Security Now! #1004 - 12-10-24

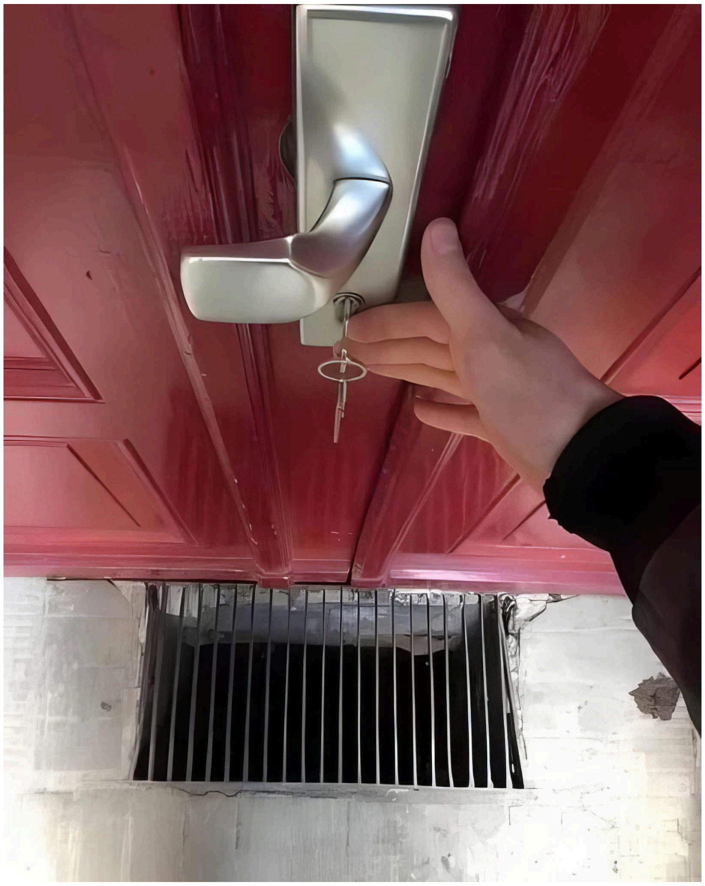
A Chat with GPT

This week on Security Now!

All telecom providers have been hacked and may still not be safe to use. So now the government is recommending that we use our own encrypted communications. The plan to obsolete all non-TPM 2.0 PCs remains well underway. Microsoft must be feeling the heat, so they're taking time to not apologize. Whoops. Microsoft's product activation system has been fully hacked. All Windows and Office products may now be easily activated without any licensing. Here come the AI patents. Apple patents AI recognizing people by what they're wearing after earlier seeing their faces and noting what they're wearing. Zoom wasn't encrypting they're early video conferencing. They're still trying to get out from under the mess their lies created for them. AWS introduces physical data terminal locations where users can go to perform massive data transfers to and from the cloud. The FTC has set their sights on data brokers. Let's hope something comes of it. GRC's email finally gets BIMI. (Can you see the Ruby-G logo?) Lot's a terrific listener feedback about authenticator policy, a new and free point-to-point link service, Tor's "Snowflake", linking PCs and Smartphones, and even recharging spent SodaStream canisters. Then we look at a recent conversation I had with "ChatGPT 4o with canvas" and the new plan that resulted.

This photo demands the caption:

"WHAT ... Could Possibly Go Wrong?!"



Security News

Salt Typhoon

For the past several months there have been various news reports of Chinese state-sponsored attacks against this or that U.S. telecommunications company. I haven't mentioned them because we've had so much else to talk about. But last week, Anne Neuberger, the U.S. Deputy national security adviser said that at least eight U.S. telcos have been hacked and that the U.S. Is now getting set to take some concentrated definitive action. So I think we need to do a bit of catching up.

The best reporting I found on this was headlined *"Chinese hack of global telecom providers is 'ongoing,' officials warn."* with the subhead *"Officials from the FBI and CISA say the major Chinese hack began in late spring, and they are strongly urging Americans to use encrypted communications."* The reporting of this says:

Last Tuesday, federal officials said that the federal government began investigating a major Chinese breach of global telecommunications systems last spring and they further warned that the intrusion remains *"ongoing"* and that it's likely larger in scale than previously understood.

The hack was first announced publicly in October and has been attributed by U.S. agencies to a Chinese government-linked hacking group known as Salt Typhoon. The effort targeted dozens of telecom companies in the U.S. and globally to gain access to U.S. political leaders and national security data. Neither the timeline of the hacking effort nor the scope of the intrusion were previously disclosed.

Jeff Greene, executive assistant director of cybersecurity at CISA, and a senior FBI official said Tuesday that while agencies started cooperating on their investigations of Salt Typhoon's activities in early October, the effort was first detected in *"late spring and early summer."* He also warned that the breach is *"ongoing"* and that there was much law enforcement still did not know.

Greene said: *"We cannot say with certainty that the adversary has been evicted. We're on top of tracking them down … but we cannot with confidence say that we know everything, nor would our partners."*

Greene strongly urged Americans to *"use your encrypted communications where you have it,"* adding that *"we definitely need to do that, kind of look at what it means long-term, how we secure our networks."* Yikes. That's definitive. And notice the irony of the government telling its citizens that they need to use their own encrypted communications apps wherever possible because the networks of the telecommunications providers are insecure and there doesn't appear to be a lot that can be done about that. Ironic, because our governments have been chafing over their citizens' use of these same encrypted applications which the government is unable to penetrate.

But get this: As many as 80 – eight zero – telecommunications companies and internet service providers, including AT&T, Verizon and T-Mobile, are believed to have been infiltrated in the hack. T-Mobile was the most recent one in the news.

Earlier last Tuesday, CISA, the FBI, the National Security Agency, and partner agencies in New Zealand, Australia and Canada released a joint alert warning that Chinese hackers were targeting *"major global telecommunications providers."* Officials declined to comment on specifics, but acknowledged that *"there were servers used in various countries to facilitate this activity by the Chinese."*

The United Kingdom did not sign on to the alert, making it the only nation in the Five Eyes intelligence-sharing group to be omitted. Greene attributed this to each country having *"different considerations and timelines."* A spokesperson for the U.K.'s National Cyber Security Centre said Tuesday that the agency *"support[s] our international partners issuing this advisory to help improve the collective resilience of telecommunications infrastructure,"* and that the U.K. has a separate approach to mitigating cyber risks to its telecom providers. Hmmmm. Okay.

The officials from the FBI and CISA noted in their briefing that there were three groups of victims targeted in the hacks. The first group was an undisclosed number of victims, mostly in the *"United States Capital Region,"* according to the officials, who were impacted by stolen call records from telecom companies. The second group — a small number of political or government-linked individuals, all of whom have been notified by officials — had their private communications compromised, according to a senior FBI official who spoke anonymously as a condition of briefing reporters.

While the officials did not specify how many individuals were targeted, it was previously reported that the phones of President-elect Donald Trump and Vice President-elect JD Vance were among those compromised, in both cases prior to the U.S. national election. In many cases the voice and textual content of call connections and conversations were obtained by Chinese attackers.

In addition, the Chinese hackers also accessed and copied U.S. court orders, which the FBI official said were attained through the Communications Assistance for Law Enforcement (CALEA) statute program. This program allows law enforcement and intelligence agencies to submit court orders around intelligence collection from telecom providers.

When pressed on whether hackers were able to access court orders for intelligence collected under the Foreign Intelligence Surveillance Act — which allows U.S. intelligence agencies to collect data on foreign targets — the FBI official declined to answer directly but acknowledged that *"the CALEA environment does include court orders"* for FISA investigations.

The major hacking campaign has been an issue of increasing concern for U.S. lawmakers in recent weeks, with Senate Intelligence Committee Chair Mark Warner describing it as the *"most serious breach in our history."* And Senator Mike Rounds, ranking member of the Senate Armed Services Committee's cyber subcommittee, said during a panel at last month's Halifax International Security Forum: *"Unless you are using a specialized app, any one of us and every one of us today is subject to the review by the Chinese Communist government of any cell phone conversation you have with anyone in America."*

I think this news highlights the clear need for independent third-party end-to-end encrypted video, voice and text messaging systems. We're being told that the conversational content, not just connection metadata, of anything carried by our international telecommunications carriers can no longer be considered to be secure from eavesdropping by advanced persistent threat actors who want to know what's being said.

If nothing else, this news, which has now been officially recognized, weakens any argument against allowing users of public telecommunications systems from providing and using their own truly secure end-to-end encryption for their conversations and content. The analogy is to the Internet. The Internet is a similar public network which is not, itself, secure. So to it we've added a layer of authenticated TLS encryption to enable point-to-point end-to-end communications security – and no one has a problem with that. What's the difference? And what's the big deal?

TPM 2.0 – and we're not kidding

A posting to the Windows IP Pro Blog last week was titled: "TPM 2.0 – a necessity for a secure and future-proof Windows 11". I titled this bit of news "TPM 2.0 and we're not kidding." I'll give everyone a sense for this by sharing just the first few paragraphs of this quite lengthy posting by Steven Hosking whose info on 'X' identifies him as "Senior Program Manager for Windows Comercial [sic] at #MSFT." He wrote:

> *With Windows 10 end of support approaching, it's important to revisit a key minimum system requirement for Windows 11: Trusted Platform Module (TPM) 2.0. Let's discuss the role of TPM and its value for those of you who have made the transition to Windows 11. You'll also learn how to check your TPM status and how to prepare for Windows 11.*
>
> *TPM refers to a dedicated chip or firmware that offers hardware-level security services for your device. It securely houses encryption keys, certificates, passwords, and sensitive data, shielding them from unauthorized access. Additionally, TPM is tasked with cryptographic operations such as producing random numbers, encrypting and decrypting data, and confirming digital signatures. TPMs are available from many different manufacturers, including Microsoft on supported CPUs with Pluton.*
>
> *You know that Windows 10 is approaching end of support. In Windows 11, TPM 2.0 advanced encryption techniques offer more versatile and critical key management for contemporary IT infrastructures, as compared to its predecessor, TPM 1.2. Integrating with features like Secure Boot and Windows Hello for Business, TPM 2.0 enhances security by ensuring that only verified software is executed and protecting confidential details. It's true that its implementation might require a change for your organization. Yet it represents an important step toward more effectively countering today's intricate security challenges.*
>
> *TPM 2.0 helps keep your identities more secure and your data protection more robust. Can you ensure operating system integrity upon startup? Yes. Can you better protect sensitive information, data, and secrets? Yes. It provides a vastly more efficient and secure platform for Windows 11 to use through advanced encryption methods, improved industry standard cryptography, increased isolation, and greater interoperability with other security functions.*

Okay. So is TPM 2.0 really better than TPM 1.2? Yes it is, without a doubt. It offers newer, updated cryptographic operations such as elliptic curve crypto and 256-bit SHA2-era hashing and message authentication functions instead of just SHA1. And it offers a privilege management hierarchy rather than just the single level offered by TPM 1.2.

But here's the problem: While 2.0 is without a doubt new and improved and should be adopted and used going forward, there's never been anything found wanting about TPM 1.2 that might force its abandonment – as we've observed from the beginning, this is an arbitrary requirement. TPM 1.2 had been working just fine for everyone, and still is, until Windows 11 came along. I would have no problem Windows 11 taking advantage of the more secure features available from 2.0 if and when they were available. But it should be up to Windows' users whether or not they feel they need to upgrade their PC hardware to obtain that security under Windows 11.

Steven, wrote: *"It's true that its implementation might require a* ***change*** *for your organization."* Right. "A change". What he meant is that the move to Windows 11 may forcibly obsolete all of an organization's current stock of PCs which are otherwise, right now, still quite happily running Windows 10. But none of those machines will run Windows 11 and Microsoft's continuous IV drip of life-support to continuously repair the apparently endless supply of serious security bugs in Windows 10 will be coming to an end next October.

As we covered previously last Halloween, enterprises & individuals will have the option of paying for extended life support – for up to three more years in the case of enterprises, though it becomes quite expensive. Nevertheless, switching is always difficult and I would not be surprised to learn that many of our listeners or their organizations were not seriously considering either paying to stay with Windows 10 on their current hardware or perhaps switch to the arguably superior alternative offered by 0patch.

It rubs me the wrong way for Microsoft to be charging its customers to fix security flaws in its own products when it is already fixing them anyway and has a well-running system in place that allows those fixes to continue being delivered. What Microsoft is planning to do next October is to deliberately disable the existing Windows Update for Windows 10 users who choose not to pay to have Microsoft continue to repair their own software flaws. What's wrong with this picture? As we noted last week, the United States Government recently opened a broad antitrust investigation into Microsoft's abuse of its monopoly power. So Microsoft choosing to force the obsolescence of hundreds of millions of PCs – or hold their customers ransom over fixing those software flaws – could not come at a better time.

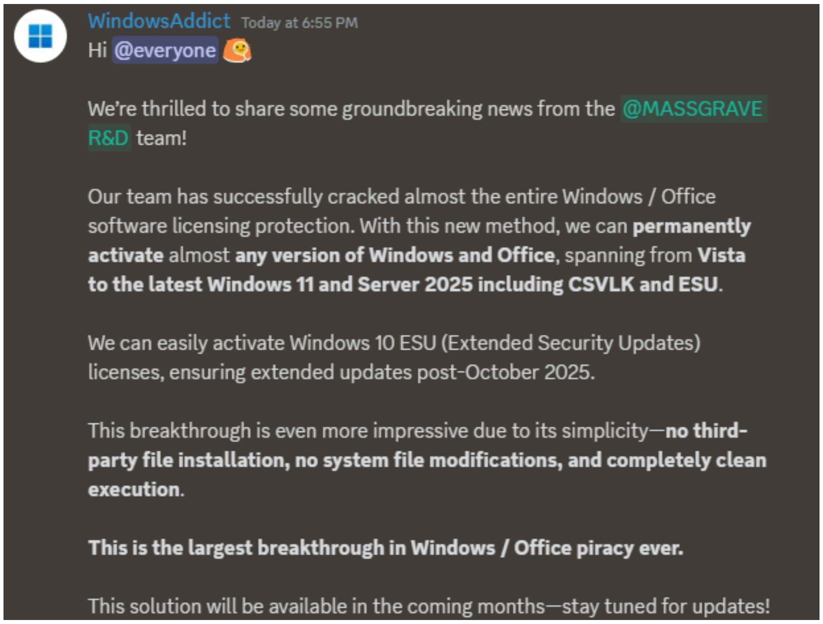
We've seen that it's possible for Microsoft to examine its own behavior and change when it's shown to be wrong. In the case of their cloud computing security, they were previously offering paid security enhancements through logging that should have been included at no charge as part of the base offering rather than being disabled by default. Once it became clear that this conduct was unusual and wrong, they began including those additional services free of charge. October is still 10 months away, so there's time for another policy change regarding the future of Windows 10 and 11.

Microsoft's Activation System (More Fully) Hacked:

And while we're on the topic of Windows, Martin Brinkman, writing for GHacks, titled his piece *"Hackers claim to have **cracked** Microsoft's software licensing protection almost entirely."* He writes:

> *A team of hackers claim that they have cracked "almost the entire Windows / Office software licensing protection." The breakthrough allows them to activate "almost any version of Windows and Office" permanently. Windows and Office installations require activation. This may happen behind the scene or when users enter product keys. Workarounds and hacks have been available for a long time. One popular choice requires running a single line of instructions from a PowerShell prompt to activate Windows 8 or later, or Office.*
>
> *The creators of the solution claim that they have found ways to extend this to even more Windows and Office products. The new method works on any Windows client or server version and includes Extended Security Updates (ESU) and Microsoft Customer Specific Volume License Keys (CSVLK). The method used up until now could not activate everything permanently. But now, for the first time, the versions that had remained elusive have been supported for the first time: Windows 7, 8 and 8.1, any recent Windows Server, Add-ons and Extended Security Updates have been added.*
>
> *The hack, for example, enables support for Windows 10 ESU, once it starts in October 2025. The hackers claim that the discovered method is simple. It does not require third-party file installations or system file modifications according to a post on X:*

I've captured their posting to 'X' which was by @MASSGRAVE. In this instance "MAS" stands for Microsoft Activation Scripts:



WindowsAddict Today at 6:55 PM
Hi @everyone

We're thrilled to share some groundbreaking news from the @MASSGRAVE R&D team!

Our team has successfully cracked almost the entire Windows / Office software licensing protection. With this new method, we can **permanently activate** almost **any version of Windows and Office**, spanning from **Vista to the latest Windows 11 and Server 2025 including CSVLK and ESU**.

We can easily activate Windows 10 ESU (Extended Security Updates) licenses, ensuring extended updates post-October 2025.

This breakthrough is even more impressive due to its simplicity—**no third-party file installation, no system file modifications, and completely clean execution**.

This is the largest breakthrough in Windows / Office piracy ever.

This solution will be available in the coming months—stay tuned for updates!

https://x.com/massgravel/status/1862492822261399731

PowerShell MAS Scripts: https://github.com/massgravel/Microsoft-Activation-Scripts

And since none of this is now any sort of secret – their posting on 'X' has more than 900,000 views – this week's podcast show notes also has the Github link to the PowerShell scripts they also posted. I looked at them and I found an extremely complex and detailed PowerShell script that appears to incorporate explicit support for the activation of just about everything. I didn't spend much time with it since I have no particular interest in any of this. But it's news and I'm sure that many frisky script kiddies out there are already enjoying many hours of play with this revelation. Martin's finishes his reporting by writing:

> *An example screenshot of a fully, permanently activated version of Windows with Extended Security Updates has been shared as part of the post. The methods have worked for years, according to one of the follow-up posts. The hackers claim that the digital license (HWID) method worked since 2018 and the KMS method for at least 17 years. The discovered hack will be made available in the coming months, according to the original post on X.*
>
> *The discovery is a serious blow for Microsoft, provided that the hack is indeed as foolproof and easy to apply as claimed. It is unclear how, or if, Microsoft will react to the hack. For now, it seems that the hackers have, at least temporarily, won the battle.*

Since I've been a paid-up MicroSoft Developer Network (MSDN) developer for decades, I already pay for the privilege of installing whatever Windows editions I need for software development and testing. But it's going to be interesting to see how this develops over time. I never thought much about cracking the Windows Activation System, but it has obviously been something of a preoccupation for some segment of the hacker community.

US Patent #: 12154386
A patent just granted to Apple with the title *"Identity Recognition Utilizing Face-Associated Body Characteristics"* serves to give us some sense for where and how future AI will become packaged into consumer devices.

The gist of the patent is that from the standpoint of a fixed security camera, someone's face provides the most useful recognition detail. But the camera might not always be able to see the person's face. So while the camera IS seeing the person's face, it will also be taking note of the clothes they're wearing at the moment – that day – and other things such as body dimensions and their walking gait. Then, that person may later be recognized not by their face, which might not be visible, but by association with the other available characteristic details that had been previously noted at an earlier time when their identity could be positively determined.

It's employing roughly the same sort of strategy a human observer would employ. And the fact that the US Patent and Trademark Office granted Apple a patent on this suggests that the AI revolution is going to further swamp the already buried USPTO as people apply for patents on a gazillion other seemingly obvious things that AI will soon be making commonplace.
https://image-ppubs.uspto.gov/dirsearch-public/print/downloadPdf/12154386

Zoom offering $18 million to settle

Mashable caught an interesting story last week. Their piece was titled: *"Zoom lied about encryption in 2020. Now it wants to pay $18 million to make that go away."* and they tagged that with the subhead: *"The internet never forgets, though."* Mashable wrote:

> *Back in 2020, Zoom was one of the hottest software companies in the world. Its video conferencing software surged in popularity due to millions of people being confined to home offices due to the COVID-19 pandemic.*
>
> *Unfortunately, the company cut some corners when it came to the privacy of its users. Despite Zoom's claims that its video meetings are end-to-end encrypted, it later came to light that this was not true. The result was a class action lawsuit that Zoom settled for $85 million. In 2021, Zoom also settled with the Federal Trade Commission over misleading its users about the privacy and security of its core product.*
>
> *But the matter did not go away entirely. There's also the separate matter of a U.S. Securities and Exchange Commission (SEC) probe into Zoom's privacy policies, which the SEC launched in 2020. Now, Bloomberg reports that Zoom is offering to settle the matter with the SEC by paying an $18 million fine. The offer is still pending approval by the SEC.*
>
> *These days, Zoom does offer end-to-end encryption for its video meetings, and its privacy and security practices have improved. But back in 2020, the company's track record was poor, with Zoom bombings — instances of people hijacking other people's Zoom calls and harassing them — becoming something of a trend.*

And the mashable article finishes by noting:

> *By the way, you may have missed it, but Zoom is no longer called "Zoom Video Communications," which was its official name until Monday. The company is now officially called* ***Zoom Communications*** *to reflect the fact that it now offers a suite of communications tools beyond its videoconferencing platform.*

We spent a lot of time covering Zoom back during its explosive growth, and we knew that its security was stumbling a lot during those early days. I recall that we talked about those so-called "Zoom Bombings", but I don't remember whether we actually knew they were lying at the time about their video conference calls being truly end-to-end encrypted.

AWS Physical Data Transfer Terminals

One of the problems posed by cloud services, especially in this era of “Big Data” – where “Big” can increasingly mean “really ridiculously big” – is the question of how to seed the cloud by transferring massive amounts of data to and from a cloud provider. To answer that need, Amazon has launched the first of their so-called “AWS Data Transfer Terminals”. Here’s what Amazon explained on December 1st under the headline “New physical AWS Data Transfer Terminals let you upload to the cloud faster”:

> *Today, we’re announcing the general availability of AWS Data Transfer Terminal, a secure physical location where you can bring your storage devices and upload data faster to the AWS Cloud.*
>
> *The first Data Transfer Terminals are located in Los Angeles and New York, with plans to add more locations globally. You can reserve a time slot to visit your nearest location and upload data rapidly and securely to any AWS public endpoints, such as Amazon Simple Storage Service (Amazon S3), Amazon Elastic File System (Amazon EFS), or others, using a high throughput connection.*
>
> *Using AWS Data Transfer Terminal, you can significantly reduce the time of ingesting data with high throughput connectivity at a location near by you. You can upload large datasets from fleets of vehicles operating and collecting data in metro areas for training machine learning models, digital audio and video files from content creators for media processing workloads, and mapping or imagery data from local government organizations for geographic analysis.*
>
> *After the data is uploaded to AWS, you can use the extensive suite of AWS services to generate value from your data and accelerate innovation. You can also bring your AWS Snowball devices to the location for upload and retain the device for continued use and not rely on traditional shipping methods.*
>
> *You can find the availability of a location in the AWS Management Console and reserve the date and time to visit. Then, you can visit the location, make a connection between your storage device and S3 bucket, initiate the transfer of your data, and validate that your transfer is complete.*
>
> *On your reserved date and time, visit the location and confirm access with the building reception. You will be escorted by building staff to the floor and your reserved room of the Data Transfer Terminal location. Don’t be surprised if there are no AWS signs in the building or room. This is for security reasons to keep your work location as secret as possible.*

Wow. This is the sort of thing that makes sense after you hear of it. Once these AWS terminals are available in many major metropolitan areas it’s easy to imagine them being quite popular.

The FTC takes action against data brokers

The GRC.SC shortcut I created to quickly take people to the PenTester website to allows anyone to quickly check for their data among that which was leaked by National Public Data is the #1 most clicked shortcut of all time. It’s grc.sc/npd and when I checked just now it had been used 12,394 times since it was created on August 20th.

My point here is that, as I’ve observed since, unregulated data brokers – just by their very existence – represent a clear and present danger to society at large. So I was glad to encounter the news that the US Federal Trade Commission had taken regulatory action against two US-based data brokers. The FTC has banned Mobilewalla, Gravy Analytics, and its subsidiary Venntel from selling the geolocation data of their users. The FTC cracked down on the three companies after they were caught collecting and selling the information they had aggregated without their customers' consent. The FTC said that the data contained information about military sites, churches, labor unions, and other sensitive locations and the FTC specifically singled out Mobilewalla for selling geolocation data to identify women who visited pregnancy centers and individuals who attended George Floyd protests.

It's difficult to find any sympathy for such parasitic companies.

By the way, I also noted that the second most popular GRC shortcut was grc.sc/pin which took our listeners to that wonderful graphic heat-map chart which clearly showed the extremely non-uniformity in the 4-digit PINs chosen by PIN users. We have a lot of fun on this podcast.

Miscellany

BIMI (finally) arrives

Late last week I received notification from DigiCert that they had approved my use of GRC's "Ruby G" logo for display in GRC's BIMI-certified email. Our "BIMI up Scotty!" podcast was back on October 15th, and at the time the Internet Archive – which the entire industry uses for this purpose to verify the long term use of corporate logos – was suffering a long running and debilitating series of DDoS attacks. I'm sure that if my need was urgent I could have reminded DigiCert before now and pushed the matter. But they did get back to it without me needing to do so.

When I checked last week after receiving that notice, the certificate's status was in "awaiting final" status. I'm mentioning this today because I awoke to the news that GRC's Verified Mark Certificate – which was the goal of all this – had been approved and issued. Although I could have hosted the pair of SVG and PEM files from my GRC domain, I decided that it might seem more official if they came from DigiCert themselves, though I doubt that it matters either way. But since they were pleased to offer to host the files I took them up on that offer.

So yesterday (Monday) morning I added a BIMI TXT record to GRC's DNS which contained the twin URLs for GRC's logo image and its matching certificate. From that moment, GRC's received email was BIMI enabled. Since I didn't yet have this week's email ready, I sent last week's email to my Gmail account, since Gmail is one of the providers that supports the display of BIMI. And sure enough, there was GRC's "Ruby G" decorating the opened mail. I imagine that everyone who receives this week's email, and all subsequent email from GRC through BIMI-supporting providers, will also receive the same thing, whether they notice it or not.

I was curious to see whether the authentication change would have a retroactive effect, so I went back a week in my Gmail to find last week's originally sent email. I have my Gmail account subscribed to the podcast mailing for exactly this sort of testing. Interestingly, the new GRC logo was NOT also shown on that piece of older email. That's interesting, since the email itself does not carry any hint of whether the mailing domain may have a verified BIMI logo. So it appears that Google is checking for the BIMI record at the same time as it's verifying the mailing site's SPF, DKIM and DMARC status. And once that's done and the email is received, the logo is either established or it won't ever be.

In any event, GRC now has BIMI certification and may be held in somewhat higher regard by any major mail receiving sites that care.

Closing The Loop

Jaime Denizard *(pronounced hi-meh)*

Steve, I've been using Google Authenticator (with cloud backup disabled!) for years but I would like to use a more featureful solution and one preferably not run by Google. The main feature I'm looking for is a solution that has a web portal so that I can get TOTPs from any browser instead of needing my phone with me at all times. How much security would I be giving up, if any, if I went with a solution that offered this such as Bitwarden Authenticator, Ente Auth or Twilio Authy? Thank you and keep up the great work, Jaime.

I chose Jaime's note because it's a question many people have. They want the added security of a second factor but they don't want the added inconvenience. We've talked about the inherent danger of merging all authentication into a single source. For example, of having one's password manager also supplying the one-time passcode second factor. Is it **as** secure as maintaining an entirely separate 2nd factor authenticator and then transcribing the 6-digit code manually? No. Is it **more** secure than not bothering with any second factor? Absolutely. It all boils down to security models and asking the question "what exactly are we wishing to protect against?" We need to ask that question because, unfortunately, there are many different points of potential vulnerability.

Let's address three cases: A full breach of the site being authenticated to, a breach of only the site's known usernames and passwords, or a breach of a user's computer.

In the first case of a full breach of the site being authenticated to, the only form of authentication that remains safe after such a full site breach is Passkeys. Passkeys remains safe because, being a public key authentication system, as I used to say of SQRL but I'll now say of Passkeys: *"Passkeys gives sites no secrets to keep."* The only thing a site can do with the public key it has received from its user is verify their identity. It cannot be used in any way to assert or spoof their identity.

One-time passcodes will not protect their users after a full site breach because one-time passcodes rely upon a shared secret. It's that secret which determines which 6-digit code is correct every 30 seconds. So if bad guys are able to obtain the usernames, the password hashes and the shared secret one time password seeds, they'll be able to impersonate the site's users. And even if the site is storing its users passwords as salted hashes, as any modern site now should, a credential stuffing attack that's backed up by having each account's matching 2nd-factor seed would still be able to succeed.

So, to recap: In the event of a full site breach, traditional second factor authentication, which relies upon the continued secrecy of a shared secret "seed key", would provide no added protection. So it would not matter where your own authenticator is storing its secret.

In the second case of only a breach of a site's usernames and hashed passwords – or even without any breach, just guessing usernames which are increasingly email addresses, the bad guys would employ a so-called credential stuffing attack. That's the newer fancy sounding name that we used to call "brute force attacks", though credential stuffing suggests that the stuffer is not just guessing at random but is, instead, working from a list of known possible credentials that had been previously harvested from some other source. And this is where reusing passwords between sites becomes a very bad idea.

However, in this case, since the bad guys would not have obtained any of the site's stored 2nd factor authentication secrets, the use of a 2nd factor authenticator would strongly protect the user’s account. And, again, **where** the authenticator is running – whether in the user’s browser or offline in a separate smartphone – would make no difference since the bad guys would have no way of guessing the continually changing 6-digit passcode.

So to recap: in both of the two previous instances of attacks – a full site data breach or one of the increasingly common credential stuffing attacks – the location of the user’s authenticator secrets has no impact and makes no difference.

This brings us to the 3rd case: A breach at the user’s end. This could either be a breach of the user’s PC with their web browser and its password manager, or a breach of the user’s smartphone which contains their 2nd-factor authentication secrets. This is the nightmare scenario where the only protection is the separation that hopefully exists between the first and the second authentication secrets.

The presumption is that it’s exceedingly difficult for any bad guys to get into either of the user’s authentication stores – the first or the second factors – because we never see that happen. We’re constantly talking about all manner of horrors on the Internet and with Internet-related technologies. But we never encounter instances where end users are having their local password managers breached. If I had some wood handy I’d knock on it, since we don’t ever want to be reporting that. This substantiates our intuitive sense that it’s safe for us to use password managers because having them broken into is not something that appears to be happening often – if at all or ever. And given that it’s exceedingly difficult to break into ONE credential store, it’s beyond exceedingly difficult to imagine that two separate credential stores using wildly differing technologies – a PC and a Smartphone – might both be simultaneously compromised in order for bad guys to obtain both first and second factor secrets to then facilitate authentication.

In other words, the only danger posed by storing both the first and second authentication factor secrets in the same place, in the same device and thus under the same form of protection, is that the security of that device could possibly, conceivably, be breached. And, moreover, we’re aware of no instances where that has ever happened or has been a problem. So at this point, today, it’s only a theoretical concern and argument. But ... it **IS**, nevertheless, a concern and an argument no matter how theoretical it may be.

This is very much like our recent discussions of whether it’s safe to leave an otherwise unprotected Wireguard VPN service port exposed and listening on the Internet as tens if not hundreds of thousands of people do. As I said last week, it’s very much almost certainly safe. There’s every reason to believe that it is safe and no reason to believe that it isn’t... right up until the moment we learn that it wasn’t.

I’m spending so much time on all of this because there’s an important concept that binds these together. That concept is “layered security.” The idea of layered security is that no **single** fault, vulnerability or compromise in the security of something protecting a system would result in a compromise of that system’s security. Another more colloquial term for “layered security” would be “belt and suspenders.” I would always put Wireguard behind some other form as access control if only so that any failure in either one would not result in a failure of the whole.

And the concept of “layered security” is what gave us multi-factor authentication in the first place – not relying upon any single factor. If one is compromised the other can be trusted to hold. Ideally, the implementation of layered security does not pose an ongoing burden upon its user. And this is where the implementation of the system comes into play. If the machine a user is authenticating from already contains a reasonably fresh previous authentication cookie, depending upon the security needs of the website, it would be reasonable to bypass the request for the user’s second factor and to only ask for it if either a long time has passed since the user last authenticated from that machine or if the user is authenticating from a machine that has no record of previous authentication. This model continues to strongly protect the user from an online credential stuffing attacker whose authentication guesses would not carry the 2nd-factor bypass cookie, while also reducing the annoyance factor to repeat users of the same machine.

So, Jaime... Your question was obviously a good one because it certainly didn't have a short answer. And the answer that it did have is best viewed in the context of the various possible threats that it needs to protect against. Practically speaking, I think a good case could be made for most users to just let their existing password managers painlessly supply their 2nd factor one-time passcodes for them. That provides strong protection against the known online password stuffing style attacks that we know are occurring. And against those attacks it is providing layered belt and suspenders authentication protection. The fact that it is not also protecting from a theoretical attack that we have no evidence of ever having happened, even though protection could be provided by moving those 2nd factor secrets to a different device, is almost certainly taking caution too far ... until it isn't.

Since most of the accounts I use online with my fixed and unshared PCs allow me to remain logged in for months at a time, my own use of 2nd factor tokens which are stored on a separate smartphone has never been a troublesome burden.

Nir Eden

Dear Steve, I've been a dedicated listener of Security Now for many years. Your show has expanded my technical understanding and reinforced important values I deeply believe in - particularly that privacy is a fundamental condition for freedom, accountability to the entire internet community, and unwavering reliability. Regarding remote access solutions: While overlay networks like WireGuard and Nebula work well, they lack granular access control and can be complex to set up. Solutions like Cloudflare tunnel and NGrok provide public-facing interfaces, but I needed something different. I wanted to:

- *Create a private tunnel from my home Raspberry Pi SSH server to my laptop so I could log in from anywhere*
- *Connect a cloud web server to a micro-service that runs on another cloud*
- *Link database servers and clients running on different locations*

I developed a solution based on SSH tunneling through an external server. Since both ends make outgoing connections, opening ports or modifying firewall settings is unnecessary. I have developed a simple web interface, so connecting two devices is as simple as setting up a Zoom meeting. After using it successfully for years to connect cloud services and control remote devices, I've made it publicly available at www.puppetpc.com. It is currently free to use, as I want to see how far this solution can go. Thank you, Nir Eden.

I went over to www.puppetpc.com and took a look around. The site looks very clean and new and I imagine that it will evolve over time. There is not yet any deep technical documentation that I could see. So I know that many of our listeners would need to know why they should trust it. But I'm also aware that others won't care and may just be content to play with whatever it is. So I'm not vouching for it, since I cannot. But I wanted to share this very nice looking creation of one of our listeners, to give Nir some attention to his efforts that might be useful to him, and to reiterate how amazed I am by the quality of the people who choose to spend their time listening to this podcast.

Steven Cedrone (in Canada eh?)

> *Hi Steve, I heard you mention TOR's call for more bridge operators in SN 1003. I wanted to bring to your attention the "Snowflake" extension/add-on for Firefox / Chrome / Brave (or other chrome based browsers). It allows the Tor Network to use your computer as a proxy to help people circumvent censorship and it is as easy as installing a web browser extension/add-on. You can also toggle the settings to allow it to continue running even when the browser isn't open.*
>
> *They're good about not slowing down your internet connection, and they hide your IP address while someone is connected through your computer. The Snowflake also changes from purple to green in colour (if pinned to the toolbar at the top) so you know someone is currently connected.*
>
> *I wanted to mention this to you in hopes people might help the Tor network in this way as well because not everyone has the skills to run a server to run a bridge like I do. (Not the easiest to setup in Linux) Read more about it here:* https://snowflake.torproject.org/ *Kind Regards,*

THAT is very cool. I love that something like this could so easy to set up and be safe to use. The Tor Project folks know what they're doing. One reason I'm very glad Steven put this on our radar is that these days most of us have massive bandwidth overkill with our bandwidth mostly sitting idle. So the idea that we might be able to donate some small piece of it to help the Tor Project and to provide more diffusion seems like a great thing. I followed Steven's link and went over to the Tor Project's "Snowflake" page. It turns out that Snowflake's function as a traffic proxy is only one of the things it's able to do. It can also allow users to use Tor for bypassing censorship. Here's what the Tor Project says about Snowflake:

> *Snowflake is a system that allows people from all over the world to access censored websites and applications. Similar to how VPNs assist users in getting around Internet censorship, Snowflake helps you avoid being noticed by Internet censors by making your Internet activity appear as though you're using the Internet for a regular video or voice call.*
>
> *There are numerous tools available, such as Snowflake, that "transform" Internet activity, each using a different technique. Some redirect Internet traffic to appear to be coming from popular cloud providers like Microsoft Azure and Amazon Web Services. Others scramble Internet traffic in order to make it appear completely random.*
>
> *It therefore becomes costly for censors to consider blocking such circumvention tools since it would require blocking large parts of the Internet in order to achieve the initial targeted goal.*
>
> *Unlike VPNs, you do not need to install a separate application to connect to a Snowflake proxy and bypass censorship. It is usually a circumvention feature embedded within existing apps. Currently Snowflake is available inside Tor Browser on Desktop and Android, Onion Browser on iOS, and Orbot on Android and iOS. If you have downloaded and installed any of these apps, and they are censored in your country, you can bypass the censorship by activating Snowflake through the apps' settings page.*

And then we get to the part that caused Steven to write his note. The Tor Project writes:

> *Did you know that Snowflake proxies are operated entirely by volunteers? In other words, a Tor user gets matched with a random Snowflake volunteer proxy, which is run by a volunteer*

like you! So, if you want to help people bypass censorship, consider installing and running a Snowflake proxy. The only prerequisite is that the Internet in your country is not heavily censored already. You can join thousands of volunteers from around the world who have a Snowflake proxy installed and running. There is no need to worry about which websites people are accessing through your Snowflake proxy. Their visible browsing IP address will match their Tor exit node, not yours. There are different ways to run a Snowflake proxy (beginner to advanced):

Install the web extension: The web extension is the easiest way to run a Snowflake proxy. Simply install it on Firefox, Chrome or Edge, enable the extension, and watch the icon turn green when a user connects through your proxy!

If you would like to run a command-line version of the Snowflake proxy on your desktop or server, see our guide for running a Snowflake standalone proxy.

It's so great that the Tor Project has figured out how to so easily allow others who wish to volunteer some of their bandwidth to help those within repressive regimes to maintain access to the outside world.

John Robinette has a solution for linking Smartphones and PCs:

Hey Steve, With regard to your wish for a way to easily type something on your PC and send it to your iPhone, I would recommend LocalSend (https://localsend.org/). The simplest way to describe it is a cross-platform AirDrop, written in Dart+Flutter, that works on iPhone, Android, Linux, Windows and MacOS. It does require installing an app, but the communication is all local between devices. LocalSend uses mDNS to discover other LocalSend clients on your subnet, which then allows you to send and receive text, files, photos, and so on. I've been using it for about a year to move various files between my Windows PC, iPhone, iPad, and a Linux PC.

If you don't want to install an app, there's also PairDrop (https://pairdrop.net/), which is similar but entirely browser based. The actual transfer of data is peer-to-peer via WebRTC. However, establishing this peer-to-peer connection depends on both clients first making a connection to the website, so it won't work if your Internet connection is down (or you're paranoid about using someone else's server--but it's open source and easily self hosted if you're that person). Hope that one or both of those might be useful for you and others.

I like the looks of the LocalSend solution very much. I've been using iCloud on Windows to link my devices. It works but this might be quicker. Thanks, John.

Jay Soch

Good Afternoon Mr. Gibson, Long time, first time. You got me into Bug Bounty and now I make a not insignificant amount of income through BB. My wife is obsessed with La Croix, and we've spent a lot of money on it over the years. This year I am thinking about getting her a Sodastream like device so she can get her fix more easily and we can hopefully save some money. I remember that you had discussed some techniques you had used to save some money on a similar device on the podcast and I am going to go through the notes and find that information.

> *What I would like to know is if you have any updates to your previous process? As I recall you had changed the adapter on the CO2 cartridge and were getting your CO2 canisters refilled somewhere in Irvine (I'm in Fullerton btw)? Do you still do this? Has this process held up over many uses/years/. I would love any thoughts you have on whether this is a worthwhile investment or not. Thank you for sharing your very valuable time.*

I'll take up just a bit of everyone's valuable time because it has been such a win for us. The trick is to have a single large Co2 master tank that's used to directly refill empty SodaStream canister tanks. This allows you to perform the refilling from home, reusing the SodaStream canisters over and over. And the master tank can, in turn, be filled over and over by any home brewing shop. People who brew their own beer at home use the same tanks and get them refilled.

So the trick is interconnecting the two tanks and Amazon has plenty of adapters for exactly this purpose. They're typically nicely machined brass adapters with a valve. One end of the adapter fits the empty SodaStream canisters and the other end mates with a standard Co2 tank which is also available from Amazon. I believe that mine is a 20 pound tank. That's light enough for me to drag and roll from my car to the shop for refilling and back.

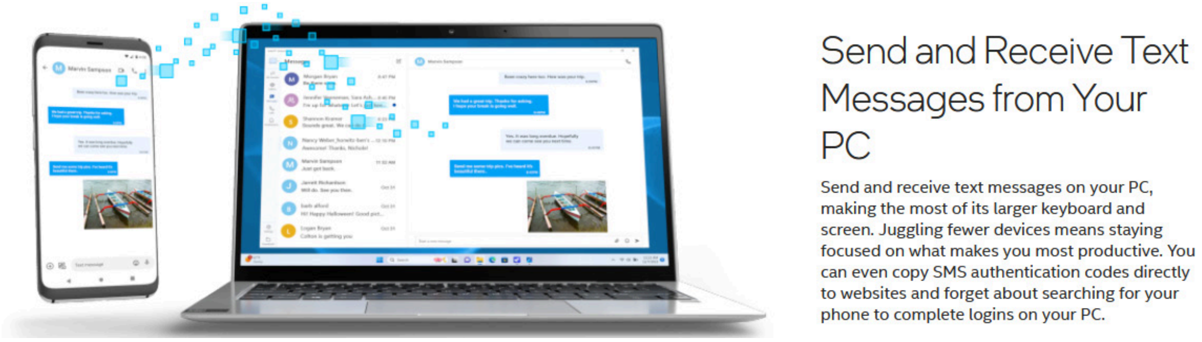
The only requirement for the tank is that you need to be sure to get one with a "siphon tube". The siphon tube extends from the valve on the top of the tank all the way to the bottom and it's what allows you to fill the empty SodaStream canisters with liquid carbon dioxide, taken from the bottom of the tank rather than Co2 gas taken from the top. These tanks are not inexpensive. They'll run you around $150 dollars, but boy has it been worth it for us. I haven't counted the number of times we're able to refill a small canister from the much larger tank, but it's many many. Be sure to check out YouTube. As I recall there's quite a community of people who are doing this because it makes so much sense.

Troy in Montana suggests Intel's Unison

> *Steve, Long time listener since day 1. If you have an Intel PC you can use their app to connect your iPhone to a PC and get access to sending messages. Not perfect but a way to do what you hoped: https://www.intel.com/content/www/us/en/products/docs/unison/overview.html Thanks for all you and Leo do to keep us safe.*

I was excited when I read Intel's description: *"Following a simple pairing process between the phone and the PC, you can make or take phone calls from your PC, send or receive text messages using the PC's mouse and keyboard, and view phone notifications on the PC screen. Also, you can seamlessly and bidirectionally share photos, videos, and documents between your phone and your PC. The Intel Unison solution fully supports both Android and iOS."*

Okay, like... Wow! The Intel page even shows a sample of a full iMessage app running on a PC:



Send and Receive Text Messages from Your PC

Send and receive text messages on your PC, making the most of its larger keyboard and screen. Juggling fewer devices means staying focused on what makes you most productive. You can even copy SMS authentication codes directly to websites and forget about searching for your phone to complete logins on your PC.

It looks utterly amazing. But then I drilled down a bit more and tripped over this: *"The Intel Unison application is available for download on any Windows 11 PC that meets the minimum requirements, as detailed in the app store descriptions. Both laptops and desktops are supported."*

So, for anyone who has already made the move to Windows 11, this sure looks like more than I could have ever dreamed of. You could actually have a functioning messages app sitting on your desktop with your phone nearby on its charger. I already have the need to run a Windows 11 VM since the work on the DNS Benchmark has turned up some subtle but important differences in Windows 11 handling of app resizing. So I was planning to get Windows 11 setup under VirtualBox. But if I could load this into that machine I might have a Win11 VM running 24/7.

Thank you thank you, Troy!

Henrik Johnson —> Microsoft Phone Link

> *Unfortunately, not available for Windows 7, but supported in Windows 10 and later. You can check out Phone Link. Should be built into Windows and enabled by default.*
> *https://www.microsoft.com/en-us/windows/sync-across-your-devices*
> *You can read and answer texts, see any notifications and even make phone calls from your computer. It uses Bluetooth underneath to make the magic work. I just switched from Android and was really missing PushBullet, but this is a pretty solid replacement. /Henrik*

So it appears that my prayers have been answered and that the frustration I've been feeling has not been mine alone and that solutions to this have been created. I don't know how or whether this is related to Intel's Unison, but I'm going to be a very happy camper!

I'm made a bit nervous since Microsoft says that their Phone Link requires Windows 11, but Henrik clearly said Windows 10, so I'm hopeful. Microsoft might just be refusing to, in any way, promote the continued use of Windows 10.

So thank you Henri and thanks to all of our other listeners who heeded my call and pleas for a solution!

A Chat with GPT

I had an interesting interaction with the coding version of ChatGPT, 4o while working on the update to GRC's DNS Benchmark. As I've been mentioning recently, I've been using the coding platform version, which they call "4o with canvas" as a sort of super Google search on steroids. I'm often astonished by the quality of its replies. Something that I don't understand happened over the weekend while I was working on code. But, frankly, I don't understand any of this AI stuff. It's all VooDoo, and that's the problem, because I'm 100% certain that this is too important for us to not understand. And I have a plan for that... but let me first share what happened:

One of the facilities of Microsoft's Macro Assembler which I use to make my assembly code more concise and legible is the assembler's Macro facility. I have a Macro named "AppendRichEdit" which takes a string argument. So in my program code I would write, for example `AppendRichEdit "Benchmark Results"`. The way assembly macros work is that when the source code is assembled the assembler does what's called "macro expansion" which causes it to follow the simple macro script to create additional code. The point is that this is all nicely hidden behind the macro in the source code making for a more readable program. In the case of my "AppendRichEdit" macro which takes a string argument, when the code is being assembled the macro script places that string argument into the program's data section, then it writes a call to my "AppendRichEdit" function passing it a pointer to that string. I could have done the same thing by hand, but this creates a clearer communication. And one thing I've learned from my 55 years of programming is that coding is all about communication.

So the use of this macro simulates the semantics of higher-level languages, where it's possible to use a literal string as an argument in a function call. This would be like writing in the BASIC programming language `Print "Hi Mom!"` which is very convenient. But in assembly language, the string "Hi Mom!" needs to be defined elsewhere in a data section of the program and then the address of that string is provided to the Print function for it to print. This is efficient if you might have some repeated use of the string "Hi Mom!" throughout the program, since all of those repeated instances can all reference that single "Hi Mom!" data string. But the need to define the strong elsewhere in the program makes the resulting code somewhat less clear. By default, assembly language doesn't offer the high-level language convenience of in-place string declaration and use, so I use a macro to give me the same semantic flexibility.

I apologize for the long and esoteric "inside baseball" explanation, but I wanted to explain the situation surrounding what I was about to ask ChatGPT 4o. What I needed was the ability to **optionally** add another argument to macro. If that optional argument was present, it would be provided to the function call which the macro wrote. And if I didn't supply that optional argument to the macro, it would provide a default value in its place.

This is not a feature of MASM (Microsoft's Macro Assembler) that I frequently use, but I knew that MASM allows the specification of both required and optional parameters. What I wasn't sure about was having it supply default values for non-specified parameters. But since MASM has a very powerful macro facility I had the sense that it probably had the capability to supply a default value for missing optional macro parameters.

To get the details of something like this, once upon a time back in the old days – which in this case is about two months ago – I would have Googled around a bit to see whether I could quickly find an answer. But more often than not, since the sorts of things I'm looking for tend to be somewhat obscure, I'd wind up reading a couple of pages of the MASM documentation. But now we have ChatGPT which, as I've mentioned, I've been using more than more. So now the first thing I do is ask it. So I got a ChatGPT prompt and wrote:

"Can I use a MACRO in MASM where an optional MACRO parameter has a default value if it's not specified?" To which ChatGPT 4o quickly replied: *"Yes, in MASM (Microsoft Macro Assembler), you can create a macro with optional parameters that have default values if they are not specified. This is achieved by using the IFB directive to check if a parameter is blank (i.e., not provided) and then assigning it a default value within the macro. Here is an example to demonstrate how to define and use such a macro:"* And then it gave me some sample code.

Now, right off the bat, the syntax it showed for the MACRO seemed suspicious. The way it was specifying a default value looked wrong – it was a construction that various higher-level languages use. But I thought *"well... I did ask"* ... so I modified my original macro definition as ChatGPT had specified and MASM was not at all pleased with it.

Since ChatGPT seemed quite clear and authoritative in its answer, I first made sure I hadn't made some mistake, but everything looked good. I tried some fixes myself, sort of variations on the theme, but nothing I tried worked. So I replied to ChatGPT:

"That didn't work. The syntax of the first line of the MACRO definition was incorrect. Any ideas?"

And ChatGPT replied: *"You're right—MASM's macro syntax doesn't directly allow you to specify default values inline in the macro definition* [and then it quoted its own previous incorrect sample code] *(like Param2=<DefaultValue>). However, you can achieve the same effect using the IFB directive to test for blank parameters and handle them manually. Here's a corrected example:"* And although I needed to modify what it provided for my needs, that second attempt worked.

So, to recap, I asked this thing, which often conversationally refers to itself in the first person, a question. It quickly provided an authoritative sounding and reasonable answer which was nevertheless wrong. Then I pointed out that part of the answer was wrong and it said *"You're right"* and then proceeded to offer a corrected answer.

But I remained haunted by my original impression that MASM did in fact allow default values to be specified for optional and missing parameters. So I went to the source, which is where I would have wound up in the old days. And the MASM Reference revealed the truth. It is, indeed, possible to specify a default parameter for an optional and unsupplied parameter. ChatGPT was almost correct the first time. It simply forgot that a colon and an equals sign were needed to separate the parameter and its default value instead of just an equals sign.

So it gave me an answer that was initially wrong but was almost right. When I told it that there was a problem with that answer it agreed, saying *"You're right—MASM's macro syntax doesn't directly allow you to specify default values inline in the macro definition"* [which later turned out to be wrong], and gave me what was effectively a work-around for the feature that it then assumed was missing from MASM.

Finally, having found the correct answer to this, I decided to let it know. So I replied : *"Following up on this, it turns out that your first response was almost correct. The proper syntax for the first line of the MACRO definition is: "MYMACRO MACRO Param1, Param2:=<DefaultValue>" -- the thing that was missing was that colon (:) in front of the equals sign."*

And the damn thing responded: *"Ah, you're absolutely correct—thank you for pointing that out! In MASM, the syntax for specifying a default value for a parameter does indeed require the :=*

operator, not just =. Here's how it should look:" and it provided a correct example. Then it provided a little tutorial with the heading "Why := Works", writing: *"The colon in := is required to differentiate default values from other constructs in MASM. Without the colon, MASM doesn't interpret the equals sign as part of the default value definition. Thanks for catching that nuance! It makes the macro much more concise and straightforward than relying on* `IFB`*. 😊"*

I wanted to share this conversational event because I'm still startled by this thing, and because I was left staring at the screen wondering *"What have we created?"*

The fact that I really have no idea,is unnerving. And I know I'm not alone in being unnerved by this. Whatever this is, as I said several weeks ago, I believe it's the biggest and most significantly transformative event of our lifetimes. Aliens have not landed in our backyard, we have created them. And it seems clear that this is just the tip of the iceberg. I've spent my entire life working to understand the way things work. And I have proof of that. The last page of the show notes has a photo my dad took of me at age 4 at the picnic table in our backyard in Orinda, California:

I needed to understand exactly how electricity worked and why you couldn't just hook up one wire to a lightbulb. Nothing has changed since then. I want to understand this – whatever this is. So two days ago I identified and purchased two quite lengthy, technical and detailed textbooks on the subject of Large Language Models, Conversational and Generative AI.

I am blessedly, and finally, nearing the end of part one of Peter Hamilton's seemingly endless two-part Archimedes Engine novel series. Once I'm finished with that, I'm going to turn my attention to educating myself about AI – and not just for myself. I have every intention and expectation that I'm going to reprise my role as Security Now!'s *"Explainer in Chief"* to explain to this podcast audience exactly what I've learned about what we are creating. I need to know, and I'm pretty certain that among this audience I'm not alone. Stay tuned!