	Config files that can contain any kind of sensitive information, from users, passwords, and environment variables that can be used in follow-up attacks.
YML files	Same as .env files, contains API secrets, users, tokens, and other sensitive information that an attacker can use to pivot.

Python mail sender	In poorly configured web servers, attackers can gain access to the credentials used to send mail en masse.
Exchange server	Vulnerable Exchange servers that can lead to further compromise of the organization and facilitate attacker lateral movement
Server status	Discloses paths, user agents, and IPs visited by everyone on the server. Insecure web apps might disclose various PII (through their path) and also provide attackers precious data to do additional recon, such as internal API endpoints.
Laravel	Popular PHP framework with various RCEs that can be further exploited by attackers. Attackers fingerprint it by scanning for unsecured default endpoints that provide version info.
WordPress	Popular PHP CMS with a history of vulnerabilities. Attackers try to find fresh versions that are still using default credentials or endpoints that provide additional fingerprinting, such as plugin versions. These are further scanned for vulnerabilities and exploited if found lacking the latest security patches.

They wrap-up the topic of reconnaissance and probing by adding:

Beyond those, scanners also commonly seek out various configuration and backup files (for example, .yml or .yaml configs, old .bak or .zip backups of the site, or files like config.php that might reveal database connection info). They probe for known software-specific files: for instance, requesting a URL ending in wp-config.php could indicate the site uses WordPress and reveal its config if not secured, or hitting /server-status on a web server could reveal internal information if that page isn't locked down. Scanners will even check for well-known vulnerable services – one example is scanning for Outlook Web Access/Exchange server paths on sites, since unpatched Exchange servers are high-value targets that could lead to a broader organizational breach.

Essentially, during the probing phase, scanner bots test the site against a predefined list of files, directories, and endpoints that should not be exposed. Every directory listing, config file, or version disclosure it finds is "loot" that can be used in the next phase of the attack. This process is highly automated and aggressive: the scanner might attempt hundreds of different URLs on your site in rapid succession, far more exhaustive and faster than any human could manage. That behavior pattern is often a telltale sign that the traffic is a scanner bot.

When you stop to think about it, there's only one reason any of this exists and any of this is worth the time and trouble on the part of the attackers: All of this technology we're using today contains a very long legacy of being insecure by default.

For example, although this characteristic is finally beginning to disappear, historically, it has been entirely possible and was even once acceptable to choose NOT to use any password at all when setting up a new operating system or device. UNIX and Linux once allowed the ROOT user's password to be null.

Today we all recognize this as bad, but we all probably also recall a time when we did that ourselves.

In the future, the option to skip a password will not exist, no one will believe it was ever possible, and they'll understand that doing so would be insane.

Once upon a time it was not so insane. But that has been entirely upended thanks to the Internet's steadily growing **foreground** radiation.

