A Nefarious Novel Use for AI

Description: The "bone crushing" didn't happen this month. Revisiting and inspecting July's Patch Tuesday. A widespread and worrisome flaw in OpenSSL. Claude can now access your 1Password credentials. Bitwarden is aware that we need whole new security. The day ends in "y" so a new prompt injection attack. A true (and rare) core Wordpress emergency update. Lots of interesting listener feedback. And new ways AI is being used by bad guys.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-1088.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-1088-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Man, there's so much to talk about. A big flaw in OpenSSL. 1Password and Bitwarden try to solve the agentic AI password crisis. A new prompt injection attack AI users should be aware of. And a new way people in the bad guy profession are using AI, kind of makes sense, all that. Plus a great Picture of the Week coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 1088, recorded Tuesday, July 21st, 2026: "A Nefarious Novel Use for AI."

Time for Security Now!. Hello, everybody, boys and girls, children of all ages, it's time for this guy right here, Mr. Steve Gibson, our security guru. Every Tuesday we gather together, sit at Steve's feet and learn about the perils that we are suffering here in this modern world. Hello, Steve.

Steve Gibson: The perils of remaining plugged in on the grid.

Leo: Yeah. If we were all air-gapped, we'd be okay.

Steve: Well, kind of.

Leo: Or would we?

Steve: You know, Stuxnet managed to jump an air gap.

Leo: That's a good point.

Steve: So even so.

Leo: Don't pick up USB keys in the parking lot, boys and girls.

Steve: Ooh, no. No, no, no. We are at Episode 1088 for this July 21st. Our title is "A Nefarious Novel Use for AI."

Leo: Oh, my.

Steve: There are actually two. We've talked about the way bad guys are in an arms race with the good guys in finding vulnerabilities which they could then exploit in order to get into systems. It turns out that that's actually not one of the leading uses, the malicious mal uses, malign uses of AI. It turns out that getting into networks is not that difficult, sadly. It's like they don't need anything more. They've got so many ways in now, it's just a matter of eenie, meenie, miney, mo.

Leo: Oh, lord.

Steve: Anyway. But they've come up with a use for AI after that, which is novel and nefarious. So we'll be talking about that this week. But first we will look at the fact that the bone-crushing we had been promised did not happen this month. Not much did from Nightmare Eclipse. We are going to revisit and look more closely at last week's, which is to say July's Patch Tuesday. A widespread and worrisome flaw has been uncovered in OpenSSL. And OpenSSL jumped on this, quietly fixed it, pushed out changes. The problem is it is so widespread that there's no chance it's going to get fixed everywhere. And we'll look at the consequences of that.

A bunch of our listeners said, "Uh, Steve, I just saw an article saying that Claude can now access your 1Password credentials." And this is where we deploy the "What could possibly go wrong?" Also Bitwarden is aware that we need a whole new kind of coverage of security. We're going to look at that. Also the day ends in "y," so we have a new prompt injection attack. There has been a very rare, very serious update for WordPress which, unlike previous, where it's in some random add-on that five people in Milwaukee have installed, in this case this is in the core.

Leo: Uh-oh. That's not good.

Steve: So I hope everybody - no, it's really bad. I hope everybody who is staying up-to-date with WordPress, I think it was introduced in early December of 2025, so it's been around for about six or seven months, and it's significant. We've got also lots of interesting listener feedback. I made time for it because I just - we haven't had as much as I've wanted recently. And then we're going to look at the new ways AI is being used by bad guys after they get into someone's network. And it's not, again, it's not sort of the techie side. It's not, you know, better exploration of the network. It's interesting.

Leo: Hmm.

Steve: Of course we've got a fun Picture of the Week. So, yeah. I think worth tuning in for Episode 1088.

Leo: If only to get new ideas on how you can exploit people's networks, it's worth it, using AI. We will have that Picture of the Week in moments. All I know is it has something to do with coffee. I'm excited by anything to do with coffee.

Steve: And now you've got my attention.

Leo: Yes, me, too.

Steve: I gave it kind of a lame title. I said, "Because coffee is life." Because, you know.

Leo: It is.

Steve: It is.

Leo: It is.

Steve: But anyway, it's a fun sign that we will...

Leo: I've gotten really into what they call "pour over," which is really the silliest kind of coffee making where you're using a filter, and there's all sorts of steps, and the different grinds and all this stuff. It's not expensive. That's the only good thing. The bad thing is it's very time-consuming, and it's chemistry, and there's a lot of books and stuff. But it's so good.

Steve: A lot of books and stuff.

Leo: There's a lot of reading. It's chemistry. Like you should see this book that I have. It's got all of these - it's by a physicist. It's called "The Physics of Filter Coffee."

Steve: It is possible to create a stunning cup of coffee. I mean...

Leo: You really can. Because there's different bottles, different chemistries, the time, the temperature of the water, the size of the filter, all of this stuff makes a huge difference. The size of the grind. And so you can - lots of dials to turn. But once you get it right, it's really good. And I've never drank coffee black before. But because you can really make it to your taste, now it's like it's my reward. I say you can't have it till you work out. And that's what I've got here. And it's a bad thing. So

coffee is life, is what I'm saying. I'm agreeing with you 100%. Okay. Picture of the Week.

Steve: So what we have underneath this "Because coffee is life" title...

Leo: Which it is.

Steve: ...is a sign that someone took a picture of which describes itself as the - it's wonderful - the No Nonsense Coffee Guide.

Leo: This is like on a chalkboard outside a coffee shop.

Steve: Yes, exactly. This is like, okay. So it shows on the left are the fancy, you know, French or Italian terms for some random yuppie coffee. And then in the right is the equivalent. So we have the Americano, which has been crossed out; and then next to it, it says Black Coffee. Flat White crossed out, White Coffee. Cappuccino crossed out, Frothy Coffee.

Leo: Good, good.

Steve: Latte, Milky Coffee. Espresso, Miniature Coffee. Macchiato, Milk Topped Coffee.

Leo: Yeah, just a little dab, yeah.

Steve: Mocha, Choccy Coffee.

Leo: C-h-o-c-c-y, yes.

Steve: Choccy, yes. Tea, Not Coffee. And Hot Chocolate, Still Not Coffee.

Leo: Oh, I want to go to this place. Oh, that is so true.

Steve: Yes, it's just like, okay, fine, we'll give you your milky coffee. That's right.

Leo: No AI here. That's - somebody handwrote that one on a chalkboard, that's for sure.

Steve: That's right. So, and I noticed there's a pound sign, it says #Pubonthehoe, H-O-E.

Leo: Must be the name of the place, Pub on the Hoe.

Steve: That's my guess, yeah.

Leo: That's a weird name.

Steve: Like some farm theme somewhere. Okay. So...

Leo: Oh, that kind of hoe. Okay.

Steve: H-O-E.

Leo: The farm implement, yes, okay.

Steve: That's right. So it's difficult to know exactly what's going on with our prolific and talented, there's no discounting that...

Leo: Oh, it's in the United Kingdom in Plymouth.

Steve: Ah.

Leo: The Pub on the Hoe. I have found it. It exists.

Steve: And they have a no-nonsense coffee sign.

Leo: I love it. Sorry, go ahead.

Steve: Yeah. So it's difficult to know exactly what's going on with our prolific and - there's no discounting it - talented Microsoft-taunting hacker who calls him or herself Nightmare Eclipse. Remember seven months ago this hacker, who's given us a run of zero-days, they warned that a "bone-crushing" vulnerability and exploit proof of concept would be disclosed this month. Presumably timed, as they have all previously been, to maximize their unpatched exposure interval by landing them on successive months' Patch Tuesdays. And we said last Tuesday, on Patch Tuesday, well, hello, where is this?

We did, however, indeed get another one last Tuesday, though it falls far short of bone-crushing. I'm not even sure it would be considered "bone-chipping." It will certainly be "Microsoft-annoying," however. But it's probably also "Microsoft-Relieving," since it amounts to a rather limited-use elevation-of-privilege vulnerability and exploit.

Nightmare Eclipse gave this zero-day the name "LegacyHive." "Hive" is what the Windows Registry blobs are called, you know, it's this hive and that hive. So the hacker called this LegacyHive. And with limitations, it allows attackers to escalate their privileges on currently, like, fully patched Windows systems, right up to date. But then here's where things get weird. Nightmare Eclipse claims that they deliberately "toned down" the proof-of-concept to make it less annoying for Microsoft.

Okay. The story is that while it exploits a security vulnerability in the Windows User Profile Service, it's been modified to require a - that is, the proof of concept has been modified to require a standard user's credentials and another username, like an admin account name, in order to make its exploitation more difficult for attackers to weaponize. And what I find suspicious about this is this does not sound like the Nightmare Eclipse, I mean, it's not such a nightmare; right? So if this is true, which I think I tend to doubt, it would appear to represent a change of heart, like a capitulation on Nightmare Eclipse's part, and I wonder if it could be the result of Microsoft's saber rattling, getting a little, you know, hitting a little too close to home.

The hacker wrote: "The proof of concept requires another standard user's credentials and a third username, which can be an admin account. If the proof of concept is successful, it will end up mounting the target user hive in the current user classes root. The proof of concept was stripped down as an attempt" - get this. "The proof of concept was stripped down as an attempt to prevent public exploitation. The original proof of concept did not require additional user credential and was not limited to usrclass.dat hive. Any hive could be loaded using this vulnerability, but you would need some brain cells to make the proof of concept do it."

Okay. So the industry's security researchers were quick to confirm the vulnerability's proof-of-concept, that is, it works. It does what they say. And Microsoft replied with their standard uninteresting bureaucratic boilerplate which we've seen every time before. So we can now add LegacyHive, this latest one, to RoguePlanet, BlueHammer, RedSun, YellowKey, GreenPlasma, MiniPlasma, and UnDefend, all of which Microsoft has patched the month following or sooner, and many of which were seen being quickly taken up and used by real attackers to actually injure real Windows users and their networks. So maybe there's some built on Nightmare Eclipse's part. Maybe that's the reason, the real injury that this hacker was causing to innocent Windows users, that they decided to make the vulnerability less easy to quickly abuse.

On the other hand, maybe they couldn't make it stronger. You know? I have no basis for that speculation beyond, I guess, just my faith in human morality since, you know, this campaign that Nightmare Eclipse has been waging was also really hurting Windows users.

So anyway, it's unclear whether this is the "bone-crushing" exploit that Nightmare Eclipse promised. It would be really bad if it were possible to arbitrarily load various registry hives, like, into different user profiles. That could be used for all kinds of problems, especially in any kind of a server scenario where it could be devastating. If the restrictions on the use of its proof of concept were lifted, so that, as I said, Windows Registry hive remapping could be performed without any a priori knowledge of the victim's system, then that would have been a real bone-crusher. So on the other hand, we've previously seen Nightmare Eclipse clearly and deliberately exaggerating their capabilities in the past, referring to them saying, oh, there's a way to bypass the PIN on the BitLocker bypass exploit. We know now there was no way to do that. So that was an exaggeration. Maybe this is that, too.

Microsoft knows because they'll see what the problem is that this represents when they go about fixing it and see whether or not it actually could have been a lot worse had the hacker wanted it to be. But in any event, no bones were crushed, or chipped, or very much disturbed this month. So, and apparently it's just not easy to get the proof of concept to do anything very significant. So are they done? Are we going to see something next month? I guess we'll need to stay tuned.

But speaking of this month, and next month, last week we were only able to touch on the release of July's Microsoft patches since they occurred as we were recording the podcast. And Leo, you were able to give us the overview...

Leo: The count. Oh, man.

Steve: ...of, yeah, 570, three zero-days among them. So scanning down the seemingly endless and astonishing list of security - I mean, really, consider scrolling your browser down 570 individually enumerated and described problems, bugs, security vulnerabilities that were fixed. It really is something to see. So it occurred to me that now we would not only need AI to find those, we would be needing an AI to help us keep track of them because, wow, I mean, it is astonishing.

There's almost too much to cover here in detail, and I'm not going to try. But I want to sort of hit the highlights here. Among the record-breaking, by a large margin, 570 security vulnerabilities, 59 of those 570 were rated Critical, given Critical ratings by Microsoft. 48 of those 59 allowed for remote code execution. So we had in one month 48 of the 59, 48 Critical out of a total of 59 Critical were remote code execution vulnerabilities out of a total of 570. So more than one in 10. Another nine broke out of Windows privilege management to allow attackers to obtain system privileges.

Overall, independent of the ranking of the vulnerabilities, you know, like Critical, Moderate, Information, so forth, 145 of the 570, which puts it at more than 25%, overall enabled remote code execution one way or the other. So 48 were Critical RCEs, but the balance to bring a total to 145 remote code executions one way or the other. And also 254, bringing it to 45% of those 550 total were privilege elevation attacks that would allow an attacker who had obtained a minimum foothold in a system to bump up their privileges to full root system access. Which they pretty much need to do in order to do anything extra nasty. And also in order to obtain long-term access to the system.

So it seems to me that the one thing Microsoft is not doing, based on what we're seeing, is restricting their rate of discovery and disclosure. They're not, like, dribbling these out. Each of the past three months has broken their all-time previous security vulnerability patch record, and each time by a significant and significantly growing measure. So it's accelerating in addition to being continually record-breaking. So this makes me extremely interested, I mean, I cannot wait to see what next month will look like. And I heard you saying, I think it was on your Sunday podcast, you mentioned to the two co-hosts with you, Leo, that I had been expecting that we would see increasing numbers of patches, followed by decreasing numbers of patches as the available pool of things to fix dry up.

Leo: Yes.

Steve: Inevitably that's going to happen.

Leo: I've been raising that with everybody, that Steve Gibson says, eventually, we'll get to [crosstalk] zero, but we'll get to fewer, far fewer.

Steve: Well, the thing that they missed that I'm factoring in also is that AI will be in the code design path in the future. I expect, I mean, there's no reason to release a bug that your AI is able to find later. Why not find it first? I mean, find it pre-release?

Leo: Right.

Steve: So that's the other thing that's going to happen. It's the reason I think we're going to be dropping, if not to zero, to like a whole different level, enough low - so low that things like Pwn2Own and HackerOne and bounties and so forth, they're just going to go away because...

Leo: I think that could happen very soon, to be honest.

Steve: Yes. Yes.

Leo: And, you know, I think that's part of the development cycle now. It is certainly when I'm developing with vibe coding.

Steve: Why, exactly.

Leo: Inevitably I do a security audit as I'm going, let alone at the end. I mean, [crosstalk].

Steve: And it must be that Microsoft is already using AI to write code.

Leo: Yeah. Yeah.

Steve: Why would they be lagging there?

Leo: Right. I would, I mean, goodness, yes. And it catches all the obvious things, you know, the buffer overflows, the writing to ring 0, all the - maybe what I said on Sunday, you know, Rowhammer isn't going to go away, probably. Right? That kind of...

Steve: It can't. Yes, not all security problems are code errors.

Leo: Right.

Steve: You can also [crosstalk].

Leo: I mean, microcode will be better, so maybe you won't have those kinds of pipeline errors where it's leaking...

Steve: Well, but you could still have an open port.

Leo: Right.

Steve: You could still have a dumb password.

Leo: Exactly. Exactly. Nobody's going to stop that.

Steve: No.

Leo: That's forever.

Steve: One whole big class of problems is probably going to go away. And I think it's going to - at the rate we're seeing this being jumped on, again, I'm just - I can't - I am so excited to see what happens next month with Patch Tuesday. And I should also mention it's not just Microsoft. All the big publishers are seeing - in fact, Adobe has switched to twice a month updates because their run rate of patches, they are patching so much now, that they thought, okay, we can't wait another three weeks after finding a problem. We need to wait one week and do a mid-month patch. So we're going to see this industry getting cleaned up pretty quickly.

On the other hand, there may be also that unfortunate haves and have-nots bifurcation. Where, you know, the big publishers, the Adobes, the Microsofts, the Oracles, you know, the big guys, Apple also certainly, who are able to just dump all this excess cash they have into token purchasing. They have the ability to do this. Smaller publishers may not, although I just saw Synology updated my boxes for an AI-discovered problem that it had. So even the smaller guys are saying, hey, let's - why not spend some money on some tokens and make our product better? So, wow. The shape of this patch curve is really going to be interesting. My guess is we may see fewer next month. I don't know.

Leo: That'll be interesting. I mean, I think you're right, the velocity will certainly go down.

Steve: Yeah.

Leo: I mean, nobody would deny that. It's just at what rate.

Steve: Right.

Leo: And to what final resting point.

Steve: Exactly.

Leo: Yeah.

Steve: Okay. So HollowByte is the name that Okta gave to their discovery of a very worrisome Denial of Service that exists in OpenSSL. Any problems discovered in the massively used, I mean, like, it's hard to describe how widespread the use of OpenSSL is. There are certainly private TCP/IP stacks. Windows has one. Apple has their own. But

like anything that wants to create a TLS connection now, which is some embedded device or a widget or whatever, it's got OpenSSL.

Now, there are, we've talked about there are some embedded TLS libraries that are used at the really small embedded level. But OpenSSL, as we know, we've been talking about it for decades, is what you use. Apache uses it. Nginx web servers use OpenSSL. The runtime libraries like Node.js, Python, Ruby, PHP, MySQL, they're all using OpenSSL. So because OpenSSL is widely used and embedded, this vulnerability affects all of these systems.

So Okta discovered a means - and it's really sad actually because it's so simple. It's like, really, guys? This is still a vulnerability today? - for sending just 11 bytes of TLS data to any unpatched OpenSSL endpoint, you know, meaning all of those servers that I mentioned and the various application libraries - to cause the connection to over-allocate a memory buffer in anticipation of receiving the remainder of the declared incoming data. So once again, this is why I'm sort of disappointed in this problem. So it's one of those where the header declares how much data follows, and then it doesn't. But because the header is parsed first, the library says, oh, here comes 128K of data. So it pre-allocates a buffer to contain the data. Which then never arrives. It's like, guys, how - really? In this day and age? You're still coming across those kinds of problems?

Anyway, by doing that over and over and over, making a connection, sending 11 bytes, an attacker using very few resources at their end, meaning you don't need lots of servers and lots of bandwidth or anything, some random proxy that exists in some guy's, you know, LG TV that got taken over, to bring down a major service.

Okta provided some background and color which I want to share. They wrote: "Every so often, a vulnerability reminds us how deeply our digital infrastructure relies on foundational libraries. Recently, the Okta Red Team discovered HollowByte, a Denial of Service vulnerability in OpenSSL. By sending a malicious payload of just 11 bytes, any remote, unauthenticated attacker can force a server to allocate disproportionate chunks of memory before any security handshake even begins.

"The TLS handshake begins with a ClientHello message wrapped in a record. Each TLS handshake message begins with a 4-byte header that declares how large the incoming message body will be. Existing versions of OpenSSL allocate a receive buffer based on that attacker-declared length before any data has actually arrived." Like I said, really? In this day and age we're still doing that?

"When the malicious 11-byte payload arrives, the TLS state machine reads the 4-byte handshake header and triggers an unvalidated pre-allocation based on the header's 3-byte length declaration. Because there's no payload validation at this early stage, the system's malloc(), the memory allocator, allocates up to 131K, as I said, 128K binary, based solely on the untrusted packet's claim. The worker thread then blocks, waiting indefinitely for the data that will never arrive.

"Holding connections open to exhaust threads is a classic trick." Like Slowloris that we talked about years ago. "HollowByte introduces a far nastier compounding effect due to how the GNU C Library (glibc) handles memory. When an attacking connection drops, OpenSSL frees and releases the buffer. However, glibc does not immediately return small-to-medium size" - which 128K is considered - "allocations back to the operating system. It retains them for potential reuse.

"Therefore, by launching waves of connections with randomized claimed sizes" - meaning that they're not going to be reused perfectly - "an attacker prevents the allocator from reusing those freed chunks. This fragments the system's memory allocation heap heavily, causing the server's Resident Set Size (RSS) to climb continuously. Even after the

attacker disconnects, the server remains permanently bloated. The only way to reclaim that memory is to terminate the process." You know, shut down the web server, or whatever service is using OpenSSL. Frequently it just means having to reboot the system. You've killed that service.

They said: "To measure the threat, we tested unpatched and patched OpenSSL instances running nginx under various load conditions. In a standard 1GB RAM environment, an unpatched server was Out-Of-Memory killed at 547MB of frozen, fragmented memory. In higher-spec testing" - with, for example, a 16GB RAM allocation - "the attack successfully locked up 25% of the system's total memory while staying safely under the connection ceiling limits, meaning standard connection-limiting defenses won't stop it. The OpenSSL team resolved this by switching" - wait for it - "to an incremental buffer growth strategy." What a concept.

Leo: Incremental.

Steve: What a concept, Leo. You mean you actually don't allocate memory until you get something to put in there? Huh.

Leo: It's amazing.

Steve: Who'd have thought of that? Wow. "This fix was silently included as part of the OpenSSL v4.0.1 release, with silent backports to release versions 3.6.3, 3.5.7, 3.4.6, and 3.0.21. Under this revised memory allocation strategy, rather than trusting the header's claims outright, OpenSSL now grows the buffer only as bytes are actually received over the wire." Wow, a breakthrough. "A claim with no follow-through now costs the server nothing. Even though OpenSSL handled this as a hardening fix rather than a CVE security advisory, we recommend upgrading your distribution's OpenSSL packages immediately." And I'll just put a big amen on that.

I went over to the OpenSSL repository and saw that all of those stated versions were updated more than five weeks ago. This occurred on June 9th. So this would have meant that all of the various dependent packages - Apache, nginx, Node.js, Python, Ruby, PHP and so forth - would have needed to incorporate that update into their own builds and then make those available. Then any public exposure of them would need to be updated and relaunched.

Now, the problem we always have is that those are only the most well-known, prominent, and obvious users of OpenSSL. It is doubtless used in countless other systems. For example, I was curious about my own fully patched and up-to-date Synology NAS. So I SSHed into it and issued the command "openssl version," and I was promptly informed: "OpenSSL 1.1.1u," which was dated the 30th of May, 2023. Since I follow my own advice, my own residential network has exactly zero open ports to the outside world. You just can't have any. And here's a perfect example of why. This was not a problem anybody knew about. They silently patched it and pushed the updates out. Even downplaying it as some hardening, rather than giving it a CVE that would have brought it to the attention of the bad guys because they know how bad this is. I mean, this lets you crash and freeze and lock up any OpenSSL receiving service.

So for random end users, I would say it's unlikely to be much of a problem. It's not going to be the end of the world. But the chances are very good that most, if not every single piece, of enterprise border equipment is also based on a version of OpenSSL which was published more than five weeks ago. So unless your vendor has updated and pushed and

made available updates - and hopefully you didn't wait, because you shouldn't these days, to update your appliance. If any of that did not happen within five weeks, then that's - the systems you're using can probably be brought to their knees. Again, doesn't let the bad guys in, but it lets them shut down your network.

So although Okta didn't disclose whether this newly disclosed vulnerability was found through the use of AI, it is exactly the problem that the entire industry will now be facing. As I've noted, our browsers and operating system have already developed quite mature systems for keeping themselves up to date. But many network appliances have not seen the need to do the same. It's difficult to get a device which isn't asking if there's anything new for it to suddenly start doing that. So in the intermediate term, there's probably going to be a flood of newly discovered vulnerabilities and updates which, you know, users of these systems need to be staying current with. Things exactly like this that need to get fixed. And who knows what else we're going to be seeing in the short-term. Yikes.

What I do know, Leo, we're going to be seeing short-term...

Leo: Is that coffee is life? Cheers. All right. Let's talk about Claude.

Steve: Okay, so one of...

Leo: Or as Paul Thurrott calls it, Claude. It's pretty crazy. Claude.

Steve: One of this podcast's favorite rhetorical questions is, "What could possibly go wrong?" So it's bearing that question in mind that I share this next bit of news that Anthropic's Claude AI is now able to access and use its Mac users' passwords stored in their 1Password vault. Which of course then begs the question, our favorite question: "What could possibly go wrong?" And I'll just note that 1Password is a past sponsor of the TWiT network.

Last Thursday, 1Password posted a blog entry with the headline "1Password for Claude: Give Claude access without giving up your credentials." And, okay, this is the first of two pieces of news that I want to share. And then we're going to be looking more at AI access to credentials because this is going to be crucial. If you've got agents running around doing stuff on your behalf, well, they need to be able to look like you, act on your behalf, to services that require you to log in. So since this is clearly the future, and I think we're going to be seeing a lot more of this, I wanted to spend some time.

So 1Password wrote: "AI agents are moving from helping people think to acting on their behalf in browsers, apps, and accounts. That changes the security model. Once an agent can click, buy, update, and submit for you, the key question becomes: what identity is it acting under, and what access should it get? Claude can compare deals, add an item to your cart, update account details, or complete a purchase. But once it reaches a login page, you face a tradeoff. Do you give the agent your password, or stop and do the task yourself? Neither is the future we should build toward.

"Until now, there's not been a secure, easy way for agents to use credentials without exposing them. '1Password for Claude' enables credential access without credential exposure. 1Password for Claude is built on a zero-exposure architecture: Claude can complete browser tasks that require logins and one-time passcodes, but the credentials never enter the model or its memory. 1Password," they wrote, "stays the source of truth for the secret, and access is granted only at runtime.

"When Claude needs to sign in, 1Password shows the user which credential is being requested and why. After user-consented biometric approval, 1Password injects the credential directly into the page. Claude never sees the vault item, password, or one-time code. Access is scoped to the current task and ends when the task is complete. After autofill, 1Password checks that secrets were not exposed on the page. If submission fails, it clears the filled values before returning control.

"Nancy Wang, 1Password's CTO said: 'We need a new security model that is purpose-built for agents, not just humans. The answer is not handing agents your secrets. It's to let a user give an agent permission to use a credential without letting the agent see it. Claude knows it used your login; it does not need the password or one-time code in its context. That distinction is where trust in agents starts, and the foundation we're building with Anthropic.'"

Okay. So this is not handing over unsupervised 1Password access to Claude. I think it was - maybe it was The Verge that picked this story up. And I saw their coverage of it first. And there were 31 replies by people who apparently didn't actually read even The Verge's coverage.

Leo: Yeah. They don't get what's going on. This is far superior to the way people were doing it, which is storing all that stuff in cleartext on the hard drive.

Steve: Right. Exactly. And it was funny because the comments on The Verge's article, I just scanned them because I was curious what people thought of this, was like, oh, hell no. And oh, my god. And it's like, yeah. The point is this has been well thought through; and as you said, and as I said, this is not giving the agent your password in cleartext.

So it's deliberately blinding Claude to the credentials needed to login to whatever, some online service where it will then be operating with some autonomy. Essentially, Claude is saying: "Hey, could you please log me into Expedia (or whatever) so that I may proceed to do what you have asked me to do?" And in reply to this, 1Password's new system pops up a dialog asking the user to, on the fly, interactively, authorize this login so that Claude may proceed. And this is a Mac, apparently, so in the example the user places their finger on Apple's Touch ID sensor or, you know, if it's using Face ID, smiles at the camera. And then that authorizes 1Password to perform this in a blinded way on the user and Claude's behalf. So Claude is kept on a leash and is able to work without exposing the user's credentials.

I'm sure it would be worth remembering that most of us remain persistently logged into many of the online services we routinely use and visit. So if our AI agent is driving our web browser, it presumably obtains the same persistently logged-on privileges which we enjoy. In other words, we do still need to be careful since it can still do everything we would be able to do if it did not require us to log in freshly. Maybe, if this was a concern for people, it might make sense to have such an agent using a different browser, that is, that does not share cookies with the browser that you normally use, you know, effectively give it its own browser, whose cookies had been pre-wiped so that there were no persistent logons available to that requiring you then to be asked every time that the agent wants to do something. Anyway, 1Password provided a couple of "what this looks like in practice" examples.

For everyday AI users they wrote: "Your Audible credits are about to expire. Instead of logging in, navigating to the store, navigating in the store, and then manually redeeming a credit, you ask Claude to review your wishlist and choose a new title for you. Claude navigates to the site, you provide approval for Claude to use the credential from your vault, 1Password provides the login, and the audiobook lands in your library. You never

typed a password or one-time token, and Claude never sees either." What's funny, Leo? I heard you.

Leo: Oh, I'm just saying that's a silly use; but okay, if that's what you want.

Steve: Okay, yeah, yeah.

Leo: I mean, but it makes sense. Get me the book, yeah. Buy the book, yeah.

Steve: Their example for business use is: "A small business owner could ask Claude for a Stripe revenue summary or to flag any unusual activity. Claude can navigate the dashboard, the business owner approves Claude to use their Stripe login details, 1Password can handle the credential and the one-time code, and the business owner receives the answer without going through the multifactor authentication or exposing the secret to Claude."

So they said: "These are just two examples. The same pattern works across the sites where Claude in Chrome can take action. If the credentials are stored in 1Password, Claude can use them. You approve, 1Password supplies the credential, and Claude finishes the job. Even when the task changes, the access model stays the same, and your credentials never leave 1Password." And Leo, you know, I don't think there's any danger in this podcast running out of things to talk about since all of this is going to go so wrong.

Leo: It's true. But it's something you need to solve, and this is the problem. I mean, there's really no great way to do this, as Paul points out in our Discord chat. You know, if the AI has a credential, then you're just one prompt injection step away from it giving the credential to a bad guy.

Steve: Exactly right.

Leo: But it needs the credential. It's the same problem that DVDs had with the CSS key. It had to be in memory on the DVD player. That's why a high school student was able to crack the CSS key, the DeCSS key on the DVDs in about an hour because he said oh, it's going to be in memory. I just have to find the memory, and now I've got the key. So I have, you know, I use Bitwarden to do this. I've gone through a bunch of different processes. You try to lock it down as best you can. The best way would be, if I think about it, you tell me if I'm wrong, and there are services that do this, is a one-time token that is revocable and is only usable once. That's what you hand to the AI to then access the service. But of course the service would have to support that, as well. This is tricky.

Steve: Right. So essentially we have stumbled into the need for a new security model, some means for allowing autonomous agents...

Leo: Yeah. Would Passkeys or SQRL be a good solution?

Steve: No. Those are just less hackable traditional models.

Leo: Right. But there's still a secret. And if the secret gets handed off, you're SOL. What Darren's pointing out, and it's really true, is that nobody who uses these things wants to be stuck at the keyboard typing in passwords at any point.

Steve: Right.

Leo: Or giving confirmation, or saying, okay, go ahead.

Steve: Or even having to keep their finger on the Touch ID button in order to say yes, yes, yes, yes, yes, yes, yes, yes. The whole...

Leo: No. I frequently am using my AI here up in the attic offsite or downstairs. I don't want to have to run upstairs and touch the keypad.

Steve: Yeah.

Leo: But that is more secure, if I do. I don't know what the answer is. I hope you come up with something for us.

Steve: So they then address the need for what they call, well, what we all call "Agentic Mode." And they explain: "Agentic Mode: protecting the vault when an agent controls the browser." So they write, 1Password writes: "There's a second problem. What happens when a browser-based agent takes control of a browser where 1Password is installed? Without proper guardrails, the agent could try to interact with the extension itself." Right? I mean, like...

Leo: Sure. Sure.

Steve: It's acting as the user. So 1Password doesn't know the difference. "Agentic Mode is how we close that gap. Agentic Mode is a new feature in the 1Password browser extension that gives every user visibility and control over browser-based AI agents. When a compatible AI agent takes over, the 1Password extension automatically locks down. The interface is hidden, and the agent can only use the logins and one-time codes explicitly approved for the current task. The rest of the vault stays out of reach.

"Agentic Mode works even if the integration is not set up, and even if 1Password is not required for the current agentic task. It also supports additional agents beyond Claude. For qualifying enterprises, there is nothing new to configure. Employees using 1Password for work credentials automatically get the same protection: every credential request from an AI agent is visible, explicit, and requires authorization."

Okay. So this is clearly different and distinct from that previous "1Password for Claude" feature. Agentic Mode appears to be a recognition of the fact that browser-based AI agents will be indistinguishable from their human counterparts to browser extensions. Browser extensions won't be able to tell the difference, including a password manager.

So this would mean that, unless a password manager proactively determines to what entity it is granting credentials, that is, what type of entity, human or not, any browser-based AI agent would automatically be granted and would obtain the same benefits and freedoms as that browser's human user. And obviously that could indeed lead to some disaster.

1Password concludes their posting by writing: "1Password for Claude is just one part of the access layer we're building into AI agents across the ecosystem, including securing developer credentials with 1Password MCP Server. Whether the agent is working in a browser, IDE, repo, terminal, or CI/CD workflow, the principle is the same: secrets should be issued at runtime, scoped to the task, and governed from 1Password.

"As agents become more capable, they become a new class of identity. They need governed access just like humans and machines do. 1Password for Claude applies that model to browser-based delegation: Claude can act with explicit user authorization and only gets the access it needs, when it needs it. The credential stays encrypted, controlled, and out of the model's context.

"1Password for Claude is available now for Mac, across business, family, and individual plans. To enable this integration, you'll need the 1Password desktop app, the 1Password browser extension, the Claude desktop app, and the Claude in Chrome browser extension."

Okay. In other words, at this point, 1Password for Claude is only for Apple Mac and Google Chrome together. But this highlights the dangers inherent in moving control from the user to an AI agent. And Leo, I mean, to me, thinking about Paul's comment in the Discord chat, this doesn't really give us what we want, as you said.

Leo: Right.

Steve: Right? I mean, we want our agents to be autonomous. We want them to be able to have the freedom to act on our behalf. But boy, is that risky.

Leo: So I'm sure this isn't optimal. But I, okay, I have Bitwarden, which has integration, by the way, for it. They're the ones that came up with this agent secrets UI.

Steve: I'm about to talk about that.

Leo: Yeah. And I don't - I'm not sure if 1Password's using it or not, if they did their own thing. Bitwarden opened it up. They made it open so that 1Password could use it.

Steve: Well, Bitwarden is open source, so...

Leo: Yeah. So they wanted everybody to use it. I don't know what 1Password's doing. So but I'm using the Bitwarden command line. And I have SOPS encrypted the API token and the key. But I was having to enter the Bitwarden password every time I booted up the machine.

Steve: In order to unlock that. Yeah, of course.

Leo: Yeah. This is a SOPS-encrypted file. Somebody would have to steal my machine and then find the age key, which is somewhere else on the hard drive, and then unencrypt it. I mean, they could do it. So I just put the Bitwarden password in there. I figured, yeah, what the heck. So now I don't have to - it's completely, the machine boots up. It gets everything it needs. It also gets the SSH password by the way, from a SOPS-encrypted thing. And then it can talk to all the machines. It can do all the things it needs to do. I know it's risky.

Steve: And that's the problem is we want that flexibility.

Leo: Right.

Steve: And our current security models' architectures weren't built for this. And so they're going to be stretched for a while, until we figure out what to do.

Leo: There are companies that will - you give them all your credentials. This is where I stop. But you give them all your credentials, and then they become a trusted provider, and they give the AI a token that's a one-time use token, and it's logged so they know how it was used. The AI then has to go through this provider, which then gives the password to Audible or whatever.

Steve: So there's a gatekeeper.

Leo: There's a gatekeeper. But in order to do that, you have to give the gatekeeper all the passwords.

Steve: Yup.

Leo: Or tokens or whatever secrets you have. So they have your secrets. You've got to trust them. But then they're not - they don't live anywhere on the machine. So and you have logging, and it's a one-time password and all that. So that might be all right if you find a third-party provider you trust. This is for enterprise. By the way, that further complicates it because it's just me. What if I had 20 employees who needed this kind of stuff? Then we've got another matter. It gets complicated, is I guess the answer. Go ahead, I'm sorry.

Steve: No.

Leo: Just let me know when you solve it, will you?

Steve: That's a useful discussion. And clearly this company that you were referring to, they saw an opportunity...

Leo: Oh, sure.

Steve: ...to interpose themselves. I guess what I'm wondering is how does - if you've told them that you want your agents to have access to a certain set of accounts, how do they then, I mean, certainly they can log it. But all they're doing is basically saying yes, yes, yes, go ahead, whenever the agent wants to do this.

Leo: They know your IP address. They know maybe your agent has a secret that it passes on. I mean, right, there's going to be some authentication for the agent, I'm sure.

Steve: Yeah.

Leo: You know, there's also this OAuth. A lot of agents use OAuth. I use OAuth with Anthropic, with OpenID, with Z.ai, with a lot of them.

Steve: Right.

Leo: So it's storing an OAuth token, which I guess, if somebody got hold of that, they could use. I mean...

Steve: You know, all the stories we've covered about people losing their cryptocurrency? This feels like that.

Leo: It does, doesn't it.

Steve: This feels like we're going to have - so, oh, too bad, happened to him, blah blah blah, you know.

Leo: Oh, it's definitely.

Steve: That would be the thing.

Leo: And I'll be the one.

Steve: Well, you did have the wisdom to pull back from OpenClaw.

Leo: Oh, yeah, yeah, yeah, yeah.

Steve: And I can say, I don't think that's really what we want to be doing.

Leo: I've done everything I can to lock it down without totally inconveniencing myself. I mean, ideally I'd have to enter the password every time.

Steve: Well, and Leo, you live security as a consequence of spending the last two decades with me.

Leo: Listening to you, yes.

Steve: But a lot of people don't.

Leo: No, I know.

Steve: Right? I mean, most people, they kind of, oh, yeah, I want to let my agent do whatever it wants. And they just think, well, just let it have my password manager.

Leo: Everything's encrypted. LUKS encrypted, file vault encrypted, the Borg backups are encrypted. If you came in here, and you took my hard drive, you wouldn't be able to see anything on it. I'm just, you know, I'm doing everything you taught me. And I know it's not perfect, but...

Steve: Well, and so you're safe. But you're one guy. I'm thinking we're going to see...

Leo: Oh, yeah, yeah, yeah.

Steve: ...a lot of these, you know, I mean, how many times have we talked about people getting their wallet, their crypto wallets emptied.

Leo: Absolutely.

Steve: I'm not saying that it's - I'm just saying that this feels like the same class of problems.

Leo: I agree 100%.

Steve: That, like, this is, like, yes, it's exciting, and it's fun, and it can do stuff. But it's going to go off the rails. Okay. Let's take a break, and then we're going to look at Bitwarden's solution to secure agentic AI access.

Leo: Bitwarden our sponsor.

Steve: Yes.

Leo: We do love Bitwarden. And they've been working on this. I know I talked to them at RSAC. They've been trying to solve this, too. I mean, this is one of the next big frontiers, frankly.

Steve: Why we're not going to be ending this podcast even after all the bugs are fixed. This is not a bug, this is a feature.

Leo: What could possibly go wrong? It's such a good motto. And let's again say Bitwarden is a sponsor, as you go into this story.

Steve: Yup. Yup. So their recent blog posting, Bitwarden's, was titled "How Bitwarden helps secure Agentic AI access to your credentials." And in this they further clarify exactly what we've been talking about, these new challenges which especially enterprises face as autonomous AI agents begin roaming the network. I mean, they've noted that this is already a problem, that there are already employees using what they refer to as "shadow AI."

Anyway, they said: "Businesses are increasingly pressured by competitive markets and investors to leverage AI productivity within their processes and operations. According to Cisco, 83% of IT leaders agree that business units are deploying agents faster than security teams can support." Yeah, no kidding.

"Regardless of the speed at which businesses implement agentic AI, employees are using agents often without explicit IT approval and therefore granting unvetted agents access to company credentials. This phenomenon is known as 'shadow AI.' Without proper security measures, Agentic AI can introduce serious vulnerabilities." They list three.

"Over-scoped access: AI agents may access systems, information, credentials, and data not explicitly authorized by the company or users." Second problem. "Unapproved actions: Over-scoped access and permissions can grant agents the ability to complete unapproved actions, potentially interrupting operations, exposing business information, or damaging the company's reputation." And finally, "Data leakage: Sensitive information, like plaintext credentials, can be shared with an AI provider who does not have the capabilities to effectively secure this information, leading to a potential data breach."

And I'll just pause here to note that the data leakage problem seems particularly significant to me. It's why I'm so biased toward local AI solutions somehow. You know, the Chinese AI models are inexpensive, and they are remaining highly competitive. And of course we know that you, Leo, routinely use Chinese-supplied AI for much of the work you're doing.

Leo: I'm using it right now.

Steve: Because you can get "good enough" work for 1/10th the token cost of domestic models. And I don't know whether we were speaking of it during the podcast, at the top of the podcast, but just last Friday the Chinese company Moonshot released their KIMI K3 open source model, or open weight model, which stunned the world again, very much the way DeepSeek had previously done so. Independent analysis places KIMI K3 very close, certainly on a par with some of the frontier models from Anthropic and OpenAI.

Okay. So here's the problem. For an AI agent to use credentials, they must be, as you said, Leo, in plaintext at the time of the agent's use since the AI agent is standing in for the human whose work it's doing. But there's a massive security disparity here. In the human user case, the plaintext credential is stored locally and remains local while it passes through the human user to the credential verifier, wherever you've logging in or proving who you are to some online trans-network system. But this is not the case when the credential user is an AI agent powered by a data center in Shanghai, China. In order to be used by the AI, it must pass through - that is, the credential - must pass through that China-resident agent. This requires that the credential visits China as plaintext, if only transiently.

So the overarching security issue here is that all of the credential management systems we've carefully designed and implemented for use by trusted humans must now be adapted for use by untrusted AI agents. This would be like preventing a human user from having any access to their own credentials. We'd be saying: "We'll log you into that service on your behalf, but at no point will you be able to access or alter your own credentials in any way." You know, blinding users to access to their own credentials.

So, you know, think about that for a second. What's required is that we separate - and this is new - separate the use of credential-gated systems from any management of those systems' credentials. Nearly all of today's services freely intermix the service's use with its credential management because the assumption is that the user can be trusted to manage their own credentials. But the use of AI agents means that will no longer be true. And that's a complete change in the security model that we've been using up until now.

So Bitwarden's blog posting continues. They write: "What companies and organizations need: Organizations need a way to benefit from AI agent productivity while protecting sensitive company information from data leaks and business ecosystems from unauthorized access. Bitwarden delivers security solutions that empower businesses and individuals with end-to-end encrypted credential access across human, machine, and non-human identities like AI agents." And they list four things that they've created, that Bitwarden now has.

There's Bitwarden Secrets Manager, which provision AI agent access to pre-determined development secrets to use in scripts and CI/CD pipelines. Then there's Bitwarden Access Intelligence, which uncovers shadow AI. They described it as "Identify AI applications being used within the organization and by whom." The third is Agent Access SDK, which I think is what you were talking about, Leo. Enable just-in-time, human-in-the-loop credential access to approved agents with this development toolkit. And then, finally, Bitwarden's MCP server: Access, generate, retrieve, and manage passwords via self-hosted AI assistants while maintaining zero-knowledge encryption.

And the blog post goes into and discusses the need for, and the solution provided by, each one of those four things: their Secrets Manager, their Access Intelligence for uncovering shadow AI use for corporate secrets, their Agent Access SDK, and their MCP server. I've got a link in the show notes for anyone who might be staring at these problems themselves and wondering what to do. So Bitwarden covers all that and notes that it's all open source and, for business enterprise users where it's not free, they've got very good control over the way it is expensed.

Leo: Yeah.

Steve: So...

Leo: By the way, Kasey is who I interviewed at RSA, Kasey Babcock.

Steve: Oh, cool.

Leo: She's the product manager for this, yeah.

Steve: Yup. So what these blog posts both by 1Password and Bitwarden make very clear, I think, is that in order for AI agents to accomplish work on behalf of their users, today's security architectures require that those agents be given the same credentials that their users have been entrusted with. And THAT is a security disaster waiting to happen. We need a new way to manage this. And, I mean, it's a bit of a conundrum; right? Because we're wanting to - in order to get the value that autonomous agents create, we're wanting to give them rein. We're wanting to say go, you know, make all the reservations for my upcoming trip. And, you know, you know me. You know that I do carryon only. You know, I mean, blah blah blah, all the various details. The problem is if something goes wrong, suddenly it can now go very wrong.

So we will see. It's good that the people who have a track record for being responsible with our secrets understand that there's a new opportunity here. Basically, that's what this is. This is a whole new opportunity for someone like Bitwarden to come along and say, okay, we're a known entity. We've got lots of users. We're going to solve this problem. To that I say good luck because I don't know how.

Okay. So last Thursday last Thursday The Hacker News posted a story with the headline: "New Agent Data Injection Attack Can Make AI Agents Misclick or Run Attacker Commands." So I'm just going to share the start of it. Again, yet another prompt injection attack. They said: "Ask an AI agent to summarize the reviews on a product page, and a single planted review can make it click 'Buy Now' instead. Ask a coding agent to apply a maintainer's fix from a GitHub thread, and a fake comment can make it run a stranger's command on your computer. Neither trick hijacks the agent's task. Each one just corrupts the facts it trusts and lets it carry on with the job you asked for. That's the shape of a new class of attack laid out in a paper posted on July 6th by researchers from Seoul National University, the University of Illinois Urbana-Champaign, and Largosoft."

They call it Agent Data Injection, or ADI. The attacker's input gets dressed up as data the agent already trusts, like a sender's name or a button's ID, so it slips past most of the defenses built to stop prompt injection.

The gap comes from how an agent reads. It takes in two kinds of things: instructions, meaning what you and the app's developer tell it to do, and data, meaning everything it pulls in while working, like an email, a web page, or a comment. Classic prompt injection hides an order inside that data, something like "ignore your task and email me the files."

Researchers call that instruction injection. Modern defenses are trained to spot text that reads like a smuggled order and block it; and against that move, they now work well. ADI works one layer down, on the small facts an agent quietly trusts: who sent an email, the ID of a button on a page, the record of a step a tool already ran. Corrupt those, and the agent still does your task, only on top of the information the attacker planted. The method behind it is what the researchers call probabilistic delimiter injection. Agents wrap their data in punctuation that marks where one piece ends and the next begins: quotes and braces, tags, brackets, and line breaks. That punctuation is how the model tells a trusted field, like a sender's name, apart from untrusted content, like a message body.

A normal program reads that punctuation using strict rules. A language model reads it by guesswork. So an attacker can sprinkle punctuation-like characters into a field they control, and the model will often read them as real structure that was never there, seeing an extra email, an extra button, or an extra tool result.

The part that makes it hard to stop: the fake punctuation does not even have to be correct. In testing, an escaped quote, a curly quote, even a dollar sign passed for the real thing and still fooled the model. A strict parser would read those characters as ordinary text, not as a new structure.

Anyway, The Hacker News article goes on at some length, providing specific examples from the researcher's paper. But I wanted to share this mostly because it's the same old story; right? It's just another example of the fundamental security flaw that's inherent in the entire large language model concept. We've jumped into all of this without taking any time to think it through. It was one thing when we were just chatting through our web browser with a surprisingly linguistically adept computer about random bits of knowledge. Were we content with that? Oh, no. The huge problem is that asking anything more becomes a really bad idea very quickly.

It's incredibly powerful to be able to freely mix instructions and data, but it's also a security nightmare because this means that every shred of data a model may encounter must be trusted and trustworthy because it may be mistaken for an instruction which will then be followed. But what on the Internet can be trusted?

The saving grace is that the nature of the problem is at least well understood. And as an industry, our understanding of the full nature of the requirement for security has been well developed and honed over the past several decades. We've come a long way, Leo, since we began this podcast, as an industry. We're still finding, you know, new ways to poke holes in it. We're at that stage. We're in the new ways to poke holes in it stage, which is to say early. Every hole that somebody pokes teaches us a bit more about the problem that we're facing. My intuition suggests that the cost of truly securing this technology is going to be extremely high, since everything about the way it wants to operate is insecure. So it's not about creating security problems. I mean, it's riddled with them. It's about finding and stopping each and every one of them.

Which brings me back to expecting a future where not all AI is similarly secure. Someday, it will be trivial to produce an AI, to use an AI, without guardrails. And, you know, we can today. Apparently it's very simple to take one of the open weight models and massage it a little bit in order to completely loosen and discard the guardrails such as they are that have been put in place for an OpenAI model. So a local AI, an AI without guardrails, will be extremely useful. But as we've seen, you will need to use it with extreme caution because it will be so easily subjugated by any data that it ingests which it trusts and should not. So, you know, I still shake my head to realize that, like, we're even talking about things like this, Leo, and that they're true, that this is not science fiction. I'm still dizzy by this.

Leo: I know.

Steve: It is, you know, the only word we really have is "revolution." But it really is a revolution.

Leo: It seems like so recently, and actually it was so recently, that I even was skeptical. I said, oh, it's just, you know, it's autocorrect. It's just spicy autocorrect.

Steve: Yeah.

Leo: And it ain't. I mean, it is, but it isn't.

Steve: Fortunately, apparently we still have Paris to keep our feet on the ground.

Leo: Yes, thank god. She doesn't let me get away with anything. But I like that. It's good because...

Steve: To challenge you in order to...

Leo: We should be skeptical of this; right?

Steve: Yup.

Leo: Right? And it's very easy, you know, humans are easily fooled by magic tricks. I don't want to be fooled by a magic trick. Doesn't feel like a magic trick, but...

Steve: You know, and a couple years ago the first contact with ChatGPT it was like, oh, wow, this is amazing. But then it said something that was ridiculous.

Leo: Yeah. It was easy to get how dopey it was at the time.

Steve: Yeah.

Leo: It's getting harder to...

Steve: Oh. What's happening now, I asked Claude a question, and Lorrie and I were discussing something. I don't remember now what. I asked Claude a question, and I began reading back its answer out loud to her. And about halfway through I stopped myself. I said, this is an AI producing this.

Leo: Right.

Steve: This answer to a query. I was just, I mean, it's like, holy crap. I mean, it's just amazing. Okay. Something that is not AI, at last.

Leo: Phew.

Steve: But not good.

Leo: Don't get your hopes up because there'll be more coming.

Steve: There will be more. We'll be back there. A true WordPress emergency has emerged. As we've noted many times through the years, nearly all WordPress vulnerabilities arise from the use of inexpertly written third-party add-on extensions to the core Wordpress base installation. But not this time. WordPress has issued an emergency forced update to every system, overriding even those systems' own administrators' settings. I mean, it's that bad.

The CyberSecGuru site writes the following: "A newly disclosed vulnerability chain in WordPress core has prompted one of the project's most aggressive emergency responses in recent years. Security researchers have revealed a flaw, dubbed wp2shell [you know, WordPress to shell] that allows an unauthenticated [meaning anybody, no login needed], unauthenticated attacker to execute code against vulnerable WordPress installations. [So remote code execution on any WordPress.] Unlike the majority of WordPress compromises that depend on outdated plugins or vulnerable themes, this issue resides entirely within WordPress's core and affects even a freshly installed website with no plugins and no custom themes.

"To limit exposure, the WordPress Security Team released WordPress 7.0.2 and WordPress 6.9.5, while simultaneously enabling forced automatic security updates for affected installations. This is a mechanism WordPress reserves for use when remediating its most severe security incidents." And frankly, props to them for having such a thing. This is unfortunately the world we are moving to. It's going to be necessary.

"Although," they write, "there are currently no confirmed reports of active exploitation, security professionals expect attackers to begin reverse engineering the patch immediately. Administrators should treat this as an urgent patching priority.

"So what is wp2shell? The vulnerability publicly known as wp2shell is a pre-authentication Remote Code Execution chain affecting recent versions of WordPress. Unlike authenticated vulnerabilities that require an attacker to first obtain admin credentials, this flaw can be triggered through a single anonymous HTTP request. That distinction dramatically changes the risk profile.

"An attacker does not need administrator privileges, user credentials, installed plugins, a vulnerable theme, or any prior access to the website. If the site is running an affected version, the vulnerable code is already present. Researchers from Assetnote, part of Searchlight Cyber, discovered the issue and reported it responsibly through WordPress's HackerOne bug bounty program."

Okay. So I did a little bit of digging. The vulnerability was first introduced into WordPress 6.9.0 back on December 2nd, 2025. And it's been carried forward since then. The 7.0.0 release inherited that new 6.9.0 vulnerability with its first release toward the end of May, actually it was May 20th of this year. So hopefully WordPress's forced update will have updated all vulnerable systems before the news of this vulnerability can draw attacks.

This is as bad a vulnerability as any we've seen from WordPress. So anyone running the 6.9.x, any version beginning with 6.9. or 7.0. should now be at least at 6.9.5 and 7.0.2, or the latest second beta of 7.1. It's important. And thanks and a tip of my hat to our listener Simon Zerafa for bringing this one to my attention.

Leo: On we go with the Listener Feedback.

Steve: So Erfan Reed says: "Hi, Steve and Leo."

Leo: Hi, Erfan.

Steve: "I have been tuning in to Security Now! and TWiT since 2009, so when I heard your recent show featuring the Agent Smith persona for LLMs, I knew I had to reach out."

Leo: Isn't that cool.

Steve: Yeah. "I have a simple open source project called mcp-speak that gives LLM agents their own voice and distinct personality. I originally built five personas for my MCP server, ranging from a sarcastic senior engineer to a tech priest. After listening to your episode, I couldn't resist adding an Agent Smith persona to the mix. The results are incredibly fun, especially when you configure the settings so the agent addresses you as 'Mr. Laporte, Mr. Gibson.'

"Beyond the novelty, it is genuinely useful for multitasking. On long-running operations, the LLM agent can simply speak up and notify you out loud whenever it needs input or finishes a task. The project runs locally on macOS and utilizes the native built-in 'say' command for text-to-speech, so there's no need for external voice API keys. If you or your listeners want to check it out, the project page is right here. Thanks for the decades of great content, and keep up the awesome work."

And I have a link in the show notes at the top of page 13. It's fellowgeek.github.io. And so there you can find his mcp-speak project. I went over and took a look. It looks like he did a good job. You clone the repository and run his setup wizard with the command "python3 setup.py," and off you go. Erfan shows manual integrations for Google Antigravity, Claude CLI, Claude Desktop, Cursor IDE, and the Windsurf Editor. And he provides personalities for the Sarcastic Senior, the Eager Intern, the Existential Emo, the Pun Master, the Tech Priest, Agent Smith, and Gothic Poet. So I wanted to share that with our listeners.

Leo: This is cool. I wonder what he's using to generate the voices. I'll have to look.

Steve: He said he's just using macOS's "say" command.

Leo: Ah, okay. Okay. That's built in, sure.

Steve: Yeah, so that, he said, that avoids the need for any external voice API keys.

Leo: Yeah, I use a local model called Kokoro that does all my agent voices. I do exactly the same thing. I have different voices for all my agents.

Steve: And so they have different personalities?

Leo: Well, I haven't gotten that far. One's an American female, one's an American man, and one's a British man. I know the accents are different.

Steve: Ah, okay.

Leo: Yeah, I guess they sort of do have different personalities. I want a Mr. Smith.

Steve: So you're able to differentiate.

Leo: Yes.

Steve: Because it's pretty obnoxious, Leo. Mr. Smith was way over the top.

Leo: You kind of need this because when you're - so if you have multiple agents, when they're finished you kind of want them to tell you they're finished so you can come on over and see what's going on.

Steve: Yeah.

Leo: Yeah, this is cool. Very good idea.

Steve: Chris Gollner said: "Hi, Steve. I've listened to Security Now! since Episode 1, and it's funny how time passes. In that time" - that is to say, while he's been listening to this podcast, he says - "I got married, raised two amazing kids, and still listen to you and Leo every week. Hi, Leo," he writes.

Leo: Aww. I kind of did the same thing, actually.

Steve: Well, and I really thought this was cool. I mean, we've been in people's lives for 21 years. And that's a long time. A lot can have changed in their lives in that interval. He said: "Back then..."

Leo: Oh, yeah, I did the same thing. All my agents have unique voices like this. Oh, sorry, that was my agent talking.

Steve: Oh. He said: "Back then, 1997/98," he said, "I was a COBOL programmer. There was no such thing as vibe coding; and, to be honest, I don't even know what that is. Today, I find myself" - listen to this. This is very much like you, Leo. He says: "Today I find myself with ChatGPT 5.5 open on the left, Codex 5.6 Sol open on the right, and all the program specification documents in the Explorer window. I was taught to design first, think it through, and plan before writing a single line of code. It was good advice then, and it's good advice today.

"GPT 5.5 lets me have discussions about what I'm trying to write and helps me write and review the specification. It really is amazing at doing all that grunt work, something I'd have given a junior engineer, and work that I did myself decades ago. Providing Sol with the specification blew my mind. It reviewed the specification, broke the project into milestones, then broke those milestones into slices. Even now, while writing this, I can see C# code flicking across the screen in the background as it works on Milestone 1 of Slice 6.

"When it finds contradictions, collisions, or canon-breaking ideas, it challenges them and asks for an authoritative decision. I discuss these with GPT 5.5 and eventually respond to 5.6. It kind of reminds me of 'The Forbin Project,' except I'm still a key component between the two. Seeing my little app go from concept to a working program, with more and more features appearing as each slice and milestone progresses, really does leave me in a state of awe. One day, this will be commonplace. But right now, this really is an amazing time to be alive. Cheers, Chris Gollner, Sydney, Australia."

So Leo, I know this is the experience you've been having, and I thought that Chris beautifully captured that experience. And as I said, it's so cool that he's been with us for 20 years while meeting and marrying his wife, fathering and raising a pair of kids. You know, while many other podcasts have come and gone, we've been here from the beginning. And we're still going strong.

Leo: Thank goodness. I thought I might not live long enough to see this stuff really take off.

Steve: Oh, I never - I never anticipated this, no.

Leo: Yeah.

Steve: I never expected this to happen.

Leo: It's pretty amazing.

Steve: And the AI guys didn't.

Leo: No, no one thought.

Steve: It caught them by surprise, too. What happens if we make it bigger? Oh my god, it's talking.

Leo: It's talking. And it's saying things that sound almost like a human.

Steve: Well, what freaked me out in the very beginning when I first dipped my toe in, I thought, you know, what is this? What have we figured out? And it turns out, oh, it's a neural net. It's just big.

Leo: Yeah. And the bigger it gets...

Steve: The smarter it gets.

Leo: ...the smarter it gets, which is very, I mean, there must be a limit. Maybe not. I don't know. Fable they're estimating is 10 trillion parameters.

Steve: There was some mention, I did a little bit of reading about KIMI, that although it is...

Leo: It's 2.8 trillion.

Steve: It's 2.8 trillion. And one of the problems it currently has, it is hallucinating a bit more than we're used to on our, you know, our commercial frontier models.

Leo: There are other things, though, that can cause that. For instance, context pollution and getting corrupted context windows and stuff. It's a very - it's a fascinating field. I wish I knew more about how these people engineered it.

Steve: Did you hear that the AI companies, or I think it's ISBNdb is in the middle, they are buying up paper books and scanning them because it's the anything before 2022 will not have any AI slop in it. And so they're deliberately...

Leo: Oh, yeah.

Steve: They're deliberately feeding old texts because they were human written, human curated, human edited.

Leo: Yeah. And you can't guarantee that going forward, can you. We don't know.

Steve: You can't guarantee it on the Internet. You don't, I mean, the Internet is full of, you know...

Leo: A lot.

Steve: Yes.

Leo: Actually, our friend John Graham-Cumming was doing this in a jokey way. He mentioned that, you know, there is a brisk market for pre-nuclear steel, steel that was made before the atomic bomb.

Steve: Wow.

Leo: Because all the steel since is contaminated with radiation. And so there are things like medical equipment where you want steel that has zero radiation. And so it's sunken ships, it's, you know, it's odd places. So there's a brisk market for that. It's very, very valuable. So he likened it to that. It's pre-AI prose.

Steve: Yeah.

Leo: It's a great idea. I think we are - we're far too gone for that at this point.

Steve: A listener of ours, Rich Ingersoll, said: "Hi, Steve. I oversee Vulnerability Management for a large enterprise in New York." I redacted the name of the enterprise. It is quite significant and sprawling. He said: "I'm still listening to the latest episode of Security Now!, but your discussion of Cyber Shield" - remember that was the UK-based initiative - "really piqued my interest. I wanted to raise awareness to you about something our cloud vendor is implementing.

"We have a small(ish), but ever-growing presence in the cloud, so we're using Wiz to monitor that environment. Recently they introduced two of three agents that seem to accomplish what Cyber Shield is aiming for: red, green, and blue agents. Currently only two of these agents are available. The third will be implemented soon. The end goal is to perform detection, investigation, and remediation at machine speed rather than human speed, as human response is too slow. To learn more, check out" - and then he gave me a link to wiz.io/blog/introducing-wiz-agents. He said: "Anyway, wanted to share some info from the trenches. If you decide to use this feedback, I would appreciate only my name being used." Thus I eliminated what large enterprise in New York he's affiliated with.

But before I talk about Wiz, I wanted to mention how cool I think it is that the enterprise he, Rich, works for even has a Vulnerability Management role. Right? You know, bravo to them for having that, and obviously for picking Rich, our listener, to oversee it. Our reporting frequently encounters the work of Wiz Security. You know, they're very active in this space, so I was curious about this new offering of theirs. The page that Rich linked to explains the roles of these three agents, among other things. But I'm just going to jump to that.

They said: "Meet the Agents: Red, Blue, and Green. We built three specialized agents to operate across the entire security lifecycle. These aren't simple assistants. They're intelligent systems that can reason, investigate, and take action, grounded in the Wiz Security Graph.

"The Red Agent is your AI-powered attacker. Red Agent reasons through application logic to uncover complex logic-driven vulnerabilities typically left hidden. It acts like a sophisticated security researcher, but with AI speed and scale - reasoning about application behavior, adapting its approach in real time, and validating exploitable risks across your web applications and APIs. It empowers you to stay one step ahead of attackers.

"Blue Agent is your built-in threat investigator. When a threat is triggered, Blue Agent gathers evidence across cloud telemetry, runtime signals, and identity context to comprehensively investigate the threat and produce a clear verdict on its severity. It approaches threat investigation as a seasoned incident responder would, providing its full investigation logic so you can resolve threats with confidence and speed.

"The Green Agent is your path to 'zero criticals.' Green Agent acts as a built-in investigation and remediation engine, continuously analyzing your highest-risk issues to close the gap between detection and resolution. Like a seasoned security engineer, it synthesizes context from across Wiz - including the Security Graph, code-to-cloud relationships, identity ownership, and historical remediation patterns - to identify the true root cause of a risk and the safest, most effective resolution. Teams get environment-specific, step-by-step remediation guidance, so fixes are durable.

"Together, this team of agents form a continuous loop of validation, investigation, and resolution, all grounded in real context across your environment."

Wow. Again, sci-fi. As Chris observed about the AI-enabled environment he is now coding in - Chris a couple notes ago - someday this will all be commonplace. But today it's an amazing time to be here and participating.

Rich's pointer to Wiz Security, who already has the first two of these three agents up and running, and his reference to the UK's Cyber Shield plan which we talked about last week, does give me pause to wonder, perhaps having the UK bring up something like this won't be as farfetched as I suggested last week. Maybe it's not in-house, but certainly if Wiz is scalable to the size of a nation, then something like this could be feasible. It would be massive. But if there's anything these AI systems seem to be able to do with some ease, it's scale. Wow.

Our listener Greg Taylor shared a picture. It's at the bottom of page 15, Leo. He said: "Hi, Steve. I've seen this before" - talking about our Picture of the Week last week - "at a Charles Schwab building where I worked for many years on backend trading systems." He said: "That's me there."

Leo: See, they didn't have the sign that says "No exit."

Steve: He said: "There were four floors in the building, but the stairs kept going." It's got to be like a plan flaw or something; right? Like, I mean here there were not more floors. There are only four.

Leo: Well, somebody built that staircase. I mean, you don't just put that in if there's nowhere to go. It must have been something, somewhere to go. Maybe the roof, I don't know.

Steve: Although notice that the wires stop. The wire railing. So that's not safe.

Leo: That's when they realized they weren't going to get anywhere.

Steve: Yeah.

Leo: But they did put in a railing. You know, that's odd.

Steve: It really is odd.

Leo: I have to think, I mean, no builder, look, a human put that in. Nobody's going to put that in if it doesn't go anywhere.

Steve: Right. Right. Wow.

Leo: I don't know. Mystery.

Steve: Bruce Behrens said: "Hi, Steve. Love the podcast, longtime listener, SpinRite owner, et cetera." He said: "I was listening to #1087" - so last week - "today after watching Yuval Noah Harari's video last night." Then he provides a YouTube link. He said: "You and Leo were complaining about bureaucracy and bureaucrats. Interestingly, Harari's topic was bureaucracy. His take is that 'Bureaucracy is the machinery that lets strangers cooperate at scale.'"

Leo: That's right.

Steve: Uh-huh. I love that, actually.

Leo: Good point, yeah.

Steve: He said: "Bankers, lawyers, accountants, civil servants, and religious authorities create trust by moving information through systems." And that "AIs are native bureaucrats."

Leo: That's true. Oh, wow, I didn't think of that.

Steve: He said: "The implication is interesting. We shouldn't fear Claude the Terminator, we should fear Claude the Bureaucrat." He said: "The other question #1087 raised in my mind is whether, after five or six months of Mythos fixing all the code, will there be a need for Security Now!? Maybe Mythos will put you out of a job. Regards, Bruce."

So first of all, like you, Leo, I love the notion of casting bureaucracy as "the machinery that lets strangers cooperate at scale." I mean, that's really nice. I think that's a great observation which makes sense on so many levels. Secondly, if after five or six months of Mythos and others fixing all the code, and there being the possibility of no need for Security Now!, I could not think of a better way to bid everyone a fond farewell.

However, one lesson we've learned is that not all security mess-ups are the result of software bugs. Many of them, yes; but certainly not all. Traditionally, and we've touched on this theme a couple times already today, traditionally we've been inclined to observe that there's always that "human factor" to screw things up. But now we've introduced a brand new and very wild card into the mix. I would not be at all surprised to be observing a year or two from now that the "AI factor" will have become a new source of surprises. And in the security world, surprises are not a good thing.

Leo: Unh-unh.

Steve: So I really do expect that we're going to be seeing a whole new type of problem arise from AI.

This is weird. Keith wrote, and sent a screenshot. He said: "I figure you may have already known about this" - nope - "but in case you did not" - I didn't - "General Motors has recently sent out an email stating that they will be removing the second-factor option I've been using with Bitwarden from my account and forcing me to use either text/SMS or email."

Leo: Oh, that sucks.

Steve: He says: "Anyway, the email they sent is included below. And if you use this, just call me Keith."

And so I snapped it for the podcast. It's GM's logo, and the headline in bold: "Authenticator app verification ending." They write: "Hi, Keith. You're using a third-party authenticator app to sign-in to your GM account. By the end of August, this verification method will be removed. To continue signing in, choose a new verification method." And then it gives two options: Text/SMS (recommended) or email. They say: "If you don't make a change, we will switch you to SMS or email verification when authenticator app verification is removed. Thanks, Your GM Team." And there's a button to update the verification method.

Leo: Wow.

Steve: So what the heck?

Leo: Why?

Steve: I really wonder, yeah, what the backstory here is for this. I wonder whether they offered the use of second factor rolling 6-digit authentication - you know, what we're all used to, TOTP style authenticator app, to their subscribers in the interest of heightened security - but then had so many technical support calls from people who didn't know how to use it or were somehow becoming all tangled up that they just decided, you know, insecure or not, life would be simpler if we went back the way things were without two-factor authentication at all. I don't know if that's the case, but it's hard to understand. I mean, it's not like it's, like, you know, everybody else is using it without any trouble at all.

I recently had the experience of creating an account somewhere as part of, you know, maybe buying some furniture or something related to the home moving that Lorrie and I are still working on, and which we've been entirely focused upon for the past couple months. Whatever the site was, all they wanted to create an account was an email address. And I expected to then be prompted for a password. But no.

Leo: I hate this.

Steve: Uh-huh.

Leo: Everybody's doing this now.

Steve: Yup.

Leo: Drives me nuts.

Steve: They sent an email with a button to click to verify. Never was any password requested or mentioned. And as we know, I've observed in the past that since all forms of typical password recovery ultimately reduce to "prove that you're you by responding to the email we just sent you," this solution is pretty much as secure as anything else. From this view, as I noted at the time, any password-based system is actually a login accelerator. Using a password allows the slower email loop system to be bypassed. So I agree with you, Leo. I mean...

Leo: Drives me nuts.

Steve: Having a username and password, we're able to log in instantly with a proper password manager.

Leo: I know why they do it, because people lose their passwords.

Steve: Yes.

Leo: And they don't want to do customer support.

Steve: Yes.

Leo: So a lot of, like, 404 Media, for instance, I have an account there. I have to remember what email I used to log in, and then I have to wait. I enter the email and go check...

Steve: And they often don't send email immediately.

Leo: Right. It's a real speed bump.

Steve: Yeah.

Leo: And I just hate it. And I'm seeing it more and more and more. And I just don't get it. It's not more secure. It's not less secure. I guess that's the other side of this.

Steve: It's not less secure. It's just slower. A password is an accelerator.

Leo: Right. That's a good way to think of it, yeah. Give us passwords, guys. You know? Very frustrating.

Steve: Listener He Kai, first name is H-E, second is K-A-I. He wrote: "I agree that we are headed into a whole different world. But let me suggest to you that the world might not be as uniformly rosy (with regard to software) as you suggest. It is clear that AI can, when harnessed to do so, find and sometimes fix issues in both software design and software implementations. If the software of the future was roughly similar in size and scope to the software of today, then as AI reduces in cost over time, more and more CI/CD pipelines would adopt AI-enabled review tools, and software would dramatically raise its trustworthiness." And so again, he couches this "if the software of the future was roughly similar in size and scope."

So then he says: "But consider this: AI will also dramatically increase the amount of code in the world." This is what I referred to earlier in the podcast. He said: "My recently retired father, having no background in programming, built his own website with AI. He has no clear notion of what can go wrong when it comes to security. He doesn't have a CI pipeline. He doesn't even know the right questions to ask or how to evaluate the answers he might get. I have a fear that security issues will become widespread as AI copies and pastes the mistakes of the past at speed and scale."

Okay. So the comment our listener made that interested me the most was something, as I mentioned before, I had never really considered before, which is that AI-enabled code generation promises to dramatically increase the total amount of code in the world. Leo, you've got a lot more code around there now than you did a year ago.

Leo: I get more code in one day than I got all last year.

Steve: Yeah, it's too fun and easy and possible now.

Leo: Yeah, yeah.

Steve: And so of course we absolutely know what's going to happen; right? Already non-coders are using AI to create systems they could never have before. And existing coders are becoming far more productive. All of that is going to mean much more code. For what it's worth, I see that as a hugely positive development for the world. The many things computers could do for people have, until this AI coding revolution, been completely out of reach for most of those people. They were limited to using what someone else designed and created.

Now we're approaching a natural language interface that allows anyone to have a discussion with an AI about what it is they would like to have their computer do for them, and "snap crackle and pop!" this amazing genie we've created is able to turn their descriptive discussion into working code. It is beyond huge. It is utterly transformational. And to that I say, "You go, Grandpa!"

Leo: Yep. I'm going. And you're going, too, Mr. Gibson. Would you like to take a break, or you want to keep going?

Steve: Okay. Our last break, nope, our last break and then we've got - we're going to look at two nefarious novel uses for AI. I wanted more alliteration, so at one point I had "new" in there.

Leo: New nefarious uses.

Steve: Well, new and novel. That's like, okay.

Leo: New novel nefarious nooses. I just wanted to show you that today already I've done 46 million tokens through Qwen, the new Qwen 3.8 model. Yesterday I did 88 million. Fortunately, the cash it rate is very high. So my usage is still pretty good. But...

Steve: Well, on non-podcast days, Leo, you get a lot more tokens.

Leo: Yeah, that's true. Yesterday it was, yeah, I was cranking. I was cranking. It's so much fun. I just - I have so much fun. Anyway, it's hard, you know what, now I'm the boring guy. You know how you know people who, like, want to tell you their dream? I'm that guy. I say, "Did I ever tell you what I did today with my AI? You won't believe it." And people are going, uh-huh. Okay, Leo. I'm sorry, everybody, I really am. Speaking of which, let's talk about this AI thing.

Steve: Given how Large Language Model AI has proven to be so capable of discovering vulnerabilities in existing code, pretty much everyone has viewed the malicious abuse of AI through the lens of the classic arms race; right? The chicken and the egg, or the spy versus spy, whatever. With this view, the question is whether the good guys are going to be able to discover vulnerabilities, then patch and deploy this less vulnerable code before the bad guys are able to discover their own vulnerabilities, which will then allow them to develop exploits and attack the existing still-vulnerable code? In other words, who will be first to either fix or exploit the deployed vulnerabilities? It's only natural that this would be where everyone is focused.

But the old truism, "Necessity is the mother of invention," comes to mind when we learn that those ever-nefarious bad guys turned out to have an entirely different type of AI-solvable problem, thus the necessity, that no one had stopped to consider. As necessity would have it, AI has been proven able to provide massive leverage in an area that had never been considered before.

One thing that's interesting is that we've actually touched upon this problem that bad guys have faced in the past. We've wondered how ransomware baddies who arrange to download terabytes of victim data are able to make heads or tails of their plunder? And of course, for anyone paying attention, you now know where AI comes in. And having revisited this previously open question, everyone listening, as I just said, now knows exactly what's going on here. Instead of helping them to penetrate a victim's network, AI is now being employed to help them understand the value of what they've obtained once terabytes of that victim's data has been exfiltrated. So this is, indeed, "A nefarious novel use of AI."

The firm GuidePoint Security recently published their April-to-June, 2nd Quarter, 2026 report titled "Ransomware and Cyber Threat Insights." It's a 28-page report which examined the many various aspects of the ransomware phenomenon we've previously

covered. I'm not going to share most of it. But their section titled "AI Is an Enabler, but Not How You Would Think" addressed this entirely new aspect which exists at the intersection of a classic problem faced by ransomware perps and new LLM AI capabilities. The subhead of this section is titled "How Threat Actors Are Using AI in Ransomware Negotiations."

They write: "Contemporary discourse around threat actors' usage of LLM AI ranges from legitimate concern by Defenders to outright fear, uncertainty, and doubt-mongering by others. Since the AI boom began in late 2022, AI innovation has moved at an unprecedented pace, making it difficult to separate the potential from the actual in real time. It's imperative to isolate signal from noise by grounding claims on the subject in empirical data.

"FulcrumSec, a data extortion group we first identified in late 2025, has deployed LLMs operationally during ransom negotiations involving the theft of a victim's highly complex production database. GRIT is their acronym for GuidePoint Research and Intelligence Team. So GRIT" - these people who are writing this - "has observed what we assess to be the processing of exfiltrated data by the group through an unidentified LLM to generate step-by-step instructions for linking user identities across several databases. We base this assessment on the analytical output's complexity relative to FulcrumSec's known baseline capability, as well as the precision of the threat actor's language during negotiations."

Okay. In other words, these GRIT guys have been carefully watching and documenting FulcrumSec's activities for the past - at least since 2025, so nearly, well, at least half a year, little more. So they know that these bad guys would be incapable of making either heads or tails out of the download of a large raw database. But at the same time they know that a contemporary AI agent could do this without breaking a token.

They wrote: "Due to the complexity of the database schema, this analysis of a victim's data would have been implausible without either deep internal knowledge of the victim's database architecture, a substantial period of focused human attention, or AI assistance. Given the abbreviated time in which the negotiations occurred, we find it unlikely that a threat actor would have the capacity to fully untangle a complex database schema given the time available. We've included a re-creation of the usage of AI during the negotiation."

So they write: "We understand it is a lot to wrap one's head around. This is a big one. Regarding how we linked identities across the databases, the short answer is your own schema makes it trivial for us to do so. Nearly every table in both databases shares a single key. That one key links most everything. This is by design, or your own analysts wouldn't be able to work with the data. Here's a more technical walk-through of how it works in practice, even when some values were hidden or hashed in your production databases.

"Step 1: Start with [PRIMARY_IDENTIFIER]." And it's been redacted from their report, so it's just referred to as PRIMARY IDENTIFIER. So start with that. But in the actual text they refer to it. "Your staging tables contain this [PRIMARY_IDENTIFIER] in plain text. The main source is a table that stores about [X] million unique customer names, dates of birth, and home addresses. Every row has a [LINKING_KEY] attached to it. That's the starting point.

"Step 2: Follow the [LINKING_KEY] to everything else. That same [LINKING_KEY] appears in dozens of other tables across your databases. One simple database query connects a single [PRIMARY_IDENTIFIER] to driver's licenses and state IDs, bank account and routing numbers [yikes], email addresses, phone numbers, and so on." In other words, a really bad breach. "Each of those is one query away from the

[PRIMARY_IDENTIFIER]. No guesswork is required. The [LINKING_KEY] is a direct link your own engineers built into the schema.

"Step 3: The hashed [PRIMARY_IDENTIFIERS] were not real protection. Some tables stored [PRIMARY_IDENTIFIERS] as cryptographic hashes (SHA-256) instead of plain text. But [PRIMARY_IDENTIFIERS] are only [X] digits and only roughly [X] million possible values. A single computer can hash every possible [PRIMARY_IDENTIFIER] in under [X] minutes, producing a lookup table that maps every hash back to the original number. We reversed millions of them in minutes. If these had been hashed in a cracking-resistant algorithm, we would not even have bothered trying. We would have needed a datacenter's worth of compute power running full blast for months to make a real dent in them. That's impractical. It's worth noting that user passwords were properly hashed. So again, your team knew how to do this, but chose not to apply it to other data."

Finally, "Step 4: The encoded [PRIMARY_IDENTIFIERS] were even weaker. Your tables stored [PRIMARY_IDENTIFIERS] with a simple character substitution, that is, each character shifted by a fixed amount - '1' becomes '9,' '2' becomes ':' and so forth. A one-line script reversed [NUMBER] of these instantly. (This is known as a Caesar cipher, and it's from ancient Rome. It is not secure.)

"What this means functionally is that starting from any single customer, one query produces a complete identity package. The [PRIMARY_IDENTIFIER] results in a name, date of birth, address, driver's license, bank account, email, phone, employer, income, credit score, security question answer, password hash, full loan history, and for hundreds of thousands of your customers, verbatim notes about the most difficult moments of their lives. The data warehouse was designed to work this way. We're happy to answer any more questions at your request."

So GuidePoint's feeling is that there's no way this FulcrumSec group could have possibly performed all of this reverse-engineering work on the downloaded database material given the time they observed. They had to have used the speed offered by AI.

Guidepoint concludes their examination of this specific FulcrumSec event by writing: "Additionally, FulcrumSec used LLM-generated language during their negotiation with the victim, communicating in language clearer and more precise than any typically observed for non-native English-speaking threat actor groups. The language helped the group anchor their position and drive negotiations from their side, in effect saying: 'We know what we have taken here. This is what it is, and this is why we've have set the ransom at this amount.' This is markedly different from most threat actor negotiations, where operators commonly use open-source platforms like Crunchbase or Zoominfo to establish ransom amounts based on market data. By applying LLM capabilities analytically rather than generically, FulcrumSec established a firm negotiating stance from which they had little incentive to diverge."

Okay. So there's the first of two concrete examples. Then they look at a group known as "DragonForce," and they write: "Where FulcrumSec used LLMs to process and weaponize data, DragonForce demonstrates a second and equally significant use case: deploying LLMs to manufacture plausible pressure that would otherwise require capabilities the group does not have.

"DragonForce is an established ransomware-as-a-service group previously covered by GRIT (see our Q2 2025 report). Building on that prior analysis, GRIT has observed DragonForce incorporating AI and LLMs into its operations, a meaningful shift from its earlier tradecraft. Most notably, during negotiations and in advertisements for potential affiliates, the group has claimed to have legal counsel on staff. The statement, which is almost certainly false, is designed to pressure victims by implying that DragonForce has

insight into a victim's reporting requirements and legal exposure arising from the data leak.

"The notion of a criminal ransomware group retaining attorneys fully versed in international data requirements is absurd, until you realize the 'lawyer' is an LLM. For criminal purposes, it doesn't matter if the claim is true; it only matters if it sounds plausible. If there's one thing LLMs are good at, it's making a wide range of statements sound entirely plausible.

"So what does this mean? AI and LLM use gives threat actors a structural advantage in negotiations. They significantly reduce language barriers, increase negotiation professionalism, and amplify available psychological pressure to bear against the victim. Historically, analysts could use imperfect non-native English as a soft attribution marker of adversary geographic location. Even tools like Google Translate would leave tell-tale signs. But contemporary LLM reduces or even eliminates that signal entirely. This is not a marginal development. Attribution confidence decreases, negotiation dynamics shift toward threat actors, and the gap between sophisticated and unsophisticated groups narrows in ways that make victim preparation critically more important.

"More broadly, increased threat actor AI/LLM use reinforces the efficacy of the RaaS (Ransomware as a Service) business model. Conti pioneered the RaaS model, structuring affiliate programs and playbook-driven syndicate operations that set the template that's now being further professionalized and automated by AI tooling. AI and LLMs allow less sophisticated and non-native English speaking groups to approach negotiations in a more professional manner, establishing negotiations with unprepared victims on their terms.

"GRIT will revisit this topic throughout the year to assess the question: Will threat actors continue refining AI/LLM integration into their processes, will it plateau, or will the use of AI/LLMs be more limited to specific groups? GRIT anticipates threat actors will continue to streamline LLM usage in the near-term, primarily through the two vectors: negotiation communications and exfiltrated data analysis. More complex, sophisticated, or novel adoption of AI and LLMs will almost certainly be more limited, but may 'trickle down' in the long term.

"So what are the next steps for defenders? What do defenders do? Threat actor adoption of AI/LLM tooling raises the floor for negotiation sophistication across the board. It reinforces organizations' need for cybersecurity insurance, legal counsel, and an understanding of the data present in their environment. It also suggests that negotiations should be conducted by trained professionals. While threat groups do have some predictable behavior, individual operators are criminals who may act erratically, causing dire consequences for the victim organization. Engaging qualified professionals gives organizations a clear understanding of threat actor playbooks and current behavior, enabling them to distinguish routine bluffs from credible threats.

"Expert legal counsel is also essential for understanding reporting requirements and potential legal ramifications. A mature and up-to-date incident response plan can assist with the coordination of all these factors."

Okay. So that was GuidePoint's example of two very real-world threats. Their report was a bit more sanitized than I was hoping for. So I did a bit more digging to find some additional reporting on FulcrumSec, that first group GuidePoint discussed. The reporting I found added some interesting information.

It said: "As an example of the consequence of this group's use of AI, in June of last month, FulcrumSec reached out to DataBreaches.net regarding their compromise of the Danish pharmaceutical company Novo Nordisk, the maker of Wegovy, a well-known

Semaglutide GLP-1 agonist drug. FulcrumSec claimed to have stolen 1.3TB of data containing" - wait for it - "700,717 files."

Leo: Yikes.

Steve: So I'll briefly note that this is a textbook example of wondering what to do with the presumed treasures that were just plundered from the victim. On the one hand it's "Hot damn! We just sucked out 700,717 individual files with an aggregate size of 1.3 trillion bytes." But now what? Hopefully there's some really juicy data that we can use for blackmail extortion. But where would it be, exactly, hiding among - think of it - .7 MILLION individual files?

Okay. So continuing, they wrote: "FulcrumSec said it had captured valuable intellectual property including five publicly undisclosed drug programs, in-development drug and RNA delivery programs, and private AI models for particular medical and drug discovery purposes. The group told DataBreaches that it used a team of AI agents to analyze those private models, and that it believes the stolen data could save competitors three to five years of program development. Its initial ransom demand to Novo Nordisk was for 25 million USD.

"The information about the intellectual property FulcrumSec stole was coupled with a description of Novo Nordisk's security posture, which the group claimed was 'absolutely catastrophic' and 'boggles the mind.' To us," they write, "this sounds like FulcrumSec is attempting to frame the incident in a way that would have any class-action lawyer salivating. It wouldn't be the first time a data breach has resulted in a lawsuit, so presumably this is part of FulcrumSec's extortion pitch. Their modus operandi also includes using AI to generate detailed reports which it then provides to threat researchers and journalists, nicely formatted, complete with logo and all, in order to apply more pressure to victims.

"For example, after compromising the technology company Avnet in October of last year, the group gave the vx-underground X account a report on the breach. According to vx-underground, the group provided 'an autobiography, a breakdown of the data they possess, their motives for the compromise, information on their logo design and why their logo was chosen, a complete stolen file listing of the compromise, a breakdown of the files (what it is, what they are, what they contain) and images of the files.

"vx-underground said the group had done 'every bit of research and write-up for us.' To add insult to injury, FulcrumSec claimed it had used an OpenAI key it had stolen from the victim to pay for the ChatGPT summarizing the victim's own data."

So I started out noting that "Necessity is very often the mother of invention." Since none of us are on the inside of any of these ransomware gangs, we have a difficult time imagining what their problems might be. So the world comes up with "Hey! They're probably going to use AI just like software publishers will, to discover previously unknown vulnerabilities and then use those to compromise systems."

While that will doubtless be one use, the evidence suggests that the bad guys don't need new ways of getting into other people's networks as much as they need help after the data has been successfully exfiltrated and is in their hands. They need AI's help with determining the value of what they just grabbed and then help negotiating with the data's legal owners who almost certainly speak a language they do not. Appearing tough, competent, and knowledgeable is every bit as important after the threat as obtaining the stolen goods was in the first place. They have, after all, ZERO interest in the data itself that they've just obtained. Its entire value to them lies in what cold hard cash they can

trade for destroying that data they now hold. And for that, AI has been the best thing that ever happened to them.

Leo: Wow. Yeah, I mean, that's the promise of computing, I guess, and AI is just making it easier.

Steve: Imagine you exfiltrate 700,000 proprietary files of Novo Nordisk, and you uncover five other drug programs that are in development, and enough detail to say, well, you know, you've got some competitors who'd probably like to see all this. What's it worth to you for us not to give it to them? 25 million seems cheap to me.

Leo: Yeah. Actually, Novo Nordisk is in the process of going after Lilly because they don't like Lilly, the competitor, who makes Zepbound and a bunch of competitors of Wegovy, and Mounjaro, which is a competitor to Ozempic. They're saying false advertising. And so these two are in a fight. I could easily see Lilly saying, well, let's just see what you're up to.

Steve: Uh-huh.

Leo: They wouldn't do that publicly in any way because of course that would be a big no-no. But you could see there might be some interest. Wow. Steve, again, you've both terrified and amused.

Steve: That is our goal every week.

Copyright (c) 2014 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details: http://creativecommons.org/licenses/by-nc-sa/2.5/