Security Now! #1025 - 05-13-25

Secure Conversation Records Retention

This week on Security Now!

- The state of Virginia passes an age-restriction law that has no chance.
- New Zealand also tries something similar, citing Australia's lead.
- A nasty Python package for Discord survived 3 years and 11K downloads.
- The FBI says it's a good idea to discard end-of-life consumer routers.
- What's in WhatsApp? Finding out was neither easy nor certain.
- The UK's Cyber Centre says AI promises to make things much worse.
- A bunch of great feedback from our great listeners, then:
- Is true end-to-end encryption possible when records must be retained?

A joke? Serious? Deliberately setting a high bar? Missing the point of deliberately posting a WiFi password?

Security News

Virginia's Folly

My title for this first piece of news for the week is *"Virginia's Folly"*, and I wasn't sure whether to file this under *"Things that will never happen"* or *"Good luck with that."* Because the event was so obviously fraught, there was a ton of coverage of the event. But I found a very clear and concise blog posting by the law firm Hunton Andrews Kurth who specialize in privacy and cybersecurity law. The headline title of their posting was: "Virginia Governor Signs into Law Bill Restricting Minors' Use of Social Media" under which they explain:

On May 2, 2025, Virginia Governor Glenn Youngkin signed into law a bill that amends the Virginia Consumer Data Protection Act ("VCDPA") to impose significant restrictions on minors' use of social media. The bill comes on the heels of recent children's privacy amendments to the VCDPA that took effect on January 1, 2025.

The bill amends the VCDPA to require social media platform operators to (1) use commercially reasonable methods (such as a neutral age screen) to determine whether a user is a minor under the age of 16 and (2) limit a minor's use of the social media platform to one hour per day, unless a parent consents to increase the daily limit.

The bill prohibits social media platform operators from using the information collected to determine a user's age for any other purpose. Notably, the bill also requires controllers and processors to treat a user as a minor under 16 if the user's device "communicates or signals that the user is or shall be treated as a minor," including through "a browser plug-in or privacy setting, device setting, or other mechanism." The bill also prohibits social media platforms from altering the quality or price of any social media service due to the law's time use restrictions.

The bill defines "social media platform" as a "public or semipublic Internet-based service or application" with users in Virginia that connects users in order to allow users to interact socially with each other within such service or application; and allows users to do all of the following:

- *construct a public or semipublic profile for purposes of signing into and using such service or application;*
- *populate a public list of other users with whom such user shares a social connection within such service or application; and*
- *create or post content viewable by other users, including content on message boards, in chat rooms, or through a landing page or main feed that presents the user with content generated by other users.*

The bill exempts from the definition of "social media platform" a service or application that (1) exclusively provides email or direct messaging services or (2) consists primarily of news, sports, entertainment, ecommerce, or content preselected by the provider and not generated by users, and for which any chat, comments, or interactive functionality is incidental to, directly related to, or dependent on the provision of such content.

The Virginia legislature declined to adopt recommendations by the Governor that would have strengthened the bill's children's privacy protections.

These amendments to the VCDPA take effect on January 1, 2026.

This legislation won't get off the ground before it is enjoined by multiple lawsuits arguing – with some strong rationale – that the imposition of these restrictions flies directly in the face of the freedom of speech rights enshrined by the 1st amendment to the U.S. Constitution.

It was for such a reason that I recently mentioned I was hoping that the nine justices on our Supreme Court enjoy working, since the upper echelons of the U.S. legal system are being put to much more use than they've seen in quite a while. How many times have we recently heard "this will eventually need to be decided by the Supreme Court"? We're no longer talking about whether party A defrauded party B. We're asking where — exactly — the line can be drawn when a state wishes to restrict what can reasonably be described as the free speech rights of a group of individuals. Consequently, many fundamental questions surrounding proposed laws and their precise relationship to the U.S. constitution are being asked, and will eventually be tried.

And in New Zealand

The Zealand press writes:

> *The National Party wants to ban 16-year-olds from accessing social media by forcing companies to use age verification measures. National MP Catherine Wedd, with the backing of leader Christopher Luxon, has put forward a members' bill which would follow Australia's lead on cracking down on the social media giants.*
>
> *The Prime Minister said he wanted to explore picking it up as a "broader government bill," which would mean it could become law more quickly. Right now the legislation does not have government endorsement, meaning it would be debated only if it was drawn from the ballot at random.*
>
> *Catherine Wedd said the Bill would put the onus on social media companies to verify someone is over the age of 16 before they access social media platforms, and is modelled off Australian legislation. "Currently, there are no legally enforceable age verification measures for social media platforms in New Zealand." Wedd said she'd heard from parents, teachers and principals that there wasn't enough protections in place. "My Social Media Age-Appropriate Users Bill is about protecting young people from bullying, inappropriate content and social media addiction by restricting access for under 16-year-olds." The bill would require social media platforms to take "all reasonable steps" to prevent under-16s from creating accounts. It would also introduce "penalties for non-compliance", including financial ones.*

This bit of news from New Zealand also reminds us that Australia has recently been exploring similar legislation. So I'd say that it should be very clear to anyone standing back and watching that sooner or later, and apparently sooner, we're going to be seeing age-based restrictions on access to social media which has until now been without any controls. Those days appear to be nearing an end.

The one we missed

I'm continually seeing reports of malicious supply chain attacks where this security company discovers 52 malicious Python libraries or that security company finds 46 malicious JavaScript libraries. But the troubling question is: did they find them all? And what would not finding some look like? It is for that reason that Socket Research's reporting of a very malicious and sophisticated Python Trojan remaining hidden for more than three years was troubling.

Their research posting was titled: *"Malicious PyPI Package Targets Discord Developers with Remote Access Trojan"* and their subtitle was: *"The Socket Research team investigates a*

malicious Python package disguised as a Discord error logger that executes remote commands and exfiltrates data via a covert C2 channel."

Here's the start of what the Socket Researchers wrote about their discovery:

> *On March 21, 2022, a Python package 'discordpydebug' was uploaded to the Python Package Index (PyPI) under the name "Discord py error logger." At first glance, it appeared to be a simple utility aimed at developers working on Discord bots using the Discord.py library. However, the package concealed a fully functional remote access trojan (RAT). Over time, the package reached over 11,000 downloads, placing thousands of developer systems at risk.*
>
> *The package targeted developers who build or maintain Discord bots, typically indie developers, automation engineers, or small teams who might install such tools without extensive scrutiny. Since PyPI doesn't enforce deep security audits of uploaded packages, attackers often take advantage of this by using misleading descriptions, legitimate-sounding names, or even copying code from popular projects to appear trustworthy. In this case, the goal was to lure unsuspecting developers into installing a backdoor disguised as a debugging aid.*
>
> *Discord's developer ecosystem is both massive and tightly knit. With over 200 million monthly active users, more than 25% of whom interact with third-party apps, Discord has rapidly evolved into a platform where developers not only build but also live test, share, and iterate on new ideas directly with their users. Public and private servers dedicated to development topics foster an informal, highly social culture where tips, tools, and code snippets are shared freely and often used with little scrutiny. It's within these trusted peer-to-peer spaces that threat actors can exploit social engineering tactics, positioning themselves as helpful community members and promoting tools like discordpydebug under the guise of debugging utilities.*
>
> *The fact that this package was downloaded over 11,000 times, despite having no README or documentation, highlights how quickly trust can be weaponized in these environments. Whether spread via casual recommendation, targeted DMs, or Discord server threads, such packages can gain traction before ever being formally vetted.*

https://socket.dev/blog/malicious-pypi-package-targets-discord-developers-with-RAT

The link to the rest of their extensive research is in the show notes for anyone who's interested.

Discarding old End-Of-Life routers

Once upon a time it may have been difficult to toss a perfectly good consumer router into the trash. And while it's still probably not easy or reflective, last Wednesday the U.S. Federal Bureau of Investigation posted a PSA – a Public Service Announcement – titled: "Cyber Criminal Proxy Services Exploiting End of Life Routers". Here's what the FBI wrote:

> *The Federal Bureau of Investigation (FBI) is issuing this announcement to inform individuals and businesses about proxy services taking advantage of end of life routers that are susceptible to vulnerabilities. When a hardware device is end of life, the manufacturer no longer sells the product and is not actively supporting the hardware, which also means they are no longer releasing software updates or security patches for the device. Routers dated 2010 or earlier likely no longer receive software updates issued by the manufacturer and could be compromised by cyber actors exploiting known vulnerabilities.*

End of life routers were breached by cyber actors using variants of TheMoon malware botnet. Recently, some routers at end of life, with remote administration turned on, were identified as compromised by a new variant of TheMoon malware. This malware allows cyber actors to install proxies on unsuspecting victim routers and conduct cyber crimes anonymously.

A proxy server is a system or router that provides a gateway between users and the Internet. It is an intermediary between end-users and the web pages they visit online. A proxy is a service that relays users' Internet traffic while hiding the link between users and their activity.

Cyber actors use proxy services to hide their identities and location. When actors use a proxy service to visit a website to conduct criminal activity, like stealing cryptocurrency or contracting illegal services, the website does not register their real IP address and instead registers the proxy IP.

TheMoon malware was first discovered on compromised routers in 2014 and has since gone through several campaigns. TheMoon does not require a password to infect routers; it scans for open ports and sends a command to a vulnerable script. The malware contacts the command and control (C2) server and the C2 server responds with instructions, which may include instructing the infected machine to scan for other vulnerable routers to spread the infection and expand the network.

Tips to Protect Yourself: Commonly identified signs of malware infections on routers include overheating devices, problems with connectivity, and changes to settings the administrator does not recognize. The FBI recommends individuals and companies take the following precautions:

- *If the router is at end of life, replace the device with an updated model if possible.*
- *Immediately apply any available security patches and/or firmware updates for your devices.*
- *Login online to the router settings and disable remote management/remote administration, save the change, and reboot the router.*
- *Use strong passwords that are unique and random and contain at least 16 but no more than 64 characters. Avoid reusing passwords and disable password hints.*
- *If you believe there is suspicious activity on any device, apply any necessary security and firmware updates, change your password, and reboot the router.*

So this is good but not surprising advice for anyone listening to this podcast. Still, it's not anything that most non-cybersecurity aware users would ever know to consider. So it's a good thing that these sorts of reminders and advisory "Public Service Announcements" are being made.

"Formal Analysis of Multi-Device Group Messaging in WhatsApp"
Three UK researchers, two from King's College and the third from Royal Holloway University of London, decided to tear WhatsApp apart to figure out how it solves the challenges of multi-device group messaging and to see whether they may have left any rough edges. Here's how they described their work in their resulting paper's Abstract. They wrote:

WhatsApp provides end-to-end encrypted messaging to over two billion users. However, due to a lack of public documentation and source code, the specific security guarantees it provides are unclear. Seeking to rectify this situation, we combine the limited public documentation with

information we gather through reverse-engineering its implementation to provide a formal description of the subset of WhatsApp that provides multi-device group messaging. We utilise this description to state and prove the security guarantees that this subset of WhatsApp provides. Our analysis is performed within a variant of the Device-Oriented Group Messaging model, which we extend to support device revocation. We discuss how to interpret these results, including the security WhatsApp provides as well as its limitations.

Their quite daunting 115-page paper starts off by explaining:

Group messaging in WhatsApp is based on the Signal two-party protocol and the Sender Keys multiparty extension. To date, in the academic literature, the ground truth for answering the question of how these building blocks are composed precisely is established by the WhatsApp security whitepaper or unofficial third-party protocol implementations.

I suppose I'm spoiled by Unix and Linux and Signal and other open-source security systems. The idea that these researchers were forced to reverse-engineer and divine the operational protocol of a critical encrypted communications application such as WhatsApp seems very wrong. It's true that my own SQRL client for Windows was not open source, but every single detail about the protocols it implemented was scrupulously detailed for the specific purpose of facilitating independent implementations and all I was doing was creating an implementation of SQRL's specification that was in the public domain from the first moment I disclosed its operation here on the podcast. And we know that the specification was correct because a number of other people created their own fully working SQRL implementations and everything inter-operated perfectly. That's the way these sorts of things should be done.

The details of the security protocols that, in the case of WhatsApp, billions of people depend upon, should not be considered proprietary. That's old thinking, like outlawing the export of ciphers using more than 40-bits. It's just not the crypto way. While I was following the references in this paper, I got a kick out of noticing that they were sorted alphabetically, and at the "M's" we have:

50. Marlinspike, M.: Private Group Messaging (May 2014), https://signal.org/blog/private-groups/

51. Marlinspike, M.: The Double Ratchet Algorithm (Nov 2016), https://signal.org/docs/specifications/doubleratchet/, revision 1

52. Marlinspike, M.: The X3DH Key Agreement Protocol (Nov 2016), https://signal.org/docs/specifications/x3dh/, revision 1

53. Marlinspike, M.: The Sesame Algorithm: Session Management for Asynchronous Message Encryption (Apr 2017), https://signal.org/docs/specifications/sesame/, rev 2

Our listeners who've been with us from the early days may recognize every one of those specifications and protocols because we have examined each of them over the years as we've followed Moxie's work from the start. And this is the point I wanted to make. The fact that Moxie and Signal have been sharing all of the details of their work all along demonstrates a fully mature understanding of security. By comparison, the fact that Meta's WhatsApp has taken advantage of all of the good parts of that work – for free – while refusing to disclose the very important workings of their own "proprietary" extensions of that work, is what I have a difficult time excusing.

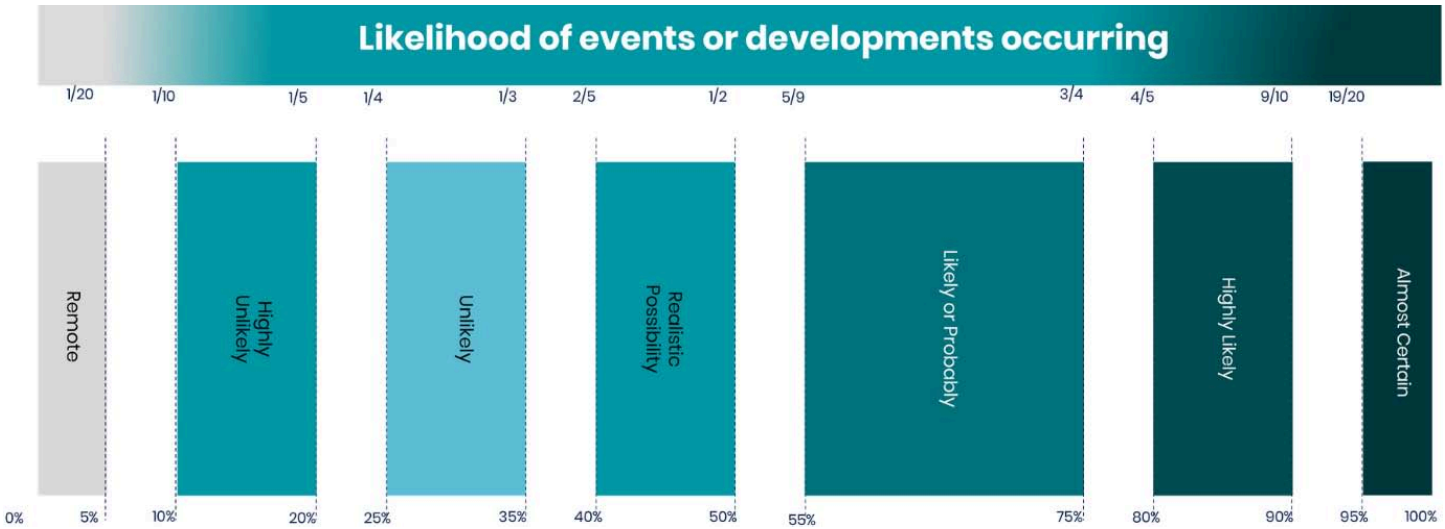
After slogging through the 115 pages of the researchers' work, they summarize their findings mostly by explaining that they were unable to find anything big that seemed to be amiss. But since they didn't have access to any source code or even to complete algorithm descriptions, *"we couldn't find anything"* is not the same as *"we looked at everything and it all looks great!"* There were a number of edge cases that they were unable to explore due to lack of information, and some others that they found that were not hugely concerning. So these intrepid academics did achieve something for their efforts, though their task could and should have been easier.

The impact of AI on cyber — threat from now to 2027

The UK's big National Cyber Security Centre (NCSC), which is roughly their equivalent of the U.S.'s CISA, just issued a report looking at the probable effect AI is expected to have upon cybersecurity over the next two years – from now to 2027. I was somewhat relieved to see that they clipped this to a relatively short-term window, since AI is advancing so rapidly that any attempt to say anything meaningful about, for example, the next ten years would be little more than a flight of fancy. No one has any remote clue what the AI in 2037 is going to look like.

They open this report by explaining who they are and what they used as the source of their various assessments. They wrote:

> *NCSC Assessment (NCSC-A) is the authoritative voice on the cyber threat to the UK. We combine source information – classified intelligence, industry knowledge, academic material and open source – to provide independent key judgements that inform policy decision making and improve UK cyber security. We work closely with government, industry and international partners for expert input into our assessments.*
>
> *NCSC-A is part of the Professional Heads of Intelligence Assessment (PHIA). PHIA leads the development of the profession through analytical tradecraft, professional standards, and building and sustaining a cross-government community.*
>
> *This report uses formal probabilistic language (see the yardstick) from NCSC-A product to inform readers about the near-term impact on the cyber threat from AI. To find out more about NCSC-A, please contact the NCSC directly.*



Then they ask themselves the question *"How likely is a 'realistic possibility'?"* And at first, this whole effort appears to go off the rails, since they then present their so-called "yardstick" which

firmly establishes the meaning of the various terms their report will use. And while I'm certainly no lover of bureaucracy, and my first instinct was to balk at this entire thing as make-work, I can see the need to define what "almost certain" means – I mean, how almost are they certain? And what about "highly likely" (how likely would that be?) or what's a "Realistic Possibility"? Those questions are all nearly answered using the handy yardstick they provide which shows essentially a spectrum of likelihoods ranging from "Remote" to "Almost Certain." So, just quickly since we're about to refer to these terms, we have, in order from least likely to most likely, the formally specified terminology: "Remote", "Highly Unlikely", "Unlikely", "Realistic Possibility", "Likely or Probably", "Highly Likely" and "Almost Certain."

Now we'll all know what they're talking about when they make the following judgements based upon all of the data they have gathered from all of their many sources. And here they are:

- *Artificial intelligence (AI) will **almost certainly** continue to make elements of cyber intrusion operations more effective and efficient, leading to an increase in frequency and intensity of cyber threats.*

- *There will **almost certainly** be a digital divide between systems keeping pace with AI-enabled threats and a large proportion that are more vulnerable, making cyber security at scale increasingly important to 2027 and beyond.*

- *Assuming a lag, or no change to cyber security mitigations, there is a **realistic possibility** of critical systems becoming more vulnerable to advanced threat actors by 2027. Keeping pace with 'frontier AI' capabilities will almost certainly be critical to cyber resilience for the decade to come.*

- *Proliferation of AI-enabled cyber tools will **highly likely** expand access to AI-enabled intrusion capability to an expanded range of state and non-state actors.*

- *The growing incorporation of AI models and systems across the UK's technology base, and particularly within critical national infrastructure (CNI), **almost certainly** presents an increased attack surface for adversaries to exploit.*

- *Insufficient cyber security will **almost certainly** increase opportunity for capable state-linked actors and cyber criminals to misuse AI to support offensive activities.*

So there is it. In other words, in the estimation of the United Kingdom's National Cyber Security Centre, it appears to be ***"highly likely"*** that the bad guys are going to be quicker to exploit the many possible nefarious benefits offered by AI than the good guys are going to be able to use AI to quickly make today's systems more secure. And if that wasn't gloomy enough, they added:

> *This report builds on NCSC Assessment of near-term impact of AI on cyber threat published in January 2024. It highlights the assessment of the most significant impacts on cyber threat from AI developments between now and 2027. It focuses on the use of **AI in cyber intrusion**. It does **not** cover wider threat enabled by AI, such as influence operations. AI and its application to cyber operations is changing fast. **Technical surprise is likely.***

"Technical Surprise" from one's adversaries is never what we want. So, overall, it appears "Almost Certain" that it would be a good idea to buckle up. We have some interesting times ahead for the industry and we'll be right here taking a look at everything every week on this podcast.

Listener Feedback

Jim Reed

Dear Steve, in SN 1024 you quoted an email from Alex regarding speed tests providers online.

"There are dozens and dozens of them…even white-label versions of the most (in)famous, the Ookla speed test. I've never really trusted the results because most of these are all about ads and the like."

As a former ISP executive, I spent a good deal of my time in speed test discussions. Until 2023, the Ookla service was operated on behalf of the Measuring Broadband America program of the FCC (https://www.fcc.gov/general/measuring-broadband-america). Using these servers would help provide "some" accountability back to the ISP. My ISP maintained a relationship to see aggregated data to help understand how we were doing.

Here's a big secret you should know. More times than not, it's not the ISP that has the network problem causing slowdowns. It's people with a TV on the patio on the edge of their Wifi coverage going, "why can't we see the game?" It's someone speed testing a gigabit connection with 100 Base-T Ethernet on their machine. It's an 8 year old iPad that has old generations of Wifi. I could go on, but you understand the issue.

I'm not exempting the ISP from the discussion, because things happen on networks. It only takes a few people to decide they need to download all 700+ episodes of The Simpsons NOW to cause in-network impacts. The ISP needs to manage for those potentials, and sometimes they just can't see things coming. Everyone starting to work from home during COVID is an example.

If I had to give advice on speed testing, here's what I would suggest. If you are concerned you are getting what you are paying for, test from a wired connection on your best device. If that doesn't meet expectations, talk to your ISP's support team.

Steve, I've been out of the ISP game for 5 years now, but I always feel like going back to the basics is a good way to start troubleshooting. Since my retirement I get the latest episode of SN every Wednesday morning and start listening on my walk. Thanks for continuing my computing education far past retirement! Best, Jim Reed P.S. - 73 to W6TWT from N4BFR

Jim's observation based upon his experience matches my own. I've almost always found that any interruption in my cable modem's connection is due to something at my end rather than something at Cox's end. I'm sure they occasionally need to rearrange things at their end. And when nothing seems to have changed at my end it would seem obvious that the other guy must be to blame. But almost invariably, when I mess with connections, even those that have been problem free for years, or reboot something that appears to be running fine, the problem will be resolved. And this is always a relief for me, since I have far more control over my end than I have over Cox's end.

Simon Griffiths

Hi Steve. First, thanks for the podcast, I've been listening since the first episode and it's really helped me in my career and my understanding of IT in general. I control a bunch of different web servers on AWS, AliCloud and others, so I was interested in your discussion of changing the SSH port, which I have done on 1 server. On AWS and AliCloud you can configure the

ports you can use to connect to the server or disable them entirely. What are your thoughts on disabling them completely, then logging in to AWS to re-enable them before you SSH in (or use AWS built in console).

I don't actively monitor SSH connection attempts, mostly because I know it would be a bit scary, but your comment on reducing traffic to the site really would be an advantage for almost all web servers. Cheers, Simon

I think Simon's approach makes a lot of sense. What we're hoping to eliminate by moving a port out of the main traffic pattern is overall exposure. So if the option exists to only enable remote access during the times it's needed, by all means take that option. A closed port is ever better than an open port that's been relocated to some backwater location.

Jon Borgen

Hey Steve, I just got done hearing you talk about Cloudflare's. speed test, and wow! It sure is good. I did want to share my favorite utility though because it's unique in a few ways. It's https://testmy.net/. What it does that I haven't seen anyone else do is actually upload and download chunks of random data, giving you a much more accurate view of your bandwidth, instead of just a theoretical throughput. You can also schedule a speed test to happen every 15 minutes or hourly if you leave the tab open, so you can get a better understanding of your bandwidth throughout the day. And it runs natively in any browser, including phones and pads. Anyway, I just wanted to share because it's my favorite. Thanks for all you and Leo do!

I went over to "testmy.net" and it does look nice. It's certainly a consumer-friendly site with some pretty graphics. And the domain "testmy.net" is convenient and easy to remember.

The only thing I'll mention is that what all of these bandwidth testing sites are doing is uploading and downloading chunks of data. They don't have any choice. That's the way they work. I can't say for certain whether that data is random. And, in truth, whether the bits are all 0's or 1's or alternating 0's and 1's or random makes absolutely no difference in the outcome. Just for the sake of argument and illustration, if some form of on-the-fly compression and decompression existed in generic Internet connections, then the content of the data that was being sent and received would matter. But typical data is not being end-to-end compressed on the Internet. But I'm happy to put "testmy.net" on everyone's radar. It looks like a very useful speed test.

Steve Main

Hi Steve, I wanted to share with you just how wrong ChatGPT can be using the PAID version of ChatGPT 4o. It is such a great tool and it's still useful but you have to be very careful as it can be so confident in its answer even if asked "Are you sure?" It took me PROVING it with a screen shot before it would admit it was 100% wrong and never once double checked its work while it told me to double check my work. I wanted to share this as I really think that this is an important point to drive home to people to be very careful using LLMs.

This is why I do not think they will be replacing anything any time soon and they are just next generation search engines. I use it for coding for PHP JavaScript and it is amazing how much it fails so many times. It's great for generating code fast but it's so buggy and it just hallucinates functions that don't exist. Again, still faster... but I've now had about 50 screwups with it over 2 months with it on a coding project.

I know that Steve's experiences, findings and experiences echoes many others, as well as mine. I thought it was notable that he referred to using AI as a next-generation Internet search engine since that's the way, several months ago, I described what my use had become, and I saw somewhere that Google's search was dropping off as many others discover that using AI as a search engine front end can help them to find what they're looking for.

Lew Wolfgang

Hi Steve, In regard to your recent explanation of SSD data retention issues, the thought occurred that it would be nice if just reading from all cells would refresh their charges. I'm reminded of reading from core memory, where the cores have to be flipped to read their polarity, if I remember correctly. (yes, I've used core ram)

I recall that we talked about the way the original magnetic core memories worked way back in the earlier years of this podcast when we were laying down the early foundations for our examination of pretty much every aspect of past and present computing.

And Lew is correct. Core memory used a technology known as "destructive read" because the process of reading a memory location inherently destroyed what was originally stored there. This core memory used tiny circular ferromagnetic rings – those were the so-called "cores". Being a closed ring, this ring could be magnetized in either a clockwise or counter-clockwise direction. A "sense" wire running through the center of the ring was able to sense when a ring switched its direction of magnetization since this switching event would induce a pulse in the so-called sense wire running through the ring and that pulse would be picked up by a sense amplifier.

So the process of reading a location of data from core memory required all of the rings representing the bits of that memory location to be written in the "0" direction. When that happened, only those rings that were originally set in the "1" direction would be reversed to the "0" direction, and that reversal event would produce a pulse on their respective bit sense wires. This allowed the computer to determine what **had** been stored in that location. But "had been stored" is the key phrase, since now that memory location was set to all "0"s.

So if, for example, the instruction the computer was executing was to simply load a memory location into one of the machine's internal solid state registers, the memory controller would need to re-write that memory location's original data back to where it came from since the act of reading it also destroyed it.

For the sake of history, I'll note one additional feature of computers of that era. The example I cited was for loading a memory location into memory. What if we wished, instead, to increment the binary value stored in a memory location. This would add one to it. The clever computer designers of the era realized that this could be accomplished at the same time as the necessary rewrite of the original data for that location. This was known as a "read-modify-write" cycle, where after reading a value from core memory, the value read would be passed through an adder to add or perhaps subtract one to or from it, and that modified value would be rewritten to the core memory instead of replacing the location's original value. A core memory's "cycle time" was significant. It was the entire limiting factor for the performance of the mainframe and mini-computers in the 60' and 70's. So the ability to fold some computational work into the required core rewrite was a significant boon.

So Lew's original point was a good one. If reading from SSDs could refresh the charges stored in their cells then a simple read pass could be used to keep cell bits firmly written. But, as we know, that's not the way today's NAND-style flash memories operate. And given that memory is,

in general, read much more frequently than it's written, if we had to choose which process would fatigue cells, it's better the way it is, where the least frequent operation is the one that takes a lot more time and also tends to produce wear.

Chris

Hey, I was just listening to this episode and the listener story about Verizon was interesting. About six months ago, I wanted to try their wireless gateway. I went to a local store and they would not sell me anything until I unfroze my credit and let them run a credit check. I'm not sure if that listener had done that or somehow the bad guy weaseled around it, but I tell anyone who will listen that they're crazy not to freeze their credit given the protections it gives you "in the real world". Enjoy the show! /Chris

I figured I'd share Chris' experience for the sake of another viewpoint. Given that the criminal purchaser presumably walked out of the store with a brand new multi-thousand dollar smartphone, and that their "creditworthiness" would seem to be a huge factor in that, it is indeed puzzling how this happened. Perhaps it was just vendor error at the "Horizon" store?

Lee MacKinnell

I decided to pull the plug on my Microsoft password as I already had passkeys set up. Microsoft makes you jump through some bizarre hoops:

1. *Install the Microsoft authenticator TOTP app on my device (it lets you use other password apps for TOTP code verification so I set it up in bitwarden, but you need THEIR app to turn off passwords... will get to that soon).*
2. *Verify you have the TOTP app installed by providing a generated code as verification*
3. *Turn on passwordless login for microsoft.com.*
4. *You are then prompted to open Microsoft's authenticator app to verify you want to disable passwords on your account.*

YAY I am now passwordless. Now ... Logging in passwordless:

1. *Enter your email address*
2. *Prompted to verify login with the Microsoft Authenticator app, asked to select the matching 2-digit code in the app. At this point, I selected 'Other ways to sign in' because I have passkeys. Shouldn't this be the default?*
3. *The 1st option is now "Use your face, fingerprint, PIN or security key.*
4. *The passkey process is then started so I can log in....*

At this point, I am unable to remove the TOTP login option from my account or set passkeys as my default login option.

I haven't tried doing any of this yet, so now I'm as confused as Lee sounds. I don't dare mess up my original Microsoft account since it's tied into my access to developer tools and downloads. So I may create a new account for testing. But I haven't done so yet. And I imagine I'll hear from other listeners who will help me figure out what's going on. Assuming that Lee is correct, it's annoying that Microsoft appears to force the use of their own authenticator app, since all time-based one-time password apps should be equal and, in fact, allowing the user to use the one they prefer to use makes more sense than forcing the temporary use of some other app.

Arthur Nilson

I am afraid I just don't get it. I have not moved to passkeys because I do not understand how they work. I have three questions:

1. *How can I use the same Microsoft account on 4 different computers and a phone with passkeys.*
2. *How can I log into my Microsoft account on a public computer with passkeys.*
3. *How can I log into my Microsoft account from my linux workstation with passkeys.*

I thought Art's question was a great one, not because I want to answer each of the use-case questions he offers, but because as he started out saying *"I have not moved to passkeys because I do not understand how they work."* One of the points I've made before is that passkeys suffer from being a mysterious hidden technology. There's a strong sense of "just trust us" magic going on. And while I'm sure that will be fine with many people, others want to first actually understand how they work. The great advantage of good old passwords is that they are a tangible real thing. And many of us prefer to work with things we can take out and look at and understand.

Bienvenido Del Rosario

Dear Steve, I'm a long time listener and Club TWIT member from the Dominican Republic.

I wanted to share my experience with an open SSH port on my home server after hearing your recount of the many authentication attempts Gaelin was receiving on Security Now Podcast episode #1023. I was using public / private keys for my own login but I had between ~70,000 to ~80,000 daily failed authentication attempts. After installing and configuring Fail2Ban the attempts went down to ~20,000 per day. Even though it was a big reduction in failed attempts I still was feeling very concerned. I started reading posts about how to mitigate even more of the failed attempts and landed on a very simple solution: Change the SSH port from the default 22 to any other number in the user port range (1024-49151) and voilà all failed attempts ceased right away. Since then, I resorted to just closing the open custom port and I use Tailscale to access my homelab and I have been very happy ever since. I hope you find this information helpful, Best regards, Bienvenido Del Rosario

Scott Schabel

Hi. The recent discussions about the use of non-standard ports for services that don't need discovery validates my use of the practice for years, so thanks. The latest Security Now discussion got me wondering... what about a non-standard port on an IPv6 address? I can't imagine it would be easy to scan and find it (again, not for security, but for "Why Not?").

I still have IPv6 turned off on my network, but am wondering if I should start working to turn it on. Thanks! -Scott

It's true that moving to IPv6 would dramatically increase the address space that bad guys would need to scan. Although only a fraction of the total 128-bit IPv6 space has been allocated to the world's ISPs, it's still true that the world's ISPs now have vast IP space for their customers.

And like Scott, I had also been routinely disabling IPv6 on my networks since being old-school I saw no need for it. But my work on the IPv6-capable DNS Benchmark code required that I have IPv6 IPs and IPv6 protocol up and running, and it's been working without any trouble.

That said, among those who have been testing the early pre-release Benchmark code there are many whose ISPs are not offering IPv6 and appear to have no plans to do so. So I've needed to have the Benchmark accommodate its users who do not have access to IPv6. They'll still have IPv4, DNS over TLS and HTTPS, but not IPv6... and after the Benchmark verifies that they need to skip IPv6 it won't bother them about it again.

Greg Bell

Hi Steve, Greg (ferrix) here under a different email due to the mailing list. Re: Windows 11 vertical taskbar, I thought you might enjoy knowing there's a bit of a story on this one.

Matching your intuition, there are some tools that give you back the "windows 10" taskbar, or something like it, on windows 11. Some work by turning back on code that MS has disabled (and may at any time delete), or by being shell replacements. Stardock and StartAllBack are examples.

I wasn't satisfied with that, because I am accustomed to my taskbar behaving precisely how I want it to. Since Windows 7 through 10, I have run the famously useful "7+ Taskbar Tweaker" utility. It improves all kinds of dumb behavior and limitations in the Windows taskbar management experience, ***without replacing the task bar or explorer****.*

Over a year ago, when I determined that MS would not support the (only sensible) vertical taskbar in Win 11, I looked to Michael, the author of the above taskbar tweaker tool. He's been improving my taskbar experience ever since Windows 7. Could he vert-ify the Windows 11 taskbar?

I learned that Michael now makes a generalized system called "windhawk" to inject various user-selected changes into the Windows UI experience. Instead of a monolith with a million checkbox features like the old tool, this is more of a framework that runs only the plugins (aka "mods") that each user wants. It doesn't replace explorer or any other process, but instead just inserts clever little hooks here and there to make windows do its bidding.

*Michael looked into the vertical taskbar issue, but despaired! Windows 11's taskbar is an almost complete rewrite; turning on verticality in the new bar simply *cannot be done*, since it was never implemented by MS in the first place. It would take at least weeks if not a couple months of development to build such a feature in the Windhawk system, if it was even possible at all — more time than Michael could afford to spend on such a hobby project. So that's where he left it.*

But my own company's small development staff (including most importantly, me) ***rely*** *on a vertical taskbar, and it would be a massive efficiency hit to lose it. So we contracted with Michael to build the feature for us! And, although we paid for the initial development, we don't own it. We agreed that the feature should be freely available to everyone, just like Windhawk itself. (Not for nothing, having a bunch of other users running this also provides feedback to make it better over time, a concept I know you are well familiar with from your own product development strategy.) In any case, I present Vertical Taskbar for Windows 11:*
https://windhawk.net/mods/taskbar-vertical

and a (somewhat dated) reddit thread about it:
https://www.reddit.com/r/Windows11/comments/1de5ww2/vertical_taskbar_for_windows_11/

I don't get anything for touting this tool, and I know the idea of running such an extension may not be universally appealing. But I thought at least you'd find it an interesting tale. I think you, Michael, and I share a certain perspective. We don't write operating systems or build the SSDs. Windows didn't plan on Windhawk being there. Active Directory didn't originally understand YubiKeys on its own. Hard drives aren't built to expect SpinRite or ReadSpeed coming along. But with a little leverage in the right place, we can make other people's systems work better than their original design parameters, and there's no feeling quite like it.

Cheers!

I perked up when I saw eMail from Greg. I know him quite well from his many years of involvement and contributions to GRC's various online forums. And this "Windhawk" system Greg brings to our attention looks truly amazing. Go to https://windhawk.net/ then at the top of the page click "Browse for Mods" or go to: https://windhawk.net/mods. By default, the page is sorted from most popular / most installed to least, with the most having nearly 143 thousand users and the least having 2. And – oh my god! – I have no idea how many mods there are overall, but the mod listing page scrolls and scrolls nearly without end. It's got to be hundreds and hundreds and very specific mod tweaks.

- Windows 11 Start Menu Styler: Customize the start menu with themes contributed by others or create your own.
- Windows 11 Taskbar Styler: Customize the taskbar with themes contributed by others or create your own.
- Taskbar height and icon size: Control the taskbar height and icon size, improve icon quality (Windows 11 only).
- Windows 11 Notification Center Styler: Customize the Notification Center with themes contributed by others or create your own.
- Taskbar Volume Control: Control the system volume by scrolling over the taskbar.
- Better file sizes in Explorer details: Optional improvements: show folder sizes, use MB/GB for large files (by default, all sizes are shown in KBs), use IEC terms (such as KiB instead of KB).
- Taskbar Clock Customization: Customize the taskbar clock. Define a custom date/time format, add a news feed, customize fonts and colors, and more.
- Slick Window Arrangement: Make window arrangement more slick and pleasant with a sliding animation and snapping.
- Taskbar Labels for Windows 11: Customize text labels and combining for running programs on the taskbar (Windows 11 only).

And, oh ... did I fail to mention that all of the source code is provided? If you click on the details for any mod, the tabs are "Details", "Source Code" and "Changelog." So as Greg said, these are going to compile down into very tiny modules.

The "Mods" page can be sorted many ways, and it has an incredibly fast and responsive incremental search. Since I was curious about the vertical taskbar Greg's company

commissioned, I entered “v”, “e”, “r”, “t" and was looking at “Vertical Taskbar for Windows 11” which describes itself as: “Finally, the missing vertical taskbar option for Windows 11! Move the taskbar to the left or right side of the screen.” That Mod currently has 6,761 users, but I have the feeling that number may soon be increasing. And, boy, as seen here in the show notes, it looks just exactly correct:

So, darnit! ... there went one of my better excuses for not moving to Windows 11. Not that I really need any. I plan to migrate GRC’s servers to Windows Server 2022, which is the server that’s synchronized with the Windows 10 desktop. Given that Windows 10 today still commands more desktops than Windows 11, I suspect that Win10 has plenty of life left in it.

But, for anyone who, like Greg, has been pining for Windows 10’s vertical taskbar under Windows 11, it is now very clearly available, and the backstory Greg has provided strongly suggests that it’s been implemented by someone who really knows his way around the Windows user-interface system. It’s clearly a solution I would choose and trust.

And, thanks Greg!!

Secure Conversation Records Retention

What is End-to-End Encryption, and what does it mean in environments with requirements for the long-term retention of records? As civil disputes have arisen in an information age, attorneys have sought to obtain records of prior events that may not have been retained. The result of this has been a growing requirement for records retention articulated by laws such as the Federal Records Act, and Freedom of Information Act, and U.S. Presidential Records Act, and the Federal Rules of Civil Procedure. And we've encountered many instances where private companies are required by law to retain their own records in the event of litigation.

The recent events surrounding Signal and TeleMessage and members of the U.S. government's use of End-to-End Encryption, with TeleMessage's mission to archive conversations, raises many questions about the intersection of secure communications and the need for long term records retention.

But I'm getting ahead of myself. I settled upon this topic for discussion during this week's podcast only after catching up with the news of an event that was the topic of last week's podcast. Because it was not my original intention to give this whole TeleMessage SignalGate story much more air, I originally had this as a news item near the top of the podcast. But we have learned some more in the past week, and the details legitimately led me to pose the question that became today's topic. So first, let's all catch up on what's transpired over the past week:

Under the summary line *"Three US departments ban TeleMessage"*, the Risky Business security newsletter wrote: According to Bloomberg, *"Three US government departments have told employees to stop using the TeleMessage service. The service allows companies to log and archive conversations taking place in secure messengers such as Signal, WhatsApp, and others. Two separate hackers breached TeleMessage's backend last week after it was revealed that White House officials use the service."*

WIRED had the headline *"Customs and Border Protection Confirms Its Use of Hacked Signal Clone TeleMessage"* with the subhead *"CBP says it has "disabled"* its use of TeleMessage following reports that the app, which has not cleared the US government's risk assessment program, was hacked." Rather than share the entire article which is padded with a bunch of stuff that we already know, I'll extract things we didn't already have on the record:

The United States Customs and Border Protection agency confirmed on Wednesday that it uses at least one communication app made by the service TeleMessage, which creates clones of popular apps like Signal and WhatsApp with the addition of an archiving mechanism for compliance with records-retention rules. CBP spokesperson Rhonda Lawson told WIRED: "Following the detection of a cyber incident, CBP immediately disabled TeleMessage as a precautionary measure. The investigation into the scope of the breach is ongoing."

So we have confirmation of the belief which arose from the hacked data, that was anonymously shared with 404 Media, that the CBP was, indeed, using the TeleMessage app. Also:

In the days since the photo was published, TeleMessage has reportedly suffered a series of breaches that illustrate concerning security flaws. Analysis of the app's Android source code also appears to indicate fundamental flaws in the service's security scheme. As these findings emerged, TeleMessage—an Israeli company that completed an acquisition last year by the US-based company Smarsh—imposed a service pause on its products pending investigation. A

Smarsh spokesperson told WIRED in a statement: "TeleMessage is investigating a potential security incident. Upon detection, we acted quickly to contain it and engaged an external cybersecurity firm to support our investigation. Out of an abundance of caution, all TeleMessage services have been temporarily suspended. All other Smarsh products and services remain fully operational."

And: *"There is still no complete public accounting of US government officials and agencies that have used the software."*

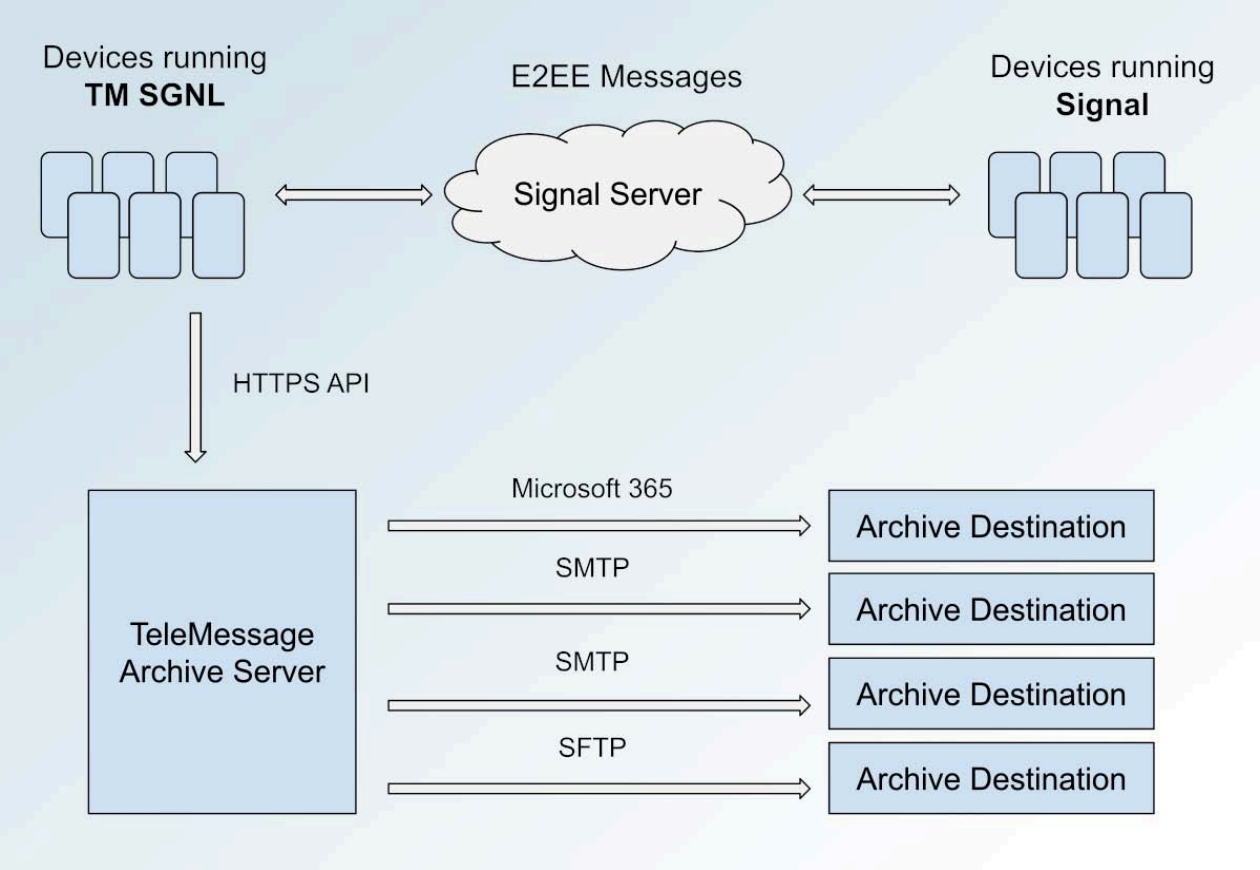
Jumping to the bottom of reporting by NBS News, we have:

> *But archives of sensitive information inherently make targets for hackers. On Sunday evening, a hacker credibly claimed to NBC News to have broken into a centralized TeleMessage server and downloaded a large cache of files. As evidence, the hacker provided a screenshot of TeleMessage's contact list of employees at the cryptocurrency broker Coinbase, which uses TeleMessage. A Coinbase spokesperson confirmed to NBC News that the screengrab was authentic, but stressed that Coinbase had not been hacked and that none of its customers' data had been affected.*
>
> *The Coinbase spokesperson said: "At this time, there is no evidence any sensitive Coinbase customer information was accessed or that any customer accounts are at risk, since Coinbase does not use this tool to share passwords, seed phrases, or other data needed to access accounts."*
>
> *The hacker told NBC News they have not fully sifted through the hacked files yet, and it is unclear if they include sensitive conversations from the U.S. government. Several government agencies, including the Department of Homeland Security, the Department of Health and Human Services, the Treasury Department and the U.S. International Development Finance Corp., appear to have active contracts with TeleMessage or other companies to use TeleMessage's services, according to government records reviewed by NBC News. Separately, a different hacker told the tech news publication 404 Media that they had also hacked TeleMessage and provided significant evidence. NBC News has not interacted with that hacker.*

Last week, security researcher Micah Lee blogged under the headline *"Despite misleading marketing, Israeli company TeleMessage, used by Trump officials, can access plaintext chat logs."*

Micah wrote: *"In this post I give a high level overview of how the TeleMessage fake Signal app, called TM SGNL, works and why it's so insecure. Then I give a thorough analysis of the source code for TM SGNL's Android app, and what led me to conclude that TeleMessage can access plaintext chat logs. Finally, I back up my analysis with as-of-yet unpublished details about the hack of TeleMessage."*

Micah created a clear and simple diagram that depicts the flow of information for anyone using TeleMessage's modified Signal app. The modified app does what it claims to do. But since the archived messages are themselves stored and forwarded in the clear, that does serve to verify that TeleMessage themselves would, indeed, be privy to the content of the message flow of any of their customers. That's pretty much sure to be a permanent deal breaker for many of the more security sensitive users of this system.



But here's something I haven't read or seen anywhere: This would be totally obvious to anyone with any operational cybersecurity understanding at all. So let me repeat this: The TeleMessage system is depositing the full plaintext transcripts of all conversations conducted with the Signal protocol whenever using this app is being used. My point is, the entire design of this system is so transparently **in**secure that any claim to security would be utterly laughable. Micah starts out saying *"Despite misleading marketing..."* Misleading marketing? It's sending everything you sent and received to Microsoft Outlook or any SMTP email in the clear. There's nothing about this that's secure for the content of these messages. Nothing.

One of our listeners, responding to my reporting of this last week, apparently feeling the need to defend the current administration, claimed that TeleMessage's use had been approved by the Biden administration. At the time I knew nothing either way, now I do. I presume this listener picked up this politically partisan fiction somewhere, and wanted to believe it. But it was never true. And these sorts of statements, expressed as fact, tend to spread and they're not helpful. The Biden administration may well have done stupid things, but this one was not among them. TeleMessage's federal use has **never** been approved for use by **anyone** within the federal government. Part of the reporting about this in the past week noted that while the TeleMessage company is **itself** a federal contractor, the consumer apps it offers are **not** approved for use under the US government's Federal Risk and Authorization Management Program, known as FedRAMP. That's the approval that would be required.

And that's actually a relief, because TeleMessage's apps, which send conversation plaintext to any email servers specified by their users, could never have ***_possibly_*** been considered safe or secure to use. This would have to be so blatantly obvious to anyone with even the slightest cybersecurity training that I would be quite worried if these apps **could** have ever been approved for use under the FedRAMP program. So the use of these communications applications by federal government officials was entirely illicit, so that one can be put to rest.

Micah's blog posting digs deeply into the entrails of TeleMessage's Android app, which is open source. I scanned through Micah's details posting and reverse engineering and did not find anything that merited further deeper discussion. But I've included its link in the show notes for anyone who might be looking for more:

https://micahflee.com/despite-misleading-marketing-israeli-company-telemessage-used-by-trump-officials-can-access-plaintext-chat-logs/

There's really nothing anyone with any training needs to know once it's understood that TeleMessage is **emailing** its users' conversation logs to external email servers. From that point on, it's simply "game over" , and it must have made those within the NSA and CIA and other security-aware agencies who know how much our adversaries would love to get their hands on these conversations.

So this brings us to the first part of the meat of today's topic. When we use the term "Secure Conversation" and where TeleMessage quite clearly failed, what, really, is End-to-End Encryption?

And by that I mean, what do we mean when we say that something is end-to-end encrypted? How is that some sort of special type of encryption? Or a special type of encrypted system? And the bigger point here is: How is that term increasingly being used, misused and abused?

The difficulty is that the term's common use appears to be diverging from its technical meaning, and it has become a buzzword term which marketing types are tossing around because it feels good and fancy to say it.

Let's first turn to the Internet's encyclopedia to see what those who have spent time working to craft a clear and concise definition of the term have come up with. Of End-to-End Encryption, Wikipedia writes:

> *End-to-end encryption (E2EE) is a method of implementing a secure communication system where only communicating users can participate. No one else, including the system provider, telecom providers, Internet providers or malicious actors, can access the cryptographic keys needed to read or send messages.*
>
> *End-to-end encryption prevents data from being read or secretly modified, except by the true sender and intended recipients. Frequently, the messages are relayed from the sender to the recipients by a service provider. However, messages are encrypted by the sender and no third party, including the service provider, has the means to decrypt them. The recipients retrieve the encrypted messages and decrypt them independently. Since third parties cannot decrypt the data being communicated or stored, services that provide end-to-end encryption are better at protecting user data when they are affected by data breaches. Such services are also unable to share user data with government authorities, domestic or international.*

I'd say that's a beautifully crafted definition of what is currently meant by the proper use of the term. By way of comparison to non-E2EE systems, the Wikipedia article then writes:

> *In many non-E2EE messaging systems, including email and many chat networks, messages pass through intermediaries and are stored by a third party service provider, from which they are retrieved by the recipient. Even if the messages are encrypted, they are only encrypted 'in transit', and are thus accessible by the service provider. Server-side disk encryption is also*

distinct from E2EE because it does not prevent the service provider from viewing the information, as they have the encryption keys and can simply decrypt it.

The lack of end-to-end encryption can allow service providers to easily provide search and other features, or to scan for illegal and unacceptable content. However, it also means that content can be read by anyone who has access to the data stored by the service provider, by design or via a backdoor. This can be a concern in many cases where privacy is important, such as in governmental and military communications, financial transactions, and when sensitive information such as health and biometric data are sent. If this content were shared without E2EE, a malicious actor or adversarial government could obtain it through unauthorized access or subpoenas targeted at the service provider.

E2EE alone does not guarantee privacy or security. For example, data may be held unencrypted on the user's own device, or be accessible via their own app, if their login is compromised.

One of this podcast's early terms and acronyms was TNO which stood for "Trust No One." We used it when talking about storing our data in the cloud well before anything was being called "The Cloud". The simple idea was that client-side encryption and decryption would be employed on the user's PC so that the only thing we were asking remote services to store was big blobs of data that were indistinguishable from completely pseudo-random data – which is exactly what good encryption produces. Later, the term was used interchangeably with PIE – P.I.E. – which stood for Pre-Internet Encryption.

So what emerges here is a very clear understanding of what the proper use of the term would be. In practical terms it means that the providers of the service and any intermediaries that they may engage, never, under any circumstances, have any access to the unencrypted content of the messages or data or whatever it is that they are conveying or storing on behalf of their users.

For this assertion to hold, any of the user-data that a provider of end-to-end encrypted services ever comes into contact with must be encrypted and the provider must never, at any time, have access to the cryptographic keys that would be capable of decrypting the data. By that definition and the set of requirements that flow from it, it's clear that the message archiving services TeleMessage was offering could never – literally by definition – have been considered to be end-to-end encrypted. That term could never be accurately applied.

By comparison, we've previously explored at length the protocols that the likes of Apple and Signal have gone to in order to truly meet and live up to the definition of "end to end encrypted" communications and storage. This entire battle that Apple is presently engaged in with the UK over the provisioning of their Advanced Data Protection amounts to exactly this. The switch on the user's UI on the phone is labeled "Enable Advanced Data Protection" but it could just as accurately be labeled "Enable End-to-End Encryption."

It might next be instructive to ask: Is there any hope for TeleMessage? Is there any way for a potentially useful service such as this to be saved? Or phrased another way: "Could message archiving of a true end-to-end encrypted system such as Signal, WhatsApp, or Telegram remain end-to-end encrypted?" And the answer to that is, yes. There's definitely a way to do this securely, but TeleMessage just never bothered to. They opted for the user convenience of forwarding the user's plaintext conversation logs to email and in doing so they probably put themselves out of business forever. I doubt anyone would ever consider trusting TeleMessage again. So when someone does produce a true end-to-end encrypted messaging system with

end-to-end encrypted long-term archiving, it won't be those guys.
So, how could what TeleMessage wanted to do be accomplished securely? The most obvious solution is to simply modify a cloned Signal app for long term local archival storage. So it would offer local storage, search and retrieval on the client-side device itself. After all, the plaintext data is already there, it's been shown to its user right there on the screen. So the app simply needs to hold onto it, right? Problem solved. The behavior everyone is objecting to is having these modified TeleMessage Signal clients emailing their conversation logs to its user's insecure email. So, remove that feature from the app and replace it with long-term archival storage of everything that app sends and receives.

And to be a responsible archiving Signal clone, any such clone which has been configured for long-term message archiving, should periodically inject a notice to all other members of any archived conversation that its user, 'member X' is permanently archiving their conversation. That might be expected to alter the behavior of the people in the group, causing it to be a bit less boisterous and flippant, but that would probably be a good thing, too. And if nothing else, these periodically injected archiving reminders would cause its recipients to ask the archivist why they were keeping the conversation.

But now we have a new problem. The problem is that high level government or other individuals who are using an archiving secure messaging clone, whose entire messaging history exists inside that device, wind up carrying around something in their pocket that might well be worth a great deal to the right parties. No one in their right mind would want to have it known that they're walking around town, riding in cabs and dining in restaurants with months or years of top secret records retained in their smartphones. So this is not a practical solution either.

And even if walking around with this conversation archive was not in itself problematic, the presumption is that aside from having the convenience of searching for and retrieving past conversation details, there is also some significant need for government records to be retained or other other executive compliance requirements to be met. If a device were to be lost or stolen, and even if it could not be unlocked and used, the loss of those records could be a huge problem.

This brings us to the inescapable conclusion that the app's local conversation memory should be kept short and non-archival, and that some secure means of removing the data from the device, while retaining it permanently, should be found.

The solution I've come up with that completely solves all of these various problems, providing truly secure records retention for users of Signal — and using only the official unadulterated Signal app — to deliver full true end-to-end encrypted conversation security, would be for these individuals to add a secure government Signal-Bot to their conversations.

This "Signal-Bot" would be running deep inside a secure NSA facility. It would auto-accept all conversation invitations from known government staffers, and it would passively receive and permanently archive all dialog from everyone else who was participating in the conversations. It would function as a silent observer and would appear as a participant named something like "Federal Records Retention" in the list of conversation members so that everyone participating would be informed and reminded that their conversation was subject to legally required retention while at the same time being assured that their conversation was being retained and that they were, thereby, abiding by their oaths of office and the law while participating in these conversations.

Being a "Signal-Bot", all of the standard end-to-end encryption guarantees offered by the Signal protocol would apply. The only possible points of vulnerability would be at the individual device

endpoints, but that’s inherent in the use of any messaging application.
Since Signal is open source and desktop clients already exist, the creation of various secure message archiving bots would be a simple matter for the likes of the NSA, CIA, CISA, or whomever. Or, better yet, perhaps a successor company to TeleMessage will arise from the ashes of TeleMessage’s clearly useful and important concept to create a commercial implementation of this solution for those who need Signal message archiving.

I would imagine that many enterprises would welcome the ability to automatically have their executives and other important individuals’ secure Signal conversations securely archived on-premises without creating the sort of serious security weaknesses we’ve all just witnessed from TeleMessage.

In retrospect, this seems like such an obvious solution for the secure archiving of secure messaging that I’m surprised such a service doesn’t already exist. Let’s hope someone creates such a system soon. It should not take too much time or effort... and it should definitely remain open source since it’s all about first understanding and then trusting the implementation.