Security Now! #961 - 02-13-24

Bitlocker: Chipped or Cracked?

This week on Security Now!

What's the story behind the massive incredible 3 million toothbrush takeover attack? How many honeypots are out there on the Internet? What's the best technology to use to access your home network while traveling? Exactly why is password security all just an illusion? Does detecting and reporting previously used passwords create a security weakness? Will Apple's opening of iOS in the EU drive a browser monoculture? Can anything be done to secure our router's UPnP? Has anyone encountered the "Unintended Consequences" we theorized last week? Are running personal eMail servers no longer practical? And what's up with the recently reported vulnerability in many TPM-protected Bitlocker systems?

Your municipal tax dollars, hard at work...

Security News

Brushing up on the facts

Just as we were recording last Tuesday's podcast news was breaking across the Internet that somewhere around 3 million electric toothbrushes had all been compromised and were enslaved into a massive global botnet. Breathless headlines reported:

- "Millions of hacked toothbrushes used in Swiss cyber attack, report says" – *The Independent*.
- "Hackers turn toothbrushes into cyber weapons" – *Fudzilla*.
- "Millions of smart toothbrushes used in botnet attack on company"" – *Boing Boing*.
- "3 million smart toothbrushes were just used in a DDoS attack. **Really**" – *ZDNet*.
- "Three million malware-infected smart toothbrushes used in Swiss DDoS attacks — botnet causes millions of euros in damages" – *Tom's Hardware*
- "Over 3 million toothbrushes 'hacked' and 'turned into secret army for criminals,' experts claim" – *The Sun*.

There were many many more similar reports, yet none of it was true. Highly respected news outlets repeated the story because, well, talk about click-bait!

How exactly this massive reporting screw-up came to pass remains unclear because the parties who were directly involved disagree about who said what to whom. But make no mistake that what was originally published was hair curling, if not teeth straightening. Here's a piece of what the world read:

She's at home in the bathroom, but she's part of a large-scale cyber attack. The electric toothbrush is programmed with Java, and unnoticed criminals have installed malware on it and approximately 3 million similar toothbrushes. One command is enough and the remote-controlled toothbrushes simultaneously access the website of a Swiss company. The site collapses and is paralyzed for four hours resulting in millions of dollars in damage.

This example, which seems like a Hollywood scenario, actually happened. It shows how versatile digital attacks have become. Stefan Zügerm head of the Switzerland offshoot of the cybersecurity specialist firm Fortinet said: "Each device connected to the Internet is a potential goal – or can be misused for an attack," Whether baby monitor, web camera or the electric toothbrush, the attackers do not care.

The day after hundreds of media outlets worldwide repeated the false claim that a botnet of three million toothbrushes had attacked a Swiss company, Fortinet, the now quite embarrassed cybersecurity firm that was at the center of the story has issued a statement:

"To clarify, the topic of toothbrushes being used for DDoS attacks was presented during an interview as an illustration of a given type of attack, and it is not based on research from Fortinet or FortiGuard Labs. It appears that due to translations the narrative on this topic has been stretched to the point where hypothetical and actual scenarios are blurred."

Right. Fortinet went on to say that its experts have *"not observed Mirai or other IoT botnets target toothbrushes or similar embedded devices."*

Graham Cluley, who has been following this mess wrote last Thursday the 8th:

I can imagine how a Fortinet researcher might have regaled a journalist with tales of how IoT devices like webcams could be hijacked into botnets for DDoS attacks (after all, this has happened.) However, giving the journalist a juicy hypothetical example of millions of smart toothbrushes taking down a Swiss company is playing a dangerous game. I'm not surprised that journalists seized the story, and as we've seen, and then other news outlets repeated it without double-checking its truth. A more experienced spokesperson would have made it clear that the toothbrush DDoS attack example was hypothetical and hadn't actually happened.

Failing that, since the original article was published on January 30th, Fortinet had plenty of time to contact the Swiss newspaper and correct the report, or post a clarification on social media debunking the story as the hysteria spread in the press. But Fortinet didn't, until skeptical voices in the cybersecurity community questioned the story.

Ironically, Fortinet's researchers have published some genuinely interesting proof-of-concept research in the past on the toothbrush topic – albeit hacking Bluetooth-enabled toothbrushes to mess with brushing time rather than knock a company's website offline.

Many of the various publications that were forced to update, amend and retract what turned out to be erroneous stories took the time to add that while, yes, whoops!, this didn't **actually** happen, it was still an entirely possible and even likely scenario. And that may also account for everyone rushing to submit the story: Even though it was not true, it carried the ring of truth for any tech publication since, as everyone listening to this podcast knows, routers and security cameras and IoT devices of all makes, models and functions are indeed being compromised and enlisted in botnets daily. It is not science fiction... even though this particularly intriguing story was pure fiction.

"There are too many damn Honeypots!"

I got a kick out of the blog headline posted at the VulnCheck website. It read "There Are Too Many Damn Honeypots!" Here's what the VulnCheck guys explained. They wrote:

Determining the number of internet-facing hosts affected by a new vulnerability is a key factor in determining if it will become a widespread or emergent threat. If there are there a lot of hosts affected there's a pretty good possibility things are about to pop off. But if only a few hosts are available for exploitation, that's much less likely. But actually, counting those hosts has become quite a bit more challenging.

*Take for example, CVE-2023-22527 affecting Atlassian Confluence. At the time of writing, Confluence has appeared on the CISA KEV [Commonly Exploited Vulnerabilities] list nine (yes, nine) times. That's a level of exploitation that should encourage everyone to get their Confluence servers off the internet. But let's look for ourselves. There are a number of generic Confluence Shodan queries floating around, but **X-Confluence-Request-Time** might be the most well known (this simply checks for an HTTP response header value).*

In other words, the Shodan Internet search scanner makes an HTTP query to Confluence's service port and the reply coming back from that port contains the reply header "**X-Confluence-Request-Time**" which would suggest that there's a running Confluence server answering queries at that IP and port. The VulnCheck guys then show a Shodan screen capture showing 241,702 occurrences of that reply header across the Internet.
They then point out one in particular, writing:

241,000 hosts is a great target base for an emergent threat! But, on closer examination, there's something off about the listed hosts. For example, this one has the Confluence "X-Confluence-Request-Time" header, but it also has an F5 favicon [as in the well known security firm F5 Systems], and it also claims to be a QNAP TS-128A. This is a honeypot.

Whoever created this honeypot was somewhat clever. They mashed together the popular Shodan queries for Confluence, F5 devices, and QNAP systems, to create an abomination that would show up in all three queries.

To avoid throwing exploits all over the internet (and thus getting quickly caught), some attackers use Shodan (or similar) to curate their target lists. This honeypot is optimized for this use case. Which is neat, but it blocks our view of what is real. Can we filter them out of our search?

At this point, it's probably useful to look at what a real Confluence server HTTP response looks like. The server has a number of useful headers to key off of, but we'll try to filter by adding in Set-Cookie: JSESSIONID=. That update brings the host count down from 241,702 to just 37,964 probably-actual Confluence servers publicly exposed to the Internet. But is that number real?

It still seems high because most of those do not respond with an actual Confluence landing page. A simple way to capitalize on that is to also search for a snippet from the Confluence login page in our search criteria: `html:"confluence-base-url":` *Ah... now we're down to 20,584, a little over half as many as before.*

They write:

That knocks off ~17,000 hosts, and things are looking more Confluency. But there seems to be a whole bunch of entries without favicons. Let's drill down into one and see…

They do that, looking at the presence or lack of any FAVICON for the site. And at one point it occurs to them to examine the value being returned in the Confluence JSESSIONID reply header ... and what do you know, a great many of those values are identical! They aren't being generated dynamically. They're part of some fixed Confluence simulating honeypot, and the simulation took some shortcuts which give it away when it's examined closely enough.

By applying this spoofed JSESSIONID filter, the number drops to 4,187 probably authentic publicly exposed Confluence servers. Again, they write:

A quick investigation suggests that this could be the complete set of real Confluence hosts (or just very very good honeypots). That's a reduction from around 240,000 hosts all the way *down to just 4,200. That means there are approximately* ***236,000 Confluence honeypots*** *on the internet or more than* ***50 times the actual number of real Confluence servers.***

A vulnerability that only impacts 4,000 hosts is much less concerning than a vulnerability that impacts 240,000. Understanding the scale of an issue is important, and therefore, being precise about the number of potentially impacted hosts is important too. Those who copy overinflated statistics or haven't done their due diligence are making vulnerabilities appear more impactful than they truly are. [3 million toothbrushes anyone?]

While we focused on Confluence, this particular problem has been repeated across many different targets. Honeypots are a net good for the security community. But their expanding popularity does make understanding real-world attack surfaces much more difficult for defenders, not just attackers.

This will be a very good rule of thumb for us to keep in mind moving forward. Academically, it's interesting that the explosion in honeypot use and population is this large. That's sort of astonishing. But this means that the tendency to immediately rely upon and believe the results of a simple Shodan search for a given open port – assuming that means there's truly a vulnerable service running there – needs to be significantly tempered. And it also suggests that future Internet vulnerability scanners will need to do a better job of filtering out the honeypots since the problem has obviously become massive.

Closing the Loop

Dextra / @Cholula_It_Up

Hello Steve - Thank you for introducing me (+13yrs ago) to the world of being security minded from a tech perspective. I travel a lot and, over the years, have been working on trying to come up with a solution where I can appear on my home network so I can access / watch content on my cable provider's app while being secured with the least amount of possibility of opening my home router up to external threats.

I've a Synology RT2600ac router at home. I've recently started to travel with a Beryl travel router. I do have an extra Synology RT2600ac router that I've traveled with in the past. Do you have any suggestions on how to go about appearing to be on my home network in a secure manner so I can access my cable provider's catalog/live tv? RM

Ten years ago the standard generic answer would have been to arrange to set up a VPN server at home and then VPN into your home network from afar. That's no longer the optimal solution.

Among other things, it's often more easily said than done, and it requires opening a static port through your home router which is then visible to everyone on the Internet, not just you. While there are ways to do this safely, it's no longer necessary thanks to the widespread availability of many free and terrific overlay networks.

The very early such network we talked about many years ago was Hamachi. It was originally free, then it went paid, and then was purchased by LogMeIn. It's still possible to use LogMeIn's Hamachi for $50 per year, but many free solutions exist. Nebula, TailScale and ZeroTier are three of the most popular.

Since I didn't know anything about Synology's RT2600ac router, I went over to Michael Horowitz's astoundingly useful and comprehensive "RouterSecurity.Org" site. His site allowed me to quickly learn (https://routersecurity.org/synology.php) that the router has some possible use as a VPN client, but it doesn't appear to be general purpose enough to host an overlay network itself. So this would mean that when traveling, some machine inside your home network would need to be left running… but that could just be a Raspberry Pi serving as a quiet network node. Then you would run another node on the laptop you're traveling with and you would be all set. Your laptop and your cable provider's catalog and video streaming would see that you were connecting to them from home.

As I've mentioned before, I haven't had the chance to do this myself since I haven't been traveling… but the next time I'm out and about I'll make time to check out the various overlay network solutions. The response from our listeners who **have** bitten the bullet and set up overlay networks has been universally positive, explaining that they've had a difficult time believing that it really could be so simple to create a state of the art secure connection between any two machines across the public Internet. But that day has truly arrived.

Evan Phyillaier / @Never3ndingCode

Hey Steve, love the show! I run an eCommerce site and my customers have been asking for an easier way to log in. I was wondering if there are any security considerations for going passwordless via email only. The system I would like to set up is registration and login via email (ie. customer just enters their email and then receives a 6 digit code in their email to authenticate and log in). Is this just as secure as email+password authentication? Thanks!

What an interesting and intriguing question. Let's answer the last question first: "*Is this just as secure as email+password authentication?*" At first we might be tempted to answer "No, it cannot be as secure since we've eliminated the "something you know" factor from the login. But of course that's a red herring, right? Since, as I've often noted, every login everywhere on the planet always and without fail has the obligatory "I forgot my password" link. And, sadly, we're now also seeing "I can't use my authenticator" links, too.

And I've even made that into a joke, where someone explains that they don't need no stinking password manager while they're creating an account by just mashing on their keyboard to fill-in their password field. When they're asked "but but but how do you login again later?" they glibly explain that they just click the "I forgot my password" link, then click on the link in their eMail they receive.

The point, of course, is that so long as all username and password logins include the "I forgot what I was supposed to remember" get out of jail free link, our ownership over and control of our eMail is the only actual security we have. The rest is just "feel good" security illusion. This, in turn, means that the service the password and a password manager are ***actually*** performing is only "login acceleration." If your password manager is able to supply the password quickly and painlessly, then the slower "I forgot my password" login process, which is always available using an eMail loop, can be bypassed.

As Bruce Schneier would describe it: "The password is just security theater." Calling passwords a "login accelerant" is the perfect context to put them in.

So let's return to Evan's interesting question: Is eMailing a one-time passcode to someone who wishes to login just as secure as using a password? It should be clear that the correct and defensible answer, is **Yes**. If the users of his eCommerce site do not wish to be hassled for a password, there is no reduction in security to eliminating passwords and using an eMail loop.

However, there's also no need for even a 6-digit code, since that does not provide any additional security and it's more hassle which Evan and his users wish to avoid. What Evan wants to verify is that someone who is wishing to login – at this moment – is in control of their previously registered eMail account. Remember, that's the same fallback test that's being used by every login challenge in the world. This means that all Evan needs to do is eMail this user a direct login link which contains a one-time passcode as a parameter. And since the user no longer needs to transcribe it, the passcode can be as long as Evan wishes. 32 digits? No problem.

The only requirement for security is that the code must be unpredictable and only valid the first time it is used. So we start with a monotonically increasing 32-bit counter. That'll be good for 4.3 billion logins. You could make it 64 bits if you like, so that the most significant 32-bit counter is incremented if the lower 32-bits should ever overflow, even though that would seem to be quite unlikely. So we have a binary value that will never repeat since it's a simple counter that only ever counts upward, never to repeat. We can do several different things with it. It could be fed into the AES Rijndael Cipher which is keyed with a random secret and unchanging key. Then the 128 bits that are output are run through a Base64 converter to produce 22 ASCII text characters. Since the key will never be changed, and the input to the cipher is an upward counting counter, the output will never repeat and it will be cryptographically unpredictable.

Alternatively, a salted hash could be used with a secret salt. The counter's value is hashed and its output is similarly converted into text. It's true that there's an infinitesimally small chance of a hash collision where two different counter values might produce the same output, but any good hash will be cryptographically secure and remember that any single bit which changes in the hash's input will, on average, case half of the output bits to change. So collisions will not be a problem.

So now we have a one time token.

Evan's eCommerce system should append that token to a link that's sent to the individual who just asked to login to his system. The instructions in the eMail are to simply click the button in the eMail. They do that, this confirms that someone who provided the eMail address is receiving eMail there, and they're instantly logged in.

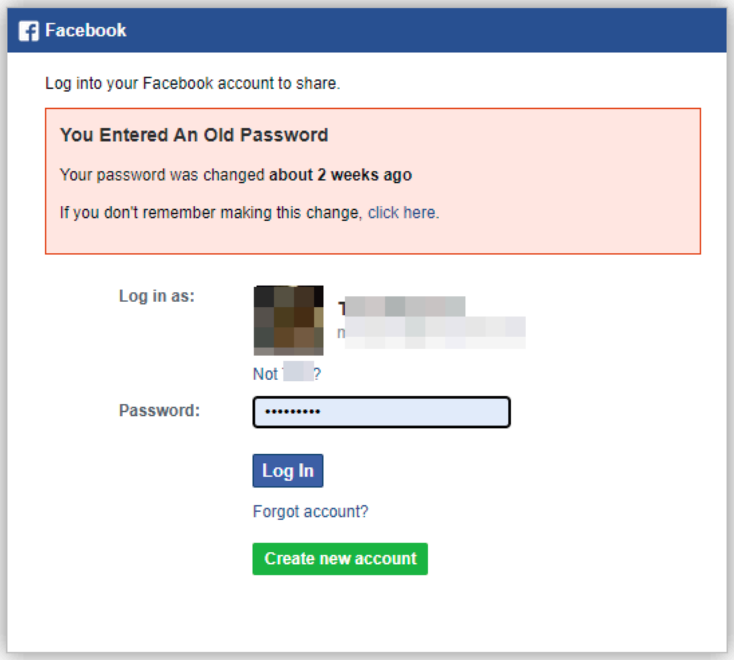
At Evan's end, when the token is obtained and the eMail is sent, those two items along with a timestamp are added to a "Pending Login" list. Anytime someone clicks a link, the list is scanned searching for a matching. The objects on this "Pending Logins" list should use the timestamp of their creation to self-expire. The way I've organized my expiring lists, technically called a queue, is that as I'm traversing the list from its start, I'm also checking timestamps of every object for their age and removing any stale pending login objects on the fly. That way, the list is self-pruning. If a match is found and the object's age is recent enough, the object is removed from the list and the user is logged on based upon the eMail address to which the token was sent as recorded in the list's object.

This simple system gives us everything we want: We have unpredictable self-expiring single-use tokens that can be used to securely confirm an eMail loop… and Evan's users no longer ever need to mess with any password. They simply go to a login page, enter their previously registered eMail address, click the "eMail me a link" button, open their eMail and click the link they have just received and they're done. No passwords to worry about and every bit as secure.

margrave / @margrave (Tom)

Hey Steve, I've been a loyal listener since the early days and though I'm not a security expert, I work in Software Quality Automation and have found the Security Now podcast incredibly helpful several times.

I recently created a LinkedIn article and was given the option to share it on social media. When I chose Facebook, I encountered an interesting situation. I remembered changing my password, but it struck me as odd that Facebook would notify me about it.


Facebook

Log into your Facebook account to share.

You Entered An Old Password

Your password was changed **about 2 weeks ago**

If you don't remember making this change, click here.

Log in as:

Password: •••••••••

I'm not entirely sure if this is a positive or negative feature for Facebook. Sure, Facebook is often filled with a lot of random stuff (like pictures of cats in sunglasses, chickens wearing hats, breathtaking sunsets from someone's backyard, and other equally ridiculous images), but this made me ponder about the implications of such notifications. I'm curious to hear your thoughts, as well as those of other listeners, on this feature Facebook is offering.

Also, I'm eagerly awaiting Spinrite 6.1! It's been a fantastic tool, and I appreciate all the other facets of your podcast, including your involvement with Vitamin D3. Best regards, Tom

I don't see any downside to this and given that Facebook caters to the people who are taking and posting those images which do not impress Tom, I can see the merit in reminding someone when their password was changed and then for whatever reason they entered their earlier password.

And, in fact, in Tom's case this was useful. He did recall having changed his password several weeks before, but for whatever reason, he entered his earlier password. The alternative to having Facebook helpfully saying hey, "You Entered An Old Password" would be "Sorry, that password is incorrect." This would be more confusing than having Facebook recognize and helpfully report that the problem was the user's attempted use of an earlier password.

I don't know whether multiple people in a household routinely share a single Facebook account. But if so, one of them might have changed their shared password and failed to inform the others. So this would be a huge help in that instance.

The only problem I can see would arise if Facebook were to honor Tom's use of his retired password, but that would obviously never happen. So, I don't see any downside. And we know that those really annoying systems which require their users to periodically change their passwords for no reason, and then also refuse to allow any recently used password to be reused, are similarly storing previous password hashes. So the practice of remembering previous password hashes is not new. So I think this amounts to a useful and user-friendly feature. :-)

gimix³ / @gimix3

Hey Steve, I've been thinking about this thing that now we'll be able to choose our browser in iOS, and whilst I'm excited to be able to run Firefox in my iPhone, I'm feeling a bit uneasy. Safari, by being imposed on iOS and the default on macOS, has gained popularity over the years, and has been "too big to ignore" until now. Are we going back to the days of the hegemony of Chrome, and websites that can only be visited on Chrome?

I thought about this for some time, and I would say that it's really up to the other browsers. All of the standards that Chrome is using are open, open source, and available for adoption by anyone. It may indeed be that if they wish to retain what market share they can, they will need to adopt the same set of open standards that Chrome has. These next few years are really going to be interesting. The only place where Apple is being forced to allow 3rd-party browser engine cores to run is the EU. And we know that Apple is infuriated by this interference with their sovereignty over their own platform. So it seems unlikely that Apple will similarly be opening their devices to other browsers elsewhere.

Also, the Internet as a whole appears to finally be maturing, waking up and sobering up a bit. We're seeing things tightening up. Advertising is pulling back. Sites that never had a clear and justifiable reason for their own existence, yet were carrying a huge overhead with the plan to make it up in volume, are disappearing. What a shock. So in today's climate, I cannot see anyone wilfully turning away visitors who come surfing in from any platform. Perhaps internal corporate sites might force their employees to use some specific browser in order to run poorly designed software that won't run universally... and not as a general rule. No matter what happens on the platform side, especially with the web standardization process so well established today, I doubt we're going to see any public sites – none that plan to survive – telling their users that they must go get another browser.

Barbara / @Barbara130

It occurs to me that the 3rd Cisa recommendation might address universal Plug-and-play issues. If UPnP is on and malware tries to open ports the user would be notified, right?

So, Barbara is referring to CISA's 3rd recommendation which we discussed last week, about configuration changes requiring a manual event of some kind. And she raises a very good point about UPnP, which we know is a real security problem. But I'm afraid that's not what CISA was referring to and there's really no good way to deal with that problem. UPnP is so ubiquitous that all of today's routers have it enabled by default. Otherwise things break and since it's not the router's fault when UPnP is abused, there's no downside for the router to default to having it enabled as they all do. The last thing any router manufacturer wants is for some online reviewer to write that they swapped in this router and a bunch of things that were working broke.
The value of UPnP for providing hands-free connectivity is that is has and needs no management

interface. It's magic. Unfortunately, it's magic is black and it is certainly prone to abuse because it allows anything on the internal network, without any authentication barriers of any kind, to create static incoming port mappings to whatever devices are chosen. Because of UPnP's totally freewheeling nature – by design – there's no way to require any sort of manual intervention. Today's networked devices just expect to be able to come and go as they please.

Guillermo García / @gmogarciag

Hi Steve, listening to SN960 and your explanation on the reaction and workaround to Google's Protected Audience solution, I have two comments. If this registration requirement is widely adopted I'm wondering how that will affect the indexing spiders that index the web for us. And then, I wonder what kind of password reuse nightmare will emerge if a login is required for every web site on the web.

Those are two good points. And the second of those two questions occurred to me last week. If we're being asked to create what are essentially throwaway accounts just for the privilege of visiting websites, then why not use a throwaway password? Come up with something that probably meets modern password requirements and reuse it for sites that just don't matter. The problem, of course, is that there will probably be some tendency to keep using that password even on sites that are not throwaway... so this reuse for convenience is instilling a very bad habit. And this would also render our password manager's "web checkup" features useless since they would be freaking out about all of our deliberate password reuse.

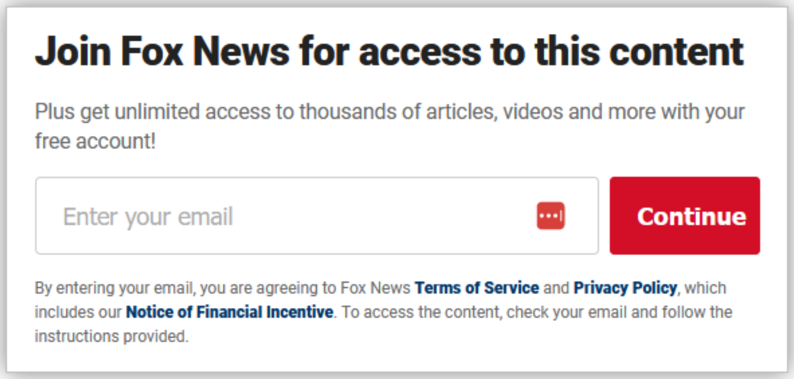
As for spiders, I hadn't considered that. And I wonder how that works today, since news sites behind paywalls appear to be indexed. One thought would be that the user-agent header which identifies a spider might be checked. But that would be easy for anyone to spoof. I suppose that the IP address blocks from which spiders crawl are likely well known and fixed – and that cannot be spoofed. So it would be possible to admit incoming requests from a set of previously well-known IP ranges without requiring a gratuitous login first. But having said that, there's really also no reason why spiders could not just login like everyone else. I'm sure that, assuming this comes to pass, the problem of keeping the web indexed will be solved.

Earl Rodd / @nc360370

Re: "Unintended Consequences" and websites requiring an "account" to view content. I first encountered this a few weeks ago and wondered why? It clearly was not a paywall. Now I understand why! In fact, no password is needed since it's not an "account" but merely a way to track me. They did verify that my email was a real one. So the "friction" for a user is minimal. Really nothing to remember except my "junk" email which I have for such purposes.

PS: The site was http://foxnews.com, one of the several entertainment sites I look at to see the going narratives related to the news.

It's very interesting that no password is needed. And Earl is correct, the only thing they really want and need is an eMail address. I went over to Fox News and poked around a bit and was not initially prompted for anything. I noticed in the URL bar that Firefox was saying that I had given the Fox site some special permissions. It turned out that I had disabled Autoplay and Audio was blocked. So I cleared any cookies that Firefox might have been carrying and then, sure enough I got the same thing Earl reported:



Join Fox News for access to this content

Plus get unlimited access to thousands of articles, videos and more with your free account!

Enter your email | **Continue**

By entering your email, you are agreeing to Fox News **Terms of Service** and **Privacy Policy**, which includes our **Notice of Financial Incentive**. To access the content, check your email and follow the instructions provided.

They do that page fade effect where you can see the top of the story, but it fades to white in this case and reads: *"Join Fox News for access to this content. Plus get unlimited access to thousands of articles, videos and more with your free account!"* Then underneath the eMail address field they add: *"By entering your email, you are agreeing to Fox News Terms of Service and Privacy Policy, which includes our Notice of Financial Incentive. To access the content, check your email and follow the instructions provided."*

I was curious about the "*Notice of Financial Incentive*" they referred to. So I followed the link which brought me to the following disclosure. Under Notice of Financial Incentive, it says:

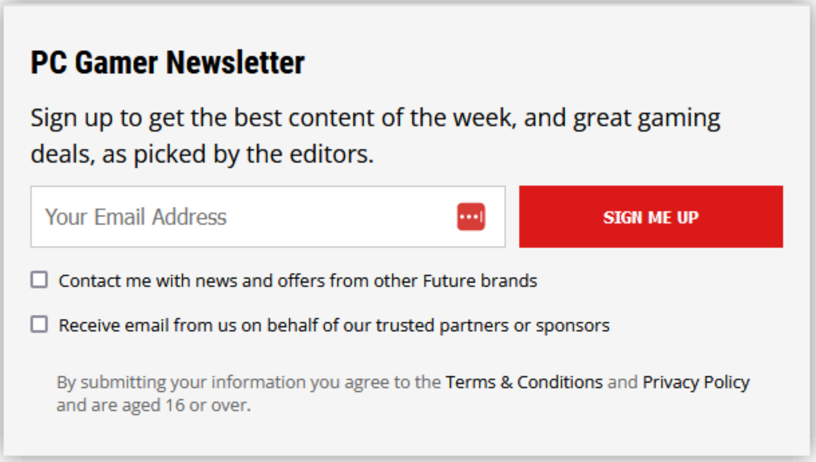
Notice of Financial Incentive. *This notice applies to our offers or programs (each an "Incentive Program") that link to this section of our privacy policy [and, of course, the page blocking eMail wall that brought me here linked to this] and which California may consider to be a financial incentive. You can opt-in to participate in an Incentive Program by providing your email address or other personal information. In exchange for providing your Personal Information and depending on the Incentive Program in which you participate, you may be able to access certain content, features, or events, receive a discounted price on an applicable subscription, or receive special news alerts or other entitlements. We will, in turn, use your Personal Information for the purposes set forth in this privacy policy, such as sending you alerts and marketing messages and personalizing your experience, including providing advertising you may find more relevant and interesting.*

To the extent we can estimate the value of your Personal Information to us, we consider the value of the offer (such as special content or features), the cost to us associated with providing the offer, and the potential benefit to us in the form of additional advertising or other revenue we may receive as a result of you using our Services. The value to us, if any, will depend on the extent to which you engage with our Services.

So it's very clear that this is exactly what Earl, who first encountered this, suggested it was. I don't visit the fox news site often enough to have appreciated this as a change of behavior, but apparently Earl does. As he put it "*I first encountered this a few weeks ago and wondered why?*". It's not obnoxious and the lack of any request for a password makes it much much less so.
So it looks like we have a perfect example of last week's topic; the unintended consequences of trying to take tracking away from an industry that doesn't want to let it go. Everyone who fills

out these new "Join our site" online forms, aside from subjecting themselves to an ever increasing torrent of spam, will be receiving a completely legal and legitimate 1st-party browser cookie to uniquely identify them to the site. So long as their browser returns that cookie during that and subsequent visits, they will be seen as a "member" of the site so they won't be bothered again. However, the site will, in turn, forward the visitor's eMail address to all partners, including all advertisers on that site, who will effectively pay for that information. Before I had switched away from the site, uBlock Origin's blocked access count was at 98 different domains.

And there's more evidence of this. As I was researching the TPM / Bitlocker decryption story, I scrolled down on the PCGamer.com site and encountered exactly the same thing:


PC Gamer Newsletter

Sign up to get the best content of the week, and great gaming deals, as picked by the editors.

Your Email Address | SIGN ME UP

☐ Contact me with news and offers from other Future brands

☐ Receive email from us on behalf of our trusted partners or sponsors

By submitting your information you agree to the Terms & Conditions and Privacy Policy and are aged 16 or over.

And, of course, the print at the bottom of the form says: *"By submitting your information you agree to the Terms & Conditions and Privacy Policy and are aged 16 or over."* In this case, my trusty uBlock Origin browser extension had blocked access 96 times and we all know what the fine print in the Privacy Policy is going to say. In this case the pop-up did not (yet) mask any of the site's content, but we can expect that behavior to become more widespread as the screws tighten over time. So it certainly does appear that we're all going to be asked to provide eMail addresses, and the annoyance of an eMail confirmation loop as an "Unforeseen Consequence" of Google's deliberate and enforced curtailment of traditional tracking techniques – now that they no longer need to do it.

One thing I didn't mention during our discussion of this last week, is that if anyone doesn't yet have a throwaway eMail account, now would certainly be a good time to establish one. No site to whom we provide this eMail address will be respecting our privacy. That's the entire point of obtaining our eMail address… it's so that our privacy can be more explicitly ignored than ever before **-and-** note that we are also implicitly agreeing to every such site's Privacy Policy which should be renamed their "Lack of Privacy Policy."

Tom Walker / @ttwiv

Hi Steve. Years ago you mentioned that you leave your phone plugged in all the time. Do you still do that? Just curious if, in your experience, that has kept the phone battery healthy?

I do keep my phones charged up all the time. I have an iPhone X that is stuck to an electromagnetic charging stand at either my day or evening location. Otherwise it's in my pocket when I'm out or between locations. But the moment I return home I walk right to the charger and it docks. Separately, I also keep an older iPhone 7 that my wife retired right here next to me as my desk phone. And it is never unplugged. It's essentially a corded phone. And I have three iPads which I use daily, each is always plugged in.

I can't claim to have any clear experimental evidence that this helps the batteries to live longer. The science all says that today's lithium-cycle batteries do not like to be deep discharged. But neither do they like to be overcharged. They much prefer to be kept nearer to their fully charged state. I assume that Apple understands all of this and is doubtless careful not to overcharge their devices.

One thing I can say is that my devices always outlive their batteries. So that's one data point. Another is that I have a friend of many years who used to allow his Apple devices to discharge fully before plugging them in. He was remembering the Nickle-Cadmium (NiCad) battery admonishments to always deep-discharge that type of battery chemistry in avoid the famous "memory effect" where NiCads that were only discharged a little before they were recharged would start thinking that they were empty at that point where they were recharged.

Anyway, he killed one Apple device after another until I noticed his red battery symbol and explained that plugging them in at every opportunity is the way to treat Lithium-cycle batteries.

Mark Jones / @mjphd

Hey Steve. I know you get no spam but would like your advice on email deliverability. I too am an old timer and maintained websites with email for decades. You haven't commented on how hard email deliverability is in the age of SPF, DKIM and DMARC. You also haven't offered advice about maintaining your own email server. February 2024 marks changes for how both Google and Yahoo regard appropriate settings. What is your take? Costs continue to escalate for services that interpret delivery failure events. EasyDMARC was free for multiple sites at one point. Now only free for one and paid plan is more than I pay per month for shared hosting. Is it time to give up running email off my own domains?

I have not commented upon the difficulty of eMail delivery in the age of SPF, DKIM and DMARC because I have not yet tried ramping up GRC's rate of eMail delivery. I do run my own eMail server and it fully supports all three sender verification standards. They are all configured and running, and GRC has been trickling eMail in and out of its domain for years with never a hint of any trouble. So there's some chance that I may have already established more of a positive reputation than I expected. It's not as if some never before seen domain were to start sending out bulk eMail. So I'll see how it goes. And I will absolutely 100% share everything that I encounter along the way!

Mark concluded his note with the question "*Is it time to give up running email off my own domains?*" I think that's a question that only he can answer. But from what he mentioned of escalating costs for EasyDMARC, for example, it doesn't sound as though he's running his own eMail server. So he's incurring additional service costs. I am running my own eMail server, so I have zero costs associated with hosting eMail environments.

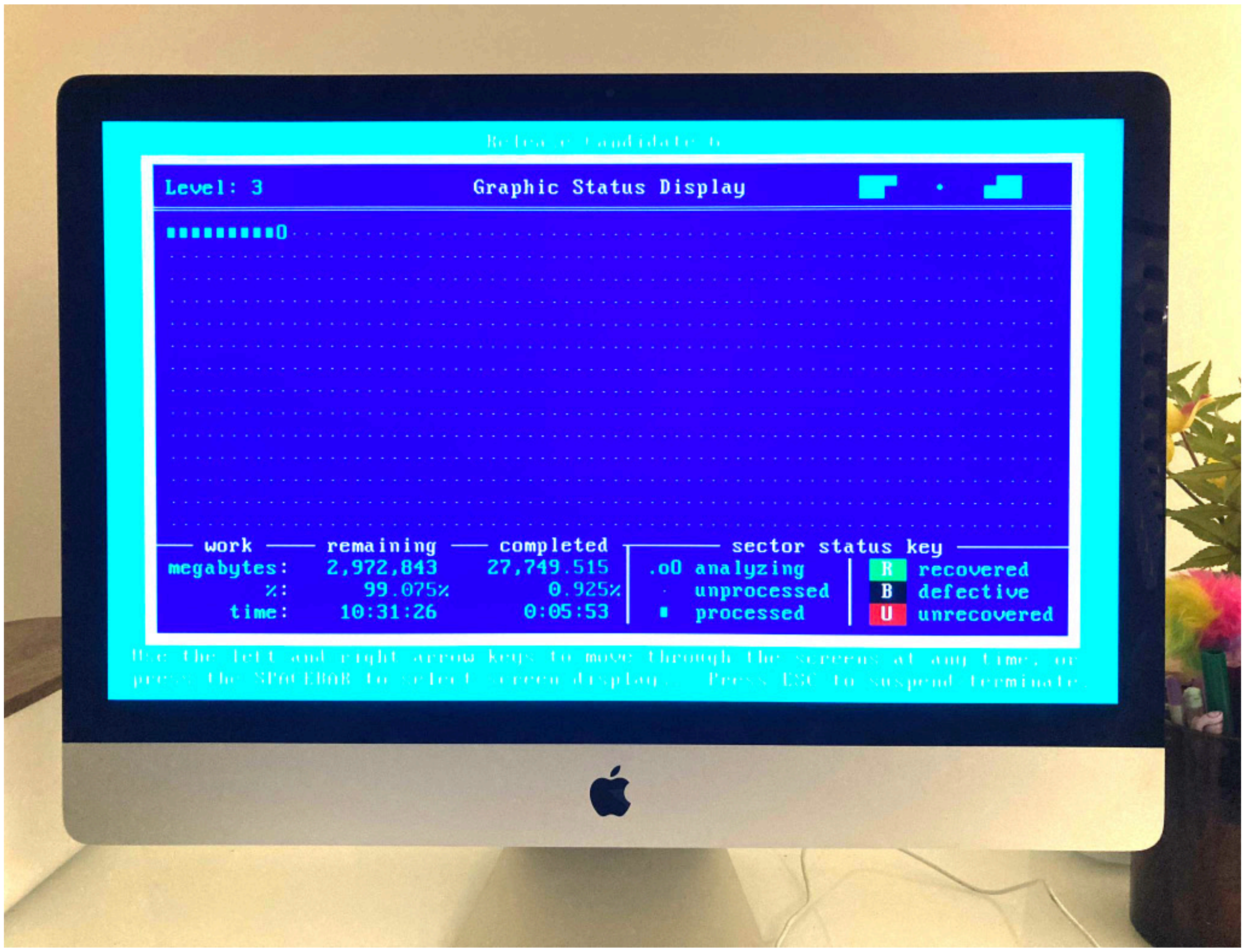
Max Feinleib / @MaxFeinleib

Thank you so much for sharing @abrenty's tip about checking iOS app sizes. I just deleted over 10 GB off my phone in what seems to be nothing but cruft.

Very nice!

Andre Arroyo / @andrearroyo

@SGgrc – Spinrite 6.1 RC6 running directly on my old iMac and booting off usb. I couldn't do this before. Now it's easy. Thanks for Spinrite and Security Now…



```
                         Release Candidate 6
Level: 3                  Graphic Status Display
■■■■■■■■■■0 . . . . . . . . . . . . . . . . . . . . . . . . . . . .

── work ──── remaining ── completed ┬──── sector status key ────
megabytes:   2,972,843   27,749.515 │ .oO analyzing    │ R recovered
       %:      99.075%       0.925% │  ·  unprocessed  │ B defective
    time:     10:31:26      0:05:53 │  ■  processed    │ U unrecovered
Use the left and right arrow keys to move through the screens at any time, or
press the SPACEBAR to select screen display.  Press ESC to suspend terminate.
```

SpinRite

Speaking of SpinRite, I am, as I had hoped, at work on SpinRite's documentation while SpinRite's paint continues to dry. One user in GRC's forums who had a dying SDHC SD card with a large non-critical file, wanted to experiment with its recovery. Part of what he wrote was:

Hi Steve - I'd like to know if there is a way to have SpinRite perform an operation like a level 2 scan multiple times. The reason I ask is that I have a Samsung 32 gb SDHC card that has a couple of spots it can't read or write to. I was able to copy all the files except one large one off it (an MP4 phone video I took that's not important) and I've decided to play with it to see if it's recoverable. The card passes the level 2 test but does not pass level 3 in two areas where I get a "Device Fault" error.

The really interesting thing about this is that in running level 2 a number of times, I've been able to "heal" some of the bad spots and increase the amount of the file being copied using Windows from 60% to 86%. My thought is, if I was able to have SpinRite do the level 2 scan overnight multiple times it might just heal any remaining bad spots.

Okay. So the first thing I explained in my reply was that SpinRite can now be completely controlled from its command line. So it's possible to start it with a command that will bypass any user interaction, select the proper drive and processing level, run SpinRite over the drive, then exit back to DOS once that's done.

Then it's a simple matter to create a DOS command script (which DOS refers to as a BATCH file) that jumps back up to loop to repeat that command over and over until it's interrupted.

The reason I'm mentioning this is that SpinRite's user can interrupt anything SpinRite is doing at any time. But if the user then manually exits to DOS in this situation, the batch file will still be in control and will immediately restart SpinRite. It would be possible to exit SpinRite, then frantically hit CTRL-C over and over to attempt to get DOS's attention and intervene in the looping. But that's inelegant.

When programs exit, a nearly universal convention is that they return an "exit code" to whatever invoked them. This code can signify whatever the program wishes, which is typically the program's success or failure. Today, SpinRite exits with a '0' exit code unless it's unable to parse its command line, in which case it returns a '1'. So what occurred to me while answering his question is that when it's exiting automatically due to the "exit" verb on its command line – and not because of a manual intervention – it could exit with an error code of '2'. This would allow for much more graceful "infinite loop" termination by using the DOS line: "if errorlevel 2 goto scan" at the bottom of the batch file.

Anyway, at some point when my eyes are crossing from writing documentation all day, I'll take a break from that to add this additional tiny convenience feature. And this is the great advantage of having some time to let the paint dry. There's still time for some minor touch ups, and history shows that once I release it as SpinRite v6.1 and have started working on its successor, I'm going to be extremely reluctant to mess with it any further. So now is a perfect time for those last tweaks.

Bitlocker: Chipped or Cracked?

Bitlocker's encryption cracked in minutes.
THE most sent to me news item of the past week was the revelation that PCs whose secret key storage Trusted Platform Module functions are provided by a separate TPM chip outside of the main CPU are vulnerable to compromise by someone with physical access to the machine.

This came as a surprise to many people who assumed that this would not be the case and that their mass storage systems were more protected by Microsoft's Bitlocker than they really were.

During system boot up, the small unencrypted startup portion of Windows sees that Bitlocker is enabled on the machine and that the system has a TPM chip which contains the key. So that pre-boot code says to the TPM chip "Hey there, I need the super-secret encryption key that you're holding" and the TPM chip replies "Yeah, no problem. Here it comes…" and then sends it to the processor. The only glitch here is that anyone with a hardware probe is able to connect the probe to the communicating pins of the processor or the TPM chip, or perhaps even to the traces on the printed circuit board which interconnect them, if those traces happen to be on the surface. Once connected, the computer can be booted and that entire happy conversation can be passively monitored. Neither end of the conversation will be any the wiser, and the probe is able to intercept and capture the TPM chip's reply to the processor's request for the Bitlocker decryption key.

These are the sorts of tricks that the NSA not only knows about but has doubtless taken advantage of countless times. But it's not made more widely obvious until a clever hacker like this "StackSmasher" guy comes along and shines a bright light on it.

The fundamental weakness in the design is that the TPM's key storage and the consumer of that stored key are located in separate components whose communication pins are readily accessible. And the obvious solution to this dilemma is to integrate the TPM's storage functions into the system's processor so that their highly sensitive communication remains inaccessible to casual eavesdropping… and, as it turns out, that's exactly that more recent Intel and AMD processors have done.

So this inherent vulnerability to physical attack occupies a window in time where discrete TPM modules exist (and are being overly depended upon) and before their functions have been integrated into the CPU. It's also not clear whether all future CPUs will always include a fully integrated TPM, or whether Intel and AMD will only do this for some higher-end models.

All this created such a stir in the industry that yesterday Ars Technica posted a nice piece about the whole issue. Under the sub-head "What PC's are affected?" Ars wrote:

BitLocker is a form of full-disk encryption that exists mostly to prevent someone who steals your laptop from taking the drive out, sticking it into another system, and accessing your data without requiring your account password. Many modern Windows 10 and 11 systems use BitLocker by default. When you sign into a Microsoft account in Windows 11 Home or Pro on a system with a TPM, your drive is typically encrypted automatically, and a recovery key is *uploaded to your Microsoft account. In a Windows 11 Pro system, you can turn on BitLocker manually whether you use a Microsoft account or not, backing up the recovery key any way you see fit.*

Regardless, a potential BitLocker exploit could affect the personal data on millions of machines. So how big of a deal is this new example of an old attack? For most individuals, the answer is probably "not very."

One barrier to entry for attackers is technical: Many modern systems use firmware TPM modules, or fTPMs, that are built directly into most processors. In cheaper machines, this can be a way to save on manufacturing—why buy a separate chip if you can just use a feature of the CPU you're already paying for? In other systems, including those that advertise compatibility with Microsoft's Pluton security processors, it's marketed as a security feature that specifically mitigates these kinds of so-called "sniffing" attacks.

That's because there is no external communication bus to sniff for an fTPM; it's integrated into the processor, so any communication between the TPM and the rest of the system also happens inside the processor. Virtually all self-built Windows 11-compatible desktops will use fTPMs, as will modern budget desktops and laptops. We checked four recent sub-$500 Intel and AMD laptops from Acer and Lenovo, and all used firmware TPMs; ditto for four self-built desktops with motherboards from Asus, Gigabyte, and ASRock.

Ironically, if you're using a high-end Windows laptop, your laptop is slightly more likely to be using a dedicated external TPM chip, which means you might be vulnerable.

The easiest way to tell what type of TPM you have is to go into the Windows Security center, go to the Device Security screen, and click Security Processor Details. If your TPM's manufacturer is listed as Intel (for Intel systems) or AMD (for AMD systems), you're most likely using your system's fTPM, and this exploit won't work on your system. The same goes for anything with Microsoft listed as the TPM manufacturer, which generally means the computer uses Pluton.

But if you see another manufacturer listed, you're probably using a dedicated TPM. I saw STMicroelectronics TPMs in a recent high-end Asus Zenbook, Dell XPS 13, and midrange Lenovo ThinkPad. StackSmashing also posted photos of a ThinkPad X1 Carbon Gen 11 with a hardware TPM and all the pins someone would need to try to nab the encryption key, as evidence that not all modern systems have switched over to fTPMs—admittedly something I had initially assumed, too. Laptops made before 2015 or 2016 are all virtually guaranteed to be using hardware TPMs when they have them.

That's not to say fTPMs are completely infallible. Some security researchers have been able to defeat the fTPMs in some of AMD's processors with "2–3 hours of physical access to the target device." Firmware TPMs just aren't susceptible to the kind of physical, Raspberry Pi-based attack that StackSmashing demonstrated.

There is some good news here, at least in the form of “what you can do if you really need and want the best possible protection” – It’s possible to add a PIN to the boot up process so that the additional factor of “something you know” can be used to strongly resist TPM-only attacks.

Microsoft provides a couple of **very good** and **extensive pages** which focus upon further hardening Bitlocker against attacks. I’ve included links to those articles (right there) in the show notes. But to give you a sense for the process of adding a PIN to your system right now, Ars explains under their sub-head “So what can you do about it?”:

> *Most individual users don't need to worry about this kind of attack; many consumer systems don't use dedicated TPM chips at all, and accessing your data requires a fairly skilled attacker who is very interested in pulling the data off of your specific PC rather than wiping it and reselling it or stripping it for parts. (This is not true of business users who deal with confidential information on their work laptops, but their IT departments hopefully do not need anyone to tell them that.)*
>
> *If you do want to give yourself an extra layer of protection, Microsoft recommends setting up an enhanced PIN that is required at startup, in addition to the theoretically sniffable key that the TPM provides. IT admins can enable this remotely via Group Policy; to enable it on your* ***own*** *system, open the Local Group Policy Editor (Windows + R, type gpedit.msc, hit Enter). Then navigate to Computer Configuration > Administrative Templates > Windows Components > Bitlocker Driver Encryption > Operating System Drives, and enable both the "require additional authentication at startup" and "allow enhanced PINs for startup" settings.*
>
> *Then open a Command Prompt window as an administrator and type* ***manage-bde -protectors -add c: -TPMAndPIN****; that command will immediately prompt you to set a PIN for the drive. Once you've done this, the next time you boot, the system will ask for a PIN before it boots into Windows.*
>
> *An attacker with physical access to your system and a sufficient amount of time may be able to gain access by brute-forcing this PIN, so it's important to make it complex, like any good password.*
>
> *A highly motivated, technically skilled attacker with extended physical access to your device may still be able to find a way around these safeguards. Regardless, having disk encryption enabled keeps your data safer than it would be with no encryption at all, and that will be enough to deter lesser-skilled, casual attackers from being able to get at your stuff.*

So, ultimately, we’re faced with the same tradeoff as always: convenience versus security. In the absence of a very strong PIN password, anyone using a system that is in any way able to decrypt itself without their assistance should recognize the inherent danger of that. If the system escapes their control, bad guys might be able to arrange to have the system do the same thing for them. Requiring “something you know” is the only true protection. At the same time, Bitlocking (Bitlockering?) a drive is certainly useful since it will very strongly prevent anyone who separates the drive from the machine from obtaining anything that’s protected that way.

So, Bitlocker: Yes. PIN: Yes. And as we’ve seen, it’s possible to add a PIN after the fact. And if your PIN is weak, you might want to strengthen it.