Security Now! #971 - 04-23-24

Chat (out of) Control

This week on Security Now!

What would you call Stuxnet on steroids? What's the latest on the Voyager 1 drama? What new features are coming to Android and Thunderbird? What's China done now? Why did Gentoo Linux say 'no' to AI? And after sharing and discussing a bunch of feedback from our terrific listeners and a SpinRite update, we're going to examine the latest update to the European Union's worrisome "Chat Control" legislation which is reportedly just over a month away from becoming law. Is the EU about to force the end of end-to-end encryption in order to enable and require the scanning of all encrypted communications? It appears ready to do just that.

"Can you go through all the old pitch decks and replace the word 'crypto' with 'A.I.'?"

Security News

What do you call "Stuxnet on steroids"??

As we know, Security Now! is primarily an audio podcast. But even those watching, though it remains unclear why anyone would, won't have the advantage of looking at my show notes.
If anyone were to be reading the notes they would see that the **spelling** of the name of this new attack is far more, shall we say, acceptable in polite company than the attack's verbal pronunciation. But this is an audio podcast, and the story of this attack that I very much want to share refers to the attack by name. And that name, which rhymes with "stuxnet" is spelled "F U X N E T" and there's really no other way to pronounce it than to just spit it out and say "fuxnet." I apologize to anyone this might offend. It's really not the 'F' bomb, even though it's audibly identical.

So, with that preamble and apology, let's look at the very interesting attack that was reported last week by Security Week. Their headline, which also did not shy away from using the attack's name was: *"Destructive ICS Malware 'Fuxnet' Used by Ukraine Against Russian Infrastructure."* Here's what we learn, they wrote:

> *In recent months, a hacker group named Blackjack, which is believed to be affiliated with Ukraine's security services, has claimed to have launched attacks against several key Russian organizations. The hackers targeted ISPs, utilities, data centers and Russia's military, and allegedly caused significant damage and exfiltrated sensitive information.*
>
> *Last week, Blackjack disclosed the details of an alleged attack aimed at Moscollector, a Moscow-based company responsible for underground infrastructure, including water, sewage and communication systems.*
>
> *<quote> "Russia's industrial sensor and monitoring infrastructure has been disabled," the hackers claimed. <quote> "It includes Russia's Network Operation Center (NOC) [that] monitors and controls gas, water, fire alarms and many others, including a vast network of remote sensors and IoT controllers." <unquote>*
>
> *The hackers claimed to have wiped database, email, internal monitoring and data storage servers.*
>
> *In addition, they claimed to have disabled [some] 87,000 sensors, including ones associated with airports, subway systems and gas pipelines. To achieve this, they claimed to have used Fuxnet, a malware they described as "Stuxnet on steroids", which enabled them to physically destroy sensor equipment.*
>
> *The hackers wrote: "Fuxnet has now started to flood the RS485/MBus and is sending 'random' commands to 87,000 embedded control and sensory systems (while carefully excluding hospitals, airports, and other civilian targets),"*
>
> *The hackers' claims are difficult to verify, but the industrial and enterprise IoT cybersecurity firm **Claroty** was able to conduct an analysis of the Fuxnet malware based on information and code made available by Blackjack.*
>
> *Claroty pointed out that the actual sensors deployed by Moscollector, which are used to collect physical data such as temperature, were likely not themselves damaged by Fuxnet. Instead,*

> *the malware likely targeted roughly 500 sensor* ***gateways****, which communicate with the sensors over a serial bus such as the RS485 or Meter-Bus that was mentioned by Blackjack. These gateways are also connected to the Internet to be able to transmit data to the company's global monitoring system.*
>
> *Claroty notes: "If the gateways were indeed damaged, the repairs could be extensive given that these devices are spread out geographically across Moscow and its suburbs, and must be either replaced or their firmware must be individually reflashed."*
>
> *Claroty's analysis of Fuxnet showed that the malware was likely deployed remotely. Then, once on a device, it would start deleting important files and directories, shutting down remote access services to prevent remote restoration, and deleting routing table information to prevent communication with other devices. Fuxnet would then delete the file system and rewrite the device's flash memory. Once it has corrupted the file system and blocked access to the device, the malware attempts to* ***physically destroy*** *the NAND memory chip and then rewrites the UBI volume to prevent rebooting.*
>
> *In addition, the malware attempts to disrupt the sensors connected to the gateway by flooding their serial communications channels with random data in an effort to overload the serial bus and the sensors. Essentially performing an internal DoS attack on all of the devices the gateway is connected to.*
>
> *Claroty explained: "During the malware operation, it will repeatedly write arbitrary data over the Meter-Bus channel. This will prevent the sensors and the sensor gateway from sending and receiving data, rendering the sensor data acquisition useless. Therefore, despite the attackers' claim of physically destroying 87,000 devices, it seems that they actually managed to infect the sensor gateways and were causing widespread disruption by flooding the Meter- Bus channel connecting the sensors to the gateway, similar to network fuzzing the different connected sensor equipment. As a result, it appears only the sensor gateways were bricked, and not the end-sensors."*

Okay. I particularly appreciated the part about attempting to physically destroy the gateway's NAND memory chip. As we know, NAND memory is fatigued by writing because writing and erasing is performed by forcing electrons to tunnel through insulation thus weakening its dielectric properties over time. So the attacking malware is likely writing and erasing and writing and erasing the NAND memory over and over. And since such memory is likely embedded into the controller and is not field replaceable, that would necessitate replacing the gateway device. And perhaps all 500 of them spread across Moscow and its suburbs.

And even if the NAND memory was not rendered unusable, the level of destruction appears to be quite severe. Wiping stored data and directories and killing the system's boot volume means that those devices probably cannot be remotely repaired. Overall, I'd have to say that this extremely destructive malware was pretty well named.

We live in an extremely, and increasingly, cyber-dependent world. Everyone listening to this podcast knows how rickety the world's cybersecurity truly is. So I really do shudder at the idea of any sort of all-out confrontation between super powers. I don't want to see that.

Voyager 1 update

We have another update on Voyager 1. Apparently, if Voyager is not going to give up on us we're not going to give up on it. But remember that no matter what, Voyager is deriving all of its diminishing operating power from the heat being generated by the decay of radioisotopes, and through the years and now decades, they are continuing to put out less and less heat.

But what equally amazes me is the intrepid group of well-past-their-retirement engineers who are now endeavoring to patch the code of this ancient machine that's twenty-two and a half light hours away. It boggles the mind.

Just yesterday, April 22nd, JPL, NASA's Jet Propulsion Laboratory posted the news under the headline: "NASA's Voyager 1 Resumes Sending Engineering Updates to Earth". They wrote:

> *After some inventive sleuthing, the mission team can — for the first time in five months — check the health and status of the most distant human-made object in existence.*
>
> *For the first time since November, NASA's Voyager 1 spacecraft is returning usable data about the health and status of its onboard engineering systems. The next step is to enable the spacecraft to begin returning science data again. The probe and its twin, Voyager 2, are the only spacecraft to ever fly in interstellar space (the space between stars).*
>
> *Voyager 1 stopped sending readable science and engineering data back to Earth on Nov. 14, 2023, even though mission controllers could tell the spacecraft was still receiving their commands and otherwise operating normally. In March, the Voyager engineering team at NASA's Jet Propulsion Laboratory in Southern California confirmed that the issue was tied to one of the spacecraft's three onboard computers, called the flight data subsystem (FDS). The FDS is responsible for packaging the science and engineering data before it's sent to Earth.*
>
> *The team discovered that a single chip responsible for storing a portion of the FDS memory — including some of the FDS computer's software code — isn't working. The loss of that code rendered the science and engineering data unusable. Unable to repair the chip, the team decided to place the affected code elsewhere in the FDS memory. But no single location is large enough to hold the section of code in its entirety.*
>
> *So they devised a plan to divide the affected code into sections and store those sections in different places in the FDS. To make this plan work, they also needed to adjust those code sections to ensure, for example, that they all still function as a whole. Any references to the location of that code in other parts of the FDS memory needed to be updated as well.*
>
> *The team started by singling out the code responsible for packaging the spacecraft's engineering data. They sent it to its new location in the FDS memory on April 18. A radio signal takes about 22 ½ hours to reach Voyager 1, which is over 15 billion miles (24 billion kilometers) from Earth, and another 22 ½ hours for a signal to come back to Earth. When the mission flight team heard back from the spacecraft on April 20, they saw that the modification worked: For the first time in five months, they have been able to check the health and status of the spacecraft.*
>
> *During the coming weeks, the team will relocate and adjust the other affected portions of the FDS software. These include the portions that will start returning science data.*

Android 15 to quarantine apps

There’s not a lot of clear information about this, yet, but Google is working on a new feature for Android that will place under quarantine any applications that may sneak past its Play Store screening only to then exhibit signs of behavior that appears malicious. The apps will reportedly have all of their activity stopped, all of their windows hidden, and notifications from quarantined apps will not be shown. These apps will also be unable to offer any services to other apps. Google began working on the feature during Android 14’s development last year and the feature is expected to finally appear in the forthcoming Android 15, though that’s not confirmed.

Thunderbird & Microsoft Exchange

This summer, Thunderbird will be acquiring support for Microsoft Exchange eMail. It will only be eMail at first. Calendar and Contacts will follow at some later date. I’m a Thunderbird user, having been forced to relinquish my beloved Eudora eMail client once I began receiving eMail containing extended non-ASCII character sets that Eudora was unable to manage. However I have zero interest in Exchange. GRC runs a simple and straightforward instance of hMailServer which handles traditional POP, IMAP and SMTP and does it effortlessly with ample features.

In any event, to support this rather massive coding effort, Mozilla chose RUST as their implementation language and they did so for all the usual reasons. They cited:

- Memory safety. Thunderbird takes input from anyone who sends an email, so we need to be diligent about keeping security bugs out.
- Performance. Rust runs as native code with all of the associated performance benefits.
- Modularity and Ecosystem. The built-in modularity of Rust gives us access to a large ecosystem where there are already a lot of people doing things related to email which we can benefit from.

China bans Western encrypted messaging apps

The Chinese government ordered Apple to remove four foreign apps from the Chinese version of the App Store. So Meta's new social network Threads is gone as are Signal, Telegram and WhatsApp. China stated that they have national security concerns about those four and as we have seen, and as I fear we’ll be seeing shortly within the EU, what countries request countries receive. Technology is ultimately unable to stand up to legislation.

Gentoo says “no” to AI

I thought this one was interesting. I’ll just jump right in by sharing the posting to the Gentoo mailserve. This was posted by a long standing (since 2010) Gentoo developer and contributor:

> *Given the recent spread of the “AI” bubble, I think we really need to look into formally addressing the related concerns. In my opinion, at this point the only reasonable course of action would be to safely ban “AI”-backed contribution entirely. In other words, explicitly forbid people from using ChatGPT, Bard, GitHub Copilot, and so on, to create ebuilds, code, documentation, messages, bug reports and so on for use in Gentoo.*

> *Just to be clear, I'm talking about our "original" content. We can't do much about upstream projects using it. Rationale:*
>
> *1. Copyright concerns. At this point, the copyright situation around generated content is still unclear. What's pretty clear is that pretty much all LLMs are trained on huge corpora of copyrighted material, and the fancy "AI" companies don't care about copyright violations. What this means is that there's good risk that these tools would yield stuff we can't legally use.*
>
> *2. Quality concerns. LLMs are really great at generating plausible looking B.S. I suppose they can provide good assistance if you are careful enough, but we can't really rely on all our contributors being aware of the risks.*
>
> *3. Ethical concerns. As pointed out above, the "AI" corporations care about neither copyright nor people. The AI bubble is causing huge energy waste. It is giving a great excuse for layoffs and increasing exploitation of IT workers. It is driving the further "enshittification" of the Internet, it is empowering all kinds of spam and scam.*
>
> *Gentoo has always stood out as something different, something that worked for people for whom mainstream distros were lacking. I think adding "made by real people" to the list of our advantages would be a good thing — but we need to have policies in place, to make sure that AI-generated crap doesn't flow in.*

I had to clean up the language in that posting for the podcast since its author, who has, as I mentioned, been a very active contributor to the Gentoo Linux project since 2010, pretty clearly doesn't think much of AI generated code. And separately, there have been some signs of GitHub repository properties and descriptions appearing to be AI generated. They're not high quality and I suppose we should not be surprised that people who are probably incapable of coding from scratch for themselves would take to using Large Language Model systems to allow them to feel that they're contributing. But are they really?

We all saw a terrific example of the value of generative AI when one of our listeners used an earlier ChatGPT to write most of that LastPass vault processor. He said that without ChatGPT he wouldn't have bothered at all. But it gave him enough of a head start that he was able to produce it without nearly as much additional work. That definitely means something.

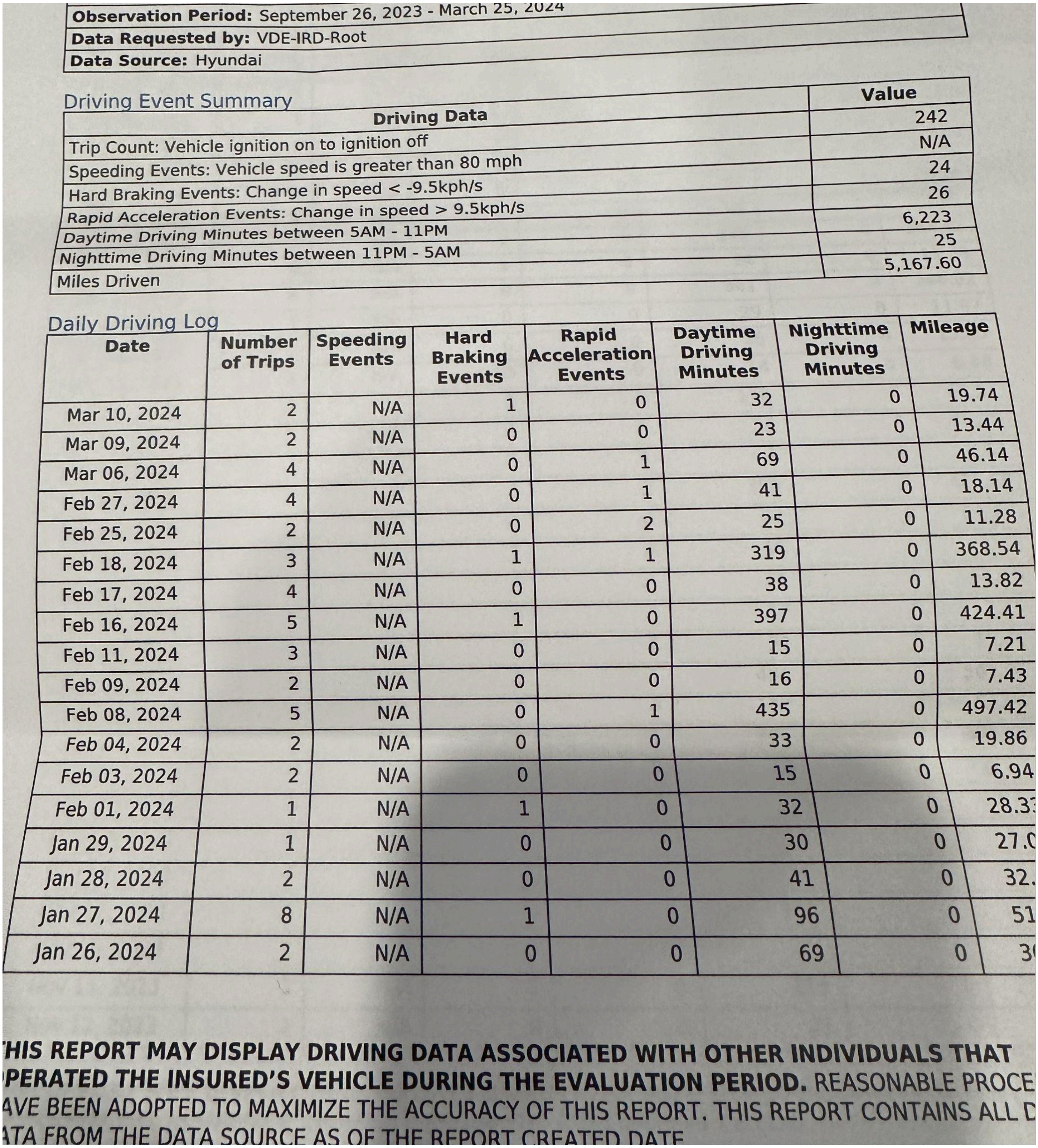
But he never claimed that the work was all his. He was always clear about how the app was created. There is something vaguely creepy about the practice of someone palming off an AI's output as their own. It would be a bit like playing online chess with someone and having them using a computer on their end while pretending that their excellent moves are their own. At the same time I think it's also clear that thanks to all of this AI stuff we're going to see a real transformation in the way people interact with computers and I'm sure we're still in the very early days.

Closing The Loop

From: “Andrew” via eMail to GRC:

I received the following note from a self-described user of ShieldsUP!, SpinRite and avid listener of Security Now! Who is also an Information Security Practitioner/Computer Geek. So my point is, not anyone’s typical naive consumer. Andrew wrote:

> *Hi Steve,*
>
> *I apologize for sending to this email as I couldn’t find a different email for contact information. Anyway, long time follower of Shields UP and SpinRite and an avid listener of Security Now – my full time gig is an InfoSec Security Practitioner/Computer Geek.*
>
> *We have a couple of Hyundais in the family, and I purchased one last fall. I use the Hyundai Bluelink app on my phone, as I can make sure I locked my doors, and get maintenance reminders. I made a point to NOT opt-in for the “driver discount” and, as a privacy cautious person, I decline sharing data wherever possible. But after the story in the NYT regarding car makers sharing data, I contacted Verisk and Lexis Nexis to see what they had on me.*
>
> *Lexis Nexis had nothing other than the vehicles I have owned in the past, but Verisk has a lot. I have attached a page of the report. It includes driving dates, minutes (day and night) , acceleration events, and braking events. The only thing missing is the actual speeds I was going or if I was ever speeding.*
>
> *What bothers me most about this, is that I have no way to challenge the accuracy. For events that are not illegal, I can still be penalized. Braking hard and accelerating fast should not be safety concerns without context. And today’s smarter cars are still imperfect. My adaptive cruise control (radar) will still hard brake at times it shouldn’t, and I will get penalized by the data. My car is also a turbo, and if I accelerate for fun or safety, that too can be a penalty. And if I happen to drive in Texas there are highways with an 85 MPH speed limit, so I would be down rated for that legal behavior.*
>
> *My family tried the “safe driving” BT dongles from another insurer years ago, but the app had too many false positives for driving over speed (posted speed limit didn’t agree with the app) , and hard braking, and accelerating, that we decided it wasn’t worth our time or the privacy concerns . My wife and I are close to Leo’s age, and she drives like a grandmother, but her scores were no better than mine.*
>
> *I have attached a picture of the document I got from Verisk (name and VIN Number removed) to give you an idea of what is reported without my consent from my car. I have contacted Hyundai, and told them I do not and did not consent to them sharing my data with Verisk. After a few back and forths I go this reply on April 12th:*
>
> *“Thank you for contacting Hyundai Customer Care about your security concerns.*
> *As a confirmation, we have been notified today that the driver's score feature and all data collecting software has permanently disabled. We do care. As always, if you ever need additional assistance, you can do so either by email or phone. Case number….”*
>
> *I will request another report from Verisk in the future to validate this. Keep up the good work, I thought you would like to see the data, and hear from someone who is 100% certain they never opted in. All the best, Andrew.*



Observation Period: September 26, 2023 - March 25, 2024	
Data Requested by: VDE-IRD-Root	
Data Source: Hyundai	

Driving Event Summary

Driving Data	Value
Trip Count: Vehicle ignition on to ignition off	242
Speeding Events: Vehicle speed is greater than 80 mph	N/A
Hard Braking Events: Change in speed < -9.5kph/s	24
Rapid Acceleration Events: Change in speed > 9.5kph/s	26
Daytime Driving Minutes between 5AM - 11PM	6,223
Nighttime Driving Minutes between 11PM - 5AM	25
Miles Driven	5,167.60

Daily Driving Log

Date	Number of Trips	Speeding Events	Hard Braking Events	Rapid Acceleration Events	Daytime Driving Minutes	Nighttime Driving Minutes	Mileage
Mar 10, 2024	2	N/A	1	0	32	0	19.74
Mar 09, 2024	2	N/A	0	0	23	0	13.44
Mar 06, 2024	4	N/A	0	1	69	0	46.14
Feb 27, 2024	4	N/A	0	1	41	0	18.14
Feb 25, 2024	2	N/A	0	2	25	0	11.28
Feb 18, 2024	3	N/A	1	1	319	0	368.54
Feb 17, 2024	4	N/A	0	0	38	0	13.82
Feb 16, 2024	5	N/A	1	0	397	0	424.41
Feb 11, 2024	3	N/A	0	0	15	0	7.21
Feb 09, 2024	2	N/A	0	0	16	0	7.43
Feb 08, 2024	5	N/A	0	1	435	0	497.42
Feb 04, 2024	2	N/A	0	0	33	0	19.86
Feb 03, 2024	2	N/A	0	0	15	0	6.94
Feb 01, 2024	1	N/A	1	0	32	0	28.33
Jan 29, 2024	1	N/A	0	0	30	0	27.0
Jan 28, 2024	2	N/A	0	0	41	0	32.
Jan 27, 2024	8	N/A	1	0	96	0	51
Jan 26, 2024	2	N/A	0	0	69	0	3

**HIS REPORT MAY DISPLAY DRIVING DATA ASSOCIATED WITH OTHER INDIVIDUALS THAT
PERATED THE INSURED'S VEHICLE DURING THE EVALUATION PERIOD.** REASONABLE PROCE
AVE BEEN ADOPTED TO MAXIMIZE THE ACCURACY OF THIS REPORT. THIS REPORT CONTAINS ALL D
ATA FROM THE DATA SOURCE AS OF THE REPORT CREATED DATE

Lon Seidman / @lonseidman

> *@SGgrc: I'am listening to the latest Security Now episode, definitely agree that freezing one's credit needs to be the default position these days. One question though: most of these credit agencies rely on the types of personal information that typically get stolen in a data breach for authentication.*

Certainly a bad actor will go for the lowest hanging fruit and perhaps move on from a frozen account, - but if there's a big whale out there, they may go through the process of unlocking that person's credit and then stealing their money. What kind of authentication changes do you think are needed?

Since I froze my credit reporting I've only had one occasion to temporarily unfreeze it, when I decided to switch to using an Amazon credit card for the additional purchase benefits. That's when I discovered to my delight that it was possible to specify an automatic re-freeze on a timer to prevent the thaw from being inadvertently permanent. Since I had very carefully recorded and stored my previously freezing authentication, I didn't need to take any account recovery measures. So I can't speak from experience. But it occurs to me that strong measures are available. The reporting agencies will have our current home address. So they could use the postal system to send an authentication code via old school paper mail that would be quite difficult, if not effectively impossible, for a criminal located in a hostile foreign country to obtain. So there certainly are strong authentication measures that could be employed if needed.

Eric Berry / @evdaycomputer

@SGgrc: What was that credit link from the podcast, I tried the address you gave out and got page not found

https://grc.sc/credit I just tried the link. It bounced me over to the Investopedia site, as expected: https://www.investopedia.com/how-to-freeze-and-unfreeze-your-credit-5075527

The Monster / @SumErgoMonstro

@SGgrc: The race condition isn't solely solved with the XCG "counter ownership" protocol unless the owner immediately (re)reads the owned memory region to be sure it wasn't altered before it got ownership.

There are aspects of computer science that are absolutely abstract and purely conceptual. I suppose that's one of the reasons I'm so drawn to it. One of the time honored masters of this craft is Donald Knuth and the title of his masterwork being "The Art of Computer Programming" is not hyperbole. There are aspects of computer programming that can be true art and his work is full of lovely constructions similar to the use of single exchange instruction being used to manage inter-thread synchronization.

In this case, as I tried to carefully explain last week, the whole point of using a single exchange instruction is that it's not necessary to reread anything because the act of attempting to acquire the ownership variable acquires it ONLY if it wasn't previously owned by anyone else, while simultaneously (and simultaneity is the point ***and*** the requirement) also returning information about whether the variable was or was not previously owned by any other thread.

If anyone wishes to give their brain a bit more exercise, think about the fact that in an environment where individual threads of execution may be preempted at any instant, nothing conclusive can **ever** be determined by reading the present state of the object ownership variable. Since the reading thread might be preempted immediately following that reading, and during its preemption the owner variable might change, the only thing that anyone reading that variable might learn is that the object being managed was or was not owned at the time of that reading. While that might be of some interest, it's not interesting to anyone who wishes to obtain ownership, since that information was already obsolete the instant it was obtained.

Javamantis / @javamantis1

Regarding episodes 970 and 969 with 'push button hardware config' options, my first thought is of the 2017 Saudi chemical plant attacked with the Triton malware. The admins working on the ICS controllers deliberately left an admin permission key in the controllers instead of walking the 10 minutes to insert the key every time a config needed changing. As a result, the attackers were able to access the IT systems and then the OT systems because the key was always left in and in admin mode. Lazy people will always work around inconvenient, very secure systems.... To 999 and beyond, like Voyager!

I thought that was a very good point. For example, the push button dangerous config change enabler needs to work on a ***change*** from not pushed to pushed rather than on whether the button is depressed. The electrical engineers among us will be familiar with the concept of "edge triggered" versus "level triggered." If this is not the way it's done, people will simply depress the button once, then do something like wedge a toothpick into the button in order to keep it depressed. My feeling is, the ability to bypass well designed and well intentioned security doesn't matter at all. There's a huge gulf separating "secure by design" and "insecure by design" and it's absolutely worth making things "secure by design" even if those features can be bypassed. The issue is not whether they can be bypassed, but that they are there in the first place to perhaps be bypassed.

If someone goes to the effort to bypass a deliberately designed security measure then the consequences of doing that is 100% on them. It's a matter of transferring responsibility. If something is insecure by design then it's the designers who are at fault for designing the system insecurely. They may have assumed that someone would come along and make their insecure system secure, but we have witnessed far too many instances where that never happened. So the entire world's overall net security will be increased if systems start out being secure and are then later, in some instances, forced to operate insecurity.

And if someone's manager learns that the reason the enterprise's entire network was taken over, all their crown jewels stolen and sent to a hostile foreign power, and then all their servers encrypted, is because someone in IT security wedged a toothpick into a button to keep it held down for their own personal convenience... well... you won't be asking that manager for a recommendation on the resume that will soon need updating.

David Sostchen / @sostchen

Hi Mr. Gibson, long time listener and SpinRite owner. I was listening to podcast #955 and I meant to message you about the Italian company Actalis, but life has a tendency to get in the way. They happen to be one of the few remaining companies that issue free S/MIME certificates. I have been using them for years to secure my email. All the best   -David

FeloniousWaffle / @FeloniousWaffle

> *Hi Steve,*
>
> *I created an account on this platform to message you. I cannot wait for your email to be up and running. I was just listening to episode 968 on my commute and believe the outrage of AT&T's encryption practices to be undersold.*
>
> *You mention that if someone is able to decrypt one string to get the four digit code then they have everyone's code who shares the same string. I believe it to be worse than that. Am I wrong in thinking that if they crack 1 they have all 10000?*
>
> *I am making some assumptions that there are only 2 ways that 10000 unique codes produces exactly 10000 unique encrypted strings. The first, and is what I am assuming, AT&T used the same key to encrypt every single code. The second would be to have a unique key for each code. So code 1234 would have a different key than 5678. That seems far fetched to me. Is there an error in my thinking?*
>
> *Thanks for the podcast and everything you do. Glad you are sticking around beyond 999.*
> *Daryle*

So I see what Daryle is thinking. He's assuming that what was done was that if the encrypted string was decrypted to obtain the user's 4-digit passcode then the other 9,999 strings could similarly be decrypted to obtain the other possible 4-digit passcodes. And he's probably correct in assuming that if one string had been decrypted then all the others could be, too. But that isn't what happened. No encrypted strings were ever decrypted and the encryption key was never learned. But due to the static nature of the passcode encryption, none of that was necessary.

I wanted to share Daryle's note because it reveals an important facet of cryptography, which is that it's not always necessary to reverse a cryptographic operation, as in decryption in this case but it's also true for hashing, in order to obtain important information. If the results of going in the forward direction only can be reapplied to other instances, then a great deal can be learned. In this case, since people tended to use highly non-random passcodes, reusing their birthday, their house's street number, or the last four digits of their phone number or social security number – all things that were also part of the exfiltrated data set – and assuming a fixed mapping between their plaintext passcode and its encryption, examining, for example, the details of all of the records having a common encrypted passcode would very quickly reveal what single passcode all of those otherwise unrelated records shared.

For example, one household lived at 1302 Willowbrook, whereas the birthday of someone else was February 13th and someone else's phone number ended in 1302. So by seeing what digits were common among a large group of records all sharing only the same encrypted passcode, it would quickly become clear what identical passcode they all chose.

SKYNET / @fairlane32

> *Hi Steve, Would having DRAM catch up and be fast enough eliminate the GhostRace issue?*

That’s a very interesting question. The question could be reframed a bit to further clarify what we’re really asking. So let’s ask: If all of the system’s memory were located in the processor’s most local, instant access L1 cache, that is, if its L1 cache were 16 gigabytes in size, so that no read to or write from that main memory took any time, would speculative execution still present problems? And I believe that the answer is yes.

Even in an environment where access to memory is not an overwhelming factor, the work of the processor itself can still be accelerated by allowing it to be more clever about how it spends its time. Today’s processors are not executing instructions one at a time. And, in fact, processors haven’t actually been executing one instruction at a time for quite a while. The concept of “out of order” instruction execution dates way back to the early CDC (Control Data Corporation) 6600 mainframe which was the first commercial computer system to implement out-of-order instruction execution. It sucked in instructions ahead of them being needed and when it encountered an instruction whose inputs and outputs were independent of any earlier instructions that were still being worked on, it would execute that later instruction in parallel with other ongoing work. The same sort of instruction pipelining still goes on today, and we would still like our processors to be faster. If a processor had perfect knowledge of the future by knowing which direction it was going to take at any branch or where a computed indirect jump was going to land it, it could reach its theoretical maximum performance given any clock rate. But since a processor’s ability to predict the future is limited to what lies immediately in front of it, it must rely upon looking back at the past and using that to direct its guesses, or its speculation, about its immediate future.

Here’s something to think about: The historical problem with 3rd party cookies has been that browsers maintained a single large shared cookie jar. So an advertiser could set its cookie while the user was at site ‘A’ and read it back when the same user moved to site ‘B’. This was never the way cookies were meant to be used. They were meant to be used in a 1st-party context to allow sites to maintain state with their visitors. The problem is that, until very recently, there has been no cookie compartmentalization.

We have the same problem with microprocessor speculation that we have had with 3rd party cookies – lack of compartmentalization: The behavior of malware code is affected by the history of the execution of the trusted code that ran before it. Malware is able to detect the behavior of its own code which gives it clues into the operation of previous code that was running in the same set of processors. In other words, lack of compartmentalization:

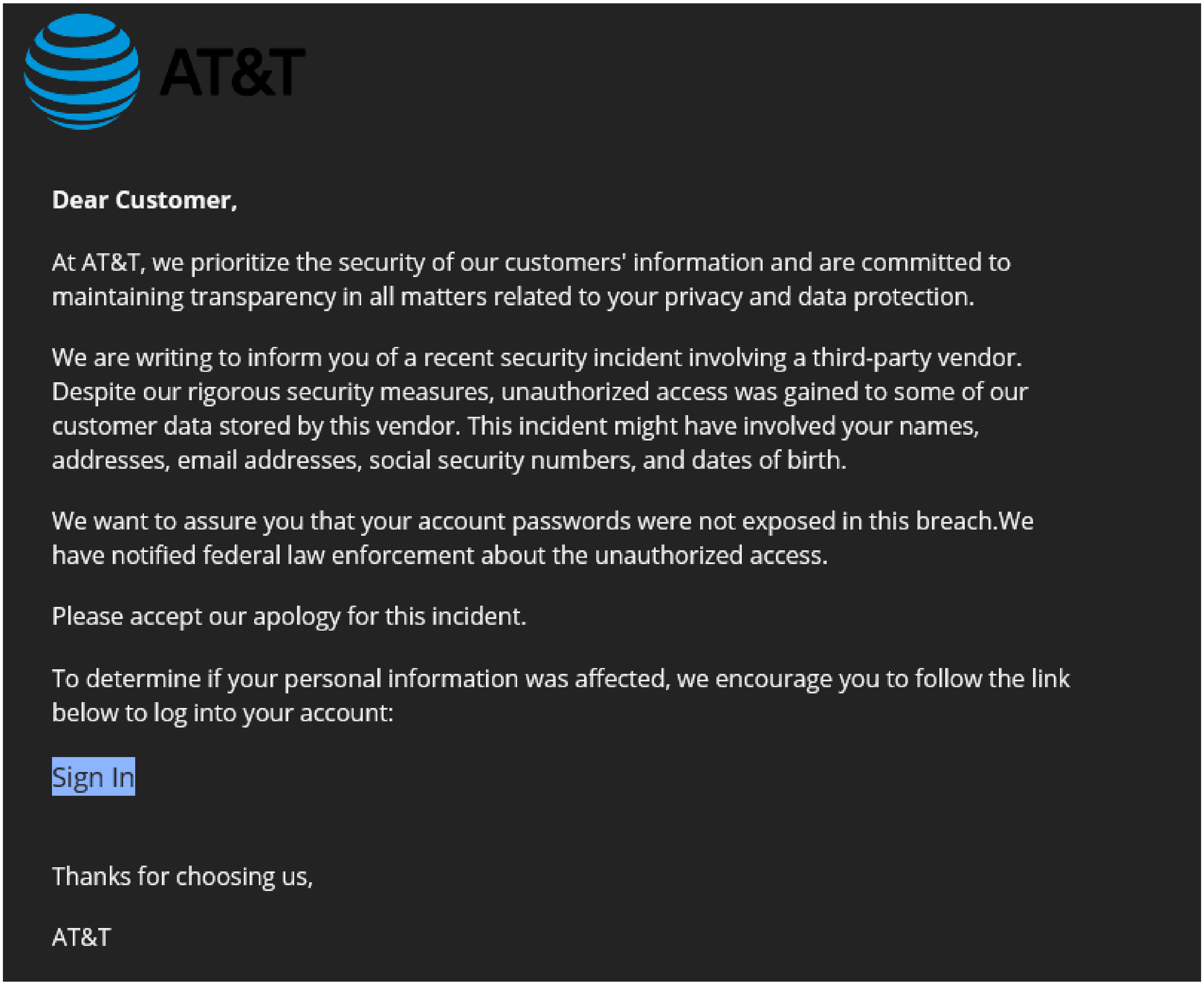
Malicious code is sharing the same microarchitectural state as non-malicious code – because today there’s only one set of state. That’s what needs to change... and I would be surprised if Intel wasn’t already well on their way to implementing this sort of change. I have no idea how large a modern microarchitecture’s speculation state is. But the only way I can see to maintain the performance we want today in an environment where our processors might be unwittingly hosting malicious code is to arrange to save and restore the microprocessor’s speculation state whenever the operating system switches process contexts.

It would make our systems even more complicated than they already are, but it would mean that malicious code could no longer obtain any hints about the operation of any other code that was previously using the system.

I'll omit this listener's full name since it's not important. John wrote:

> *I got nailed in a phishing email for AT&T, see attached picture. No excuse but atleast i realized it immediately and changed my password (which is not one that has been used anywhere else of course) Feel stupid...*

Here's what he received:



AT&T

Dear Customer,

At AT&T, we prioritize the security of our customers' information and are committed to maintaining transparency in all matters related to your privacy and data protection.

We are writing to inform you of a recent security incident involving a third-party vendor. Despite our rigorous security measures, unauthorized access was gained to some of our customer data stored by this vendor. This incident might have involved your names, addresses, email addresses, social security numbers, and dates of birth.

We want to assure you that your account passwords were not exposed in this breach.We have notified federal law enforcement about the unauthorized access.

Please accept our apology for this incident.

To determine if your personal information was affected, we encourage you to follow the link below to log into your account:

Sign In

Thanks for choosing us,

AT&T

What makes this particularly insidious is that, as we've covered during the previous two weeks, AT&T is authentically sending out authentic breach notifications to its 51 million current and previous customers. What I assume they would not be doing is ending their note with a solicitation to the recipient to log into their account. So this must be what John realized. He clicked "Sign In" and was taken to a look-alike phishing site which wanted him to log in. Had he done so, and perhaps he did, he would have actually been providing the phishermen with his username and password.

This is worth sharing since we really do need to be continuously vigilant.

TomMinnick / @TomMinnick

With these "atomic operations" to mitigate race conditions. How does that work with multi core processors when multiple threads are running in parallel? Couldn't a race condition still occur? I probably don't understand enough about how multicore processors handle threads.

Tom's question is a terrific one and it occurred to many of our listeners. He and everyone were right to wonder. The atomicity of an instruction only applies to the threads running on a single core since it can only be doing one thing at a time; but what of multi-core or multi-processor systems? The issue is important enough that all systems provide some solution for this. In the case of the Intel architecture, there's a special instruction prefix called "Lock" which, when it immediately precedes any of the handful of instructions that might find it useful, forces the instruction that follows it to also be "atomic" in the sense of multiple cores or multiple memory-sharing processors. Only one processor at a time is able to access the targeted memory location. And there's one other little tidbit: The simple exchange instruction is so universally useful for thread synchronization that the Lock prefix functionality is built-in. Any time exchange is being used to exchange the contents of a memory location with the contents of a register, doing that is automatically not only thread-safe but also multiprocessor-safe.

Michael Hagberg / @MichaelHagberg

Credit Freeze: Rather than unlock your entire account, it should work this way: I'm buying a car, the dealer tells me which credit service they use and their ID number. I go to the credit service website, provide my SSN, PIN (assigned by the site when I froze it) and the car dealer's ID number. My account will then allow that car dealer only to access my account for 24 hours.

Michael, I agree 100%... and this just shows us that the child in you has not yet been beaten into submission and that you are still able to dream big. More power to you! Wouldn't it be nice if the world were so well designed. It's just not clear to me how we get there from here.

SpinRite

mike shales / @mikeshales

> *Recently I've run into some issues with my old iMac (mid 2017 model). I have wanted to support your valuable Security Now efforts for some time, but investing the time to see if I could even run SpinRite on my Macs when they were running without problems discouraged me. But now…*
>
> *You mentioned on your April 9 podcast: I wanted to remind any would-be Mac purchasers that this is the reason I created GRC's freeware named "Bootable" in favor of "DOS Boot". If you can get Bootable to congratulate you on your success in booting it, then exactly the same path can be taken with SpinRite...*
>
> *But Bootable is a Window .exe file and needs a Windows machine to create a bootable USB flash drive, right? Lacking a Windows machine I made a bootable DOS drive from your ReadSpeed IMG download.*
>
> *Following instructions from ChatGTP I used dd to write the ReadSpeed IMG to a 4 GB flash drive. Then following instructions in the GRC forum post "Boot a Mac into FreeDOS for SpinRite or ReadSpeed" I succeeded in booting my iMac into DOS and running ReadSpeed!!*
>
> *So far so good, BUT... I believe the current SpinRite 6.1 includes the capability to recognize more drives than previously and might rely on features not provided in the version of DOS that I now have installed on my flash drive. If so, perhaps downloading the SpinRite 6.1 .exe file and copying it to my flash drive might not be ideal. Is this an issue?*
>
> *Thanks for your help!! Mike*

Mike very cleverly arranged to use various tools at GRC (and ChatGPT amazingly enough) to create a bootable USB drive which successfully booted his mid-2017 iMac. So, first responding to Mike directly: Everything you did was 100% correct and if you place your copy of the SPINRITE.EXE into that USB stick and boot it, everything will work perfectly. And if you run it at Level 3 on any older Macs with solid state mass storage you can expect to witness a noticeable and perhaps even very significant subsequent and long lasting improvement in the system's performance. And while it won't be obvious, there's also very good reason to believe that in the process you will have significantly improved the system's reliability. The reason the SSD will now be faster is that it's needing to struggle much less to return the requested information. We are going to be learning far more about this during the work on SpinRite 7, and although 6.1 is a bit of a blunt instrument in this regard, it works and it's here today. To Mike's question, the specific version of FreeDOS doesn't matter at all, since DOS is only used to load SpinRite and to write its log files. Otherwise SpinRite ignores DOS and interacts directly with the system's hardware.

I wanted to share Mike's question because I just finished making some relevant improvements. He mentioned correctly that BootAble is Windows-only freeware. But over the weekend the BootAble download was changed from an EXE to a ZIP archive and the ZIP archive now also contains a small BootAble file system image file which can be used by any Mac or Linux user to directly create a BootAble boot-testing USB drive.

One of the guys in GRC's web forums put me onto a perfect and easy-to-use cross platform freeware file image installer known as Etcher by a company called Balena. It's perfect for any Mac or the less techie Linux user since it was expressly made to be impossible to mess up. Mike and ChatGPT managed to get the job done but now there's an easy to use solution for answering the question of whether SpinRite 6.1 will be able to run on any given machine.

And as is clear, I'm still scrambling to pull together all of the post release pieces. At the moment, Mac and Linux users can visit grc.com/upgrade.htm to download a directly bootable SpinRite image without the user of Windows. But my next order of business after today's podcast is to integrate that into our existing product download links.

Sean 🏳️‍🌈 🇨🇦📷 / @DarkElfLX

> *Hey Steve, I'm sure you're hearing this a lot, but Windows did not trust Spinrite despite all your signing efforts. I had to clear 3 Severe warnings before it would allow me to keep 6.1 on my system for use. I hope it gets better soon for users less willing to ignore the scary warnings from Microsoft. /Sean*

Yep. I don't recall whether I had mentioned it here also, since I've participated in a lot of discussion about this in GRC's newsgroups. One thing that's been learned is that Microsoft has decided to deprecate any and all special meaning for EV, extended validation, code signing certificates. So all of those hoops I jumped through to get remote server-side EV code signing to work turn out not to be useful going forward. Now that I have it, it's certainly useful to have GRC's code signing certificate protected by an HSM – hardware security module – but it will no longer have any value for providing an additional assertion of signed code integrity.

When I first encountered this I reached out to Jeremy Rowley, my friend and primary contact at DigiCert to ask him if I had read Microsoft's announcement correctly and he confirmed that Microsoft had just surprised everyone in the CAB forum with that news. Apparently, what's at the crux of this is that for a while, end users were able to use EV code signing certificates to sign kernel drivers. That was the thing Microsoft most cared about as far as EV was concerned. But after the problems with malicious Windows drivers, Microsoft has decided to take that away and require that only THEY, meaning Microsoft, will be authorized to sign Windows kernel drivers in the future. In their eyes, this eliminated the biggest reason for having and caring about EV code signing certs, so while they will continue to be honored for code signing, they will no longer confer any benefit over non-EV.

I suspect that the biggest problem is that something Windows Defender sees inside SpinRite 6.1's code absolutely convinces it that this is a very specific Trojan named "Wacatac.B". If I knew what part of SpinRite was triggering these false positives I could probably change it there. I have some ideas, so we'll see what I can arrange. In any event, the interesting news is that Microsoft has indeed completely deprecated any special meaning for extended validation code signing certificates and only they will be able to sign kernel drivers going forward.

Chat (out of) Control

Across the pond from the US, the EU is continuing to inch forward on their controversial legislation, commonly referred to as "Chat Control", which proposes to require providers of encrypted messaging services to somehow arrange to screen the content that's carried by those services for child sexual abuse material, commonly known as CSAM. As I said when we last looked at this last year, 2024 will prove to be quite interesting since all of this will likely be coming to a head this year. What's significant about what's going on in the EU, unlike in the UK, is that the legislation's language carries no exclusion over the feasibility of performing this scanning.

Just to remind everyone who has a day job and who might not be following these political machinations closely, last year the UK was at a similar precipice with their own legislation and at the 11th hour they added some language that effectively neutered that legislation while allowing everyone to save face.

For example, last September 6th, ComputerWorld's headline read *"UK rolls back controversial encryption rules of Online Safety Bill"* and followed that with *"Companies will not be required to scan encrypted messages until it is "technically feasible" and where technology has been accredited as meeting minimum standards of accuracy in detecting only child sexual abuse and exploitation content."* Since it's unclear how any automated technology might successfully differentiate between child sexual abuse material and a photo a concerned mother might send of her child to their doctor, there's little concern that the high bar of "technical feasibility" will be met in the foreseeable future. While the UK came under some attack for "punting" on this, the Big Tech companies all breathed a sigh of relief.

But so far, there is no sign of the same thing happening in the EU – not even a murmur of it. One of the observations we've made about all such legislation was the curious fact that, if passed, the legislation would mean that the legislator's own secure, encrypted and private communications would similarly be subjected to surveillance and screening. ***Or would they??***

Two weeks ago, on April 9th, the next iteration of the legislation appeared in the form of a daunting 203-page tome. Fortunately, the changes from the previous iteration were all shown in bold type, making it at least somewhat possible to see what's changed. This was brought to my attention by the provocative headline on an EU website *"#ChatControl: EU ministers want to exempt themselves"* and that article went on to say:

> *According to the latest draft text of the controversial EU Child Sexual Abuse Regulation proposal leaked by the French news organization Contexte, which the EU member states discussed, the EU interior ministers want to exempt professional accounts of staff of intelligence agencies, police and military from the envisioned scanning of chats and messages (Article 1 (2a)). The regulation should also not apply to "confidential information" such as professional secrets (Article 1 (2b)). The EU governments reject the idea that the new EU Child Protection Centre should support them in the prevention of child sexual abuse and develop best practices for prevention initiatives (Article 43(8)).*

The EU has something called the “Pirate Party” which is formed from a collection of many member parties across and throughout the European Union. The Party was formed ten years ago, back in 2014, with a focus upon Internet governance. So the issues created by this pending legislation is of significant interest to this group. To that end, one of the members of Parliament, Patrick Breyer, had this to say about these recent changes to the proposed legislation:

> *“The fact that the EU interior ministers want to exempt police officers, soldiers, intelligence officers and even themselves from chat control scanning proves that they know exactly just how unreliable and dangerous the snooping algorithms are that they want to unleash on us citizens. They seem to fear that even military secrets without any link to child sexual abuse could end up in the US at any time. The confidentiality of government communications is certainly important, but the same must apply to the protection of business and of course citizens communications, including the spaces that victims of abuse themselves need for secure exchanges and therapy. We know that most of the chats leaked by today’s voluntary snooping algorithms are of no relevance to the police, for example family photos or consensual sexting. It is outrageous that the EU interior ministers themselves do not want to suffer the consequences of the destruction of digital privacy of correspondence and secure encryption that they are imposing upon us.”*
>
> *“The promise that professional secrets should not be affected by chat control is a lie cast in paragraphs. No provider and no algorithm can know or determine whether a chat is being conducted with doctors, therapists, lawyers, defense lawyers, etc. so as to exempt it from chat control. Chat control inevitably threatens to leak intimate photos sent for medical purposes and trial documents sent for defending abuse victims.”*
>
> *“It makes a mockery of the official goal of child protection that the EU interior ministers reject the development of best practices for preventing child sexual abuse. It couldn’t be clearer that the aim of this bill is China-style mass surveillance and not better protecting our children.*
>
> *Real child protection would require a systematic scientific evaluation and implementation of multidisciplinary prevention programmes, as well as Europe-wide standards and guidelines for criminal investigations into child abuse, including the identification of victims and the necessary technical means. None of this is planned by the EU interior ministers.”*

After the article finished quoting this Patrick Breyer, it noted that the EU governments want to adopt the chat control bill by the beginning of June. We’re approaching the end of April, so the only thing separating us from June is the month of May. I was curious to see whether the breadth of the exclusion might have been overstated in order to make a point, so I found the newly added section of the legislation on page 6 of the 203-page PDF.   It reads:

> *(12a) In the light of the more limited risk of their use for the purpose of child sexual abuse and the need to preserve confidential information, including classified information, information covered by professional secrecy and trade secrets, electronic communications services that are not publicly available, such as those used for national security purposes, should be excluded from the scope of this Regulation. Accordingly, this Regulation should not apply to interpersonal communications services that are not available to the general public and the use of which is instead restricted to persons involved in the activities of a particular company, organization, body or authority.*

I'm not trained in the law, but that doesn't sound to me like an exclusion for legislators who would probably be using iMessage, Messenger, Signal, Telegram, WhatsApp, etc. It says *"this Regulation should not apply to interpersonal communications services that are **not** available to the general public."* So, you know, internal proprietary intelligence agency communications.

Remember that it's this proposed EU legislation which includes the detection of "grooming" behavior in textual content. So it's not just imagery that needs to be scanned, but the content of all textual messaging. We're also not talking only about previously known and identified content which is apparently circulating online but also anything the legislation considers "new" content.

As I read through section after section of what has become a huge mess of extremely weak language that leaves itself open to whatever interpretation anyone might want to give it, my own lay feeling is that this promises to create a huge mess. I've included a link to the latest legislation's PDF in the last page of the show notes for anyone who's interested:

https://www.patrick-breyer.de/wp-content/uploads/2024/04/2024-04-11-conseil-csam-compromis-9-avril.pdf

You'll only need to read the first eight pages or so to get a sense for just what a catastrophe this promises to be. As is the case with all such legislation, what the lawmakers say they want, and via this legislation will finally be requiring, is not technically possible. They want detection of previously unknown imagery and textual dialog which might be seducing children while at the same time honoring and actively enforcing EU citizen privacy rights. Oh, and did I mention that 78% of the EU population that was polled said that they did not want any of this?

And it occurred to me that encryption providers cannot just say they're complying when they are not, because activist children's rights groups will be able to trivially test any and all private communications services to verify that they do, in fact, detect and take the action that the legislation requires of them. All that's needed is for such groups to register a device as being used by a child, then proceed to have a pair of adults hold a seductive grooming conversation and perhaps escalate to sending some naughty photos back and forth. And you can believe that if the service they're testing doesn't quickly identify and red-flag the communicating parties involved, those activist children's rights groups will be going public with the service's failure under this new legislation.

I've said it before, and I understand that it can sound like an excuse and a cop out, but not all problems have good solutions. There are problems that are fundamentally intractable. This entire debate surrounding the abuse of the absolute privacy created by modern encryption is one such problem. This is not technology's fault. Technology simply makes much greater levels of privacy practical, and people continually indicate that's what they prefer. As a society we have to decide whether we want the true privacy that encryption offers, or whether we want to deliberately water it down in order to perhaps prevent some of the abuse that absolute privacy also protects.