

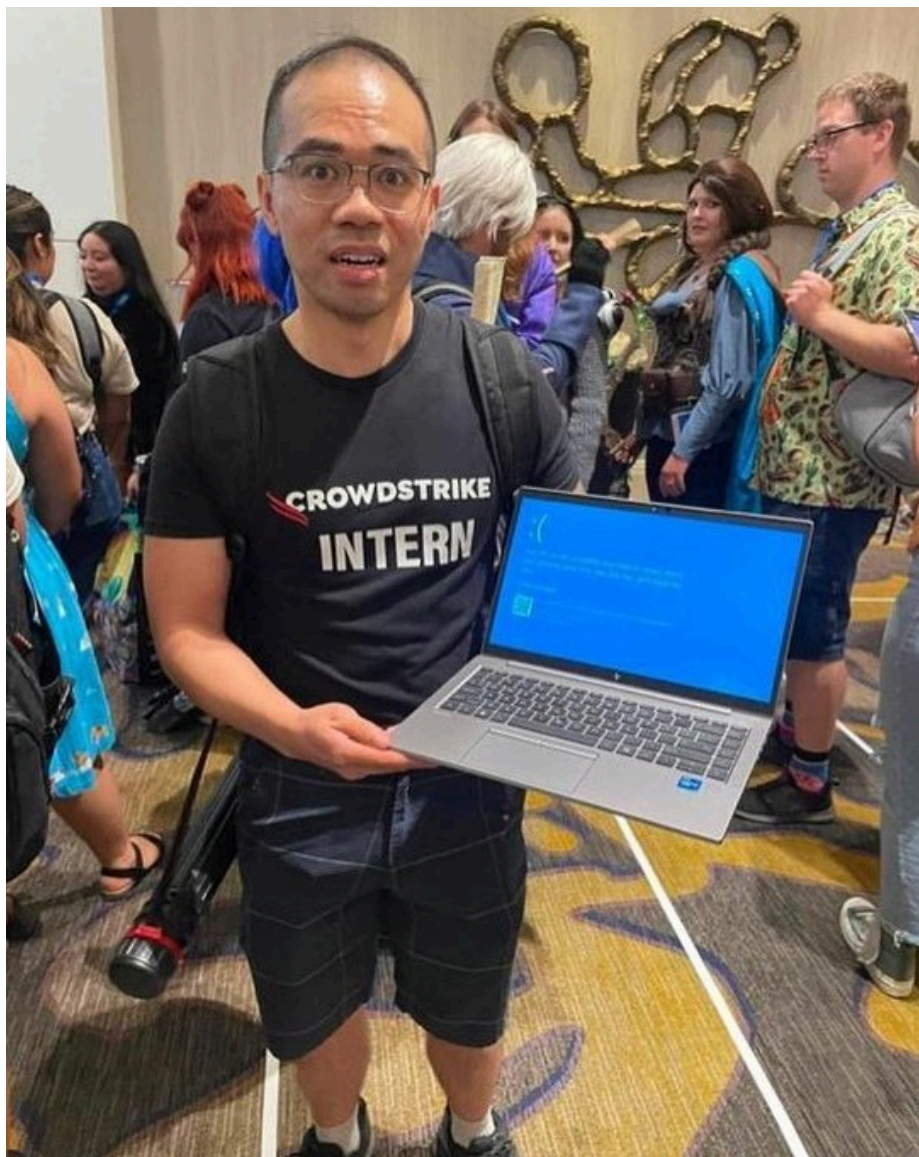
Security Now! #986 - 08-06-24

How Revoking!

This week on Security Now!

What's been learned over the past week about the PKfile Platform Key misuse issue? What is "IsBootSecure?" and why does that sound suspiciously like a new piece of GRC freeware? There's plenty of news on the 3rd-party cookie front. What's going on with Firefox and what position has the World Wide Web Consortium (W3C) taken on this important issue? Now that we're a few weeks downstream of the CrowdStrike disaster, the attorneys have come out to play. What are we learning about the legal side of this massive outage? What's been going on with GRC's incoming "SecurityNow" email system? And we finish by looking at DigiCert's recent mass certificate revocation event. Why it happened? What happened? Did it matter? Was it necessary? And how does it compare to Entrust's past behavior?

Is it too late to change my mind?



Security News

Before we get into the past week's news, we have a bunch of interesting follow-up news from last week.

Platform Key Disclosure

First off, last week's discussion of Binarly's discovery that by their estimate the manufacturers of around 1 in every 10 PCs, even those sold recently, never bothered to replace the "sample" and inherently insecure platform key which AMI clearly marked as DO NOT USE and DO NOT TRUST. Over the weekend one of the people in the newsgroup discovered that one of his recently purchased Dell PCs did indeed contain one of the insecure "DO NOT USE" platform keys.

Since last week's podcast I received a great deal of email from listeners who were unable to get the command-line command that Binarly had supplied, which I quoted in the show notes and mentioned last week, to work. When I tried it myself it was a true mess. For one thing, in order for any of the UEFI commands to work, an extra UEFI module must be installed. And then, since it needs to run a script, the normally protective powershell script blocker must be disabled by disabling some of powershell's security. And then it turns out that if SecureBoot is not enabled the command doesn't work, nor if the machine is booting from BIOS rather than UEFI. And to make matters worse, the Binarly command-line silently checks the output text from the command, which might well be complaints about any of the above, searching for a match on the "DO NOT USE" and "DO NOT SHIP" strings. As a result, this will tend to return massively false negative results since any output which is not the platform key's certificate will not contain those strings, even though the bogus AMI sample platform key might be sitting there on the motherboard. So, you all know where this is going, right? ...

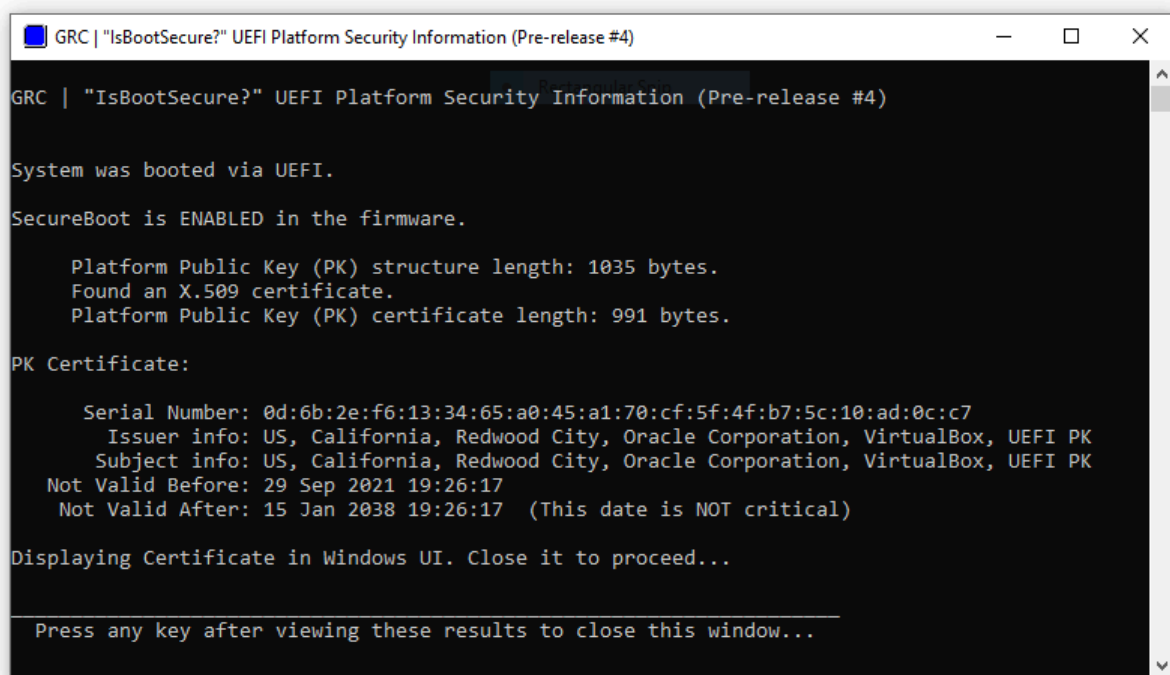
By Friday morning around 9am I had collected all of this feedback and tried using the command-line myself. So I thought "this needs another one of GRC's little Windows apps to let anyone quickly and easily get to the truth of what's going on."

I stopped working on SpinRite v6.1's documentation, which is my current focus since I need to get that behind me. I opened a new code project and began writing code to extract and analyze the motherboard's platform key.

While I was doing that and publishing incrementally more capable pre-releases for the gang there to test, we were also playing the name game, as we call it, over in GRC's SecurityNow newsgroup. That collective name brainstorming had previously resulted in the name "ValiDrive" and years before the "DCOMbobulator". So I was hopeful that someone (or ChatGPT) would come up with something terrific. But no really terrific name emerged. So this one won't have a flashy catchy name. It has the simple and straightforward name: "IsBootSecure?" It's listed under GRC's main menu under "Freeware" and "Security", its on GRC's master Freeware page, and its homepage is just grc.com/isbootsecure.htm

It's currently a very small, 21k Windows executable that can simply be run on any Windows machine and it will tell you exactly what's going on. It detects which firmware boot mode the system is in, either BIOS or UEFI. If UEFI, it determines whether the UEFI firmware is in SecureBoot mode or not and displays that. And if SecureBoot mode is enabled, it then extracts the Platform Key from the motherboard and displays the certificate's most useful information: It

shows the certificate's serial number, its Issuer name and Subject name information (which is where the "DO NOT TRUST" or "DO NOT SHIP" strings will be found), and it displays the certificate's Not Valid Before and Not Valid After dates – although those are non-critical for any long-lived self-signed certificate:



```
GRC | "IsBootSecure?" UEFI Platform Security Information (Pre-release #4)

System was booted via UEFI.

SecureBoot is ENABLED in the firmware.

Platform Public Key (PK) structure length: 1035 bytes.
Found an X.509 certificate.
Platform Public Key (PK) certificate length: 991 bytes.

PK Certificate:

Serial Number: 0d:6b:2e:f6:13:34:65:a0:45:a1:70:cf:5f:4f:b7:5c:10:ad:0c:c7
Issuer info: US, California, Redwood City, Oracle Corporation, VirtualBox, UEFI PK
Subject info: US, California, Redwood City, Oracle Corporation, VirtualBox, UEFI PK
Not Valid Before: 29 Sep 2021 19:26:17
Not Valid After: 15 Jan 2038 19:26:17 (This date is NOT critical)

Displaying Certificate in Windows UI. Close it to proceed...

Press any key after viewing these results to close this window...
```

And, as an extra bonus, it pops-up the standard Windows certificate dialog box so that all other details of the certificate can be examined. For a while over the weekend it was exporting the certificate, writing it to a file. But since it didn't make sense to have it always do that, I've turned that off for the moment.

What's available right now does all that, but it does it in a simple text window since what I have so far was meant as a tech development proof of concept. What I'm going to do now, is turn this into one of GRC's standard little Windows GUI apps. It'll interpret what's going on to display the important bits and draw a conclusion for its user. And if you wish to view the full certificate dialog for that certificate or to save it out as a file, there will be buttons for that.

Since this Platform Key mess is going to be a persistent and enduring concern, since there's no telling how long it's going to take to address these bad keys, if they ever are addressed, I felt as though that this was exactly the sort of freeware app that GRC should offer. It can be downloaded right not in its pre-release text-window incarnation and I'm sure I'll be announcing the final app on next week's podcast.

Okay. So much for PKfail.

Firefox's 3rd-party Cookie mess

Let's talk about Firefox's 3rd-party cookie mess and GRC's Cookie Forensics test. Two things are going on with Firefox and what GRC's cookie forensics page shows. Part of the problem is that Firefox is being a bit "foxy" and appears to not have updated its user interface after it integrated "Total Cookie Protection" into Firefox. There are no UI controls for it in evidence, anywhere. So

as many people have noted, the only way to make GRC's Cookie Forensics testing happy is to use the custom setting and really turn off 3rd-party cookies. Turning them off on other browsers simply works.

There are a couple of interacting things going on. Most important is that I never designed GRC's testing with an awareness of the inter-site 3rd-party cookie isolation that Mozilla has built into Firefox under the banner of "Total Cookie Protection." As I commented when the idea first surfaced, I think this is a really terrific idea. It potentially changes the game by "stovepiping" cookie collections into individual sites. As Mozilla described it, instead of browsers having one large cookie jar that's shared among all websites, which is the way it's already been and is the reason that tracking works, Firefox creates individual isolated min-cookie jars, one for every 1st-party domain that a browser visits. So an Ad-Tech company is welcome to place their 3rd-party cookie at site 'A'. But not that cookie will be tagged with a site 'A' context and it will ONLY be returned from site 'A'. So when a user visits another site where the same Ad-Tech company is also using cookies, that company will not receive the cookie it set over on site 'A' so it will not know that this visitor is the same one who was viewing ads over there. This is a huge change, and it's a big win.

But, I registered the independent domain "grctech.com" on New Years Day of 2002, because I needed to have an unaffiliated 3rd-party domain available – not a subdomain of GRC – around which I could build a Cookie Forensics system. And back then, there was no notion of the cookie site isolation that we have today with Firefox. As a consequence, Firefox may well be isolating cookies by domain, but my 22 year old test doesn't know about that. It's just seeing that Firefox appears to be very reluctant to fully disable 3rd-party cookies. It can be forced to, but it doesn't want to. And the reason Firefox can do this is because it has the site isolation trick up its sleeve.

What we're going to need in order to verify that site isolation is working and real, is an updated Cookie Forensics test that's aware of the possibility that while 3rd-party's cookies might be enabled, they might **not** be shared with **other** 1st-party sites. I don't recall why this was on my mind 12 days ago on July 25th, well before last week's podcast, but the question of the need to confirm the presence of inter-site cookie isolation must have occurred to me, since Hover shows that I registered another independent domain, "grctech.dev" on that Thursday. I needed a second unaffiliated domain to create a second 1st-party site to see whether it would be able to see the cookies that had been set by "grctech.com" when visitors were at "grc.com". So, at some point in the not too distant future, I plan to dust off the cookie forensics site and update it for awareness of today's cookie management. And sadly, with Google having lost its war on cookies, and with tracking being an issue that appears to be with us for now, having an updated facility for the characterization of browser cookie handling is as important today as it was back in 2002.

But there's more. It turns out that simply blocking all cross-domain 3rd-party cookies can break things. A more nuanced approach is needed to create a real-world solution that doesn't cause more harm than good. I want to share Mozilla's discussion of this, since it helps to demonstrate how complex the web has become, and also how dependent upon the details of its operation other services have become. As a result, changing anything can create unexpected problems.

Mozilla's explanation of this is titled "Third-Party Trackers", and they write:

Cookies are invisible pieces of data that a website can ask your browser to store on your device. The next time you visit the same website, it can ask the browser to read that cookie. That's how a website can "remember" things such as your preferences for that website and that you're logged on.

Another use for cookies is to transfer information from one website to another. For example, a sales website can store information about your purchase in cookies and redirect you to a payment or a review website. From the website's point of view, the cookies created by the sales website are called third-party cookies. There are also several Web libraries that developers use to add functionality to their websites. These libraries can set cookies on your device, too. If cookies are set by a library that is on a different domain from the website's domain, they are also third-party cookies.

Popular libraries are used by numerous websites. When you visit a website that uses a particular library, that library can set a cookie on your device. If you later visit another website that uses the same library, that library can read the cookie that was set when you visited the previous website. These third-party cookies, set and read by libraries from multiple websites, are called cross-site cookies.

There are two main reasons websites and libraries use cross-site cookies:

Cross-site tracking is by far the most common use of cross-site cookies. Trackers use cross-site cookies to collect information about the websites you visit and send them to other companies, often for advertising purposes. When you feel like an advertisement is following you around while you browse, this is a result of cross-site tracking. If the same tracker is present on multiple sites, it can build a more complete profile about you over time.

The other type of cross-site cookie is a functional cookie: Some websites rely on these cookies in order to function properly. For example, some websites may need access to cross-site cookies to let you use their service to sign in to another website (e.g., Facebook Login) or to process a payment for that website (e.g., Amazon Pay).

Firefox's Enhanced Tracking Protection blocks cookies from cross-site trackers and isolates cookies from all other third parties. This helps prevent your browsing activity on one website from being visible to other websites. Total Cookie Protection is now enabled by default. It is an advancement built into Enhanced Tracking Protection that works by maintaining a separate cookie jar for each website you visit.

While cross-site cookies from trackers are blocked in Firefox by default, a site may signal to the browser that it needs to use them for important functionality. In this case, Firefox will allow a third-party website to use cross-site cookies five times (or up to 1% of the number of unique sites you visit in a session, whichever is larger) without prompting you. After that, Firefox will prompt you to block these cookies. Without your consent, Firefox blocks these cookies from that point because a site requesting access that many times may be a tracker.

To interrupt for a moment, the first time I read that I had to go back to make sure I had read that correctly. This means that although Firefox now blocks cross site cookies by default, they are also providing websites with a mechanism to ask to please have cross-site cookies turned back on – but only just a little – presumably only enough to accomplish some specific task that

needs to communicate with cookies across sites. They understand that if it was just a blanket request websites would eventually all claim to need it all the time. Presumably they've done some testing of this and have decided that no website has a legitimate need for more than one or two so they added some margin and set the limit at five. They also look at the number of unique sites visited during a session of browser use, taking the larger of 5 or 1% of the unique site's count. That means that the breakeven point would be after visiting 500 different sites. Once you've visited another 100, 1% of 600 sites total is 6. I suppose this means that if someone is using their browser like crazy, the site they're visiting might need a bit more leeway. But a bit more is all they're getting.

If all of this sounds like a horrendous mess to you, then I'd say you've been paying attention, because it is a horrendous mess. This sort of heuristic is not the way the Internet and then the web was designed. The Internet is as robust as it is because it doesn't have any of this sort of nonsense. The Internet was designed with clear, clean and simple rules. But this "allow sites to request an exception, but not too many times" is the very definition of a kludge. I'm surprised they're not just tossing a coin. "Heads: you can use a cross-site cookie; Tails: Awww... Sorry about that, better luck next time." What a mess. They continue, by writing:

Third-parties will only be able to prompt you if you interact with the website you are on. For example, if you visit dogs.com and select the payment field, Amazon Pay cross-site cookies may be allowed to facilitate that transaction. After that, Firefox will ask you if you want to keep allowing them.

You can view the Permissions panel to see if a website has been allowed or denied permission to use cookies, by clicking the permissions icon in the address bar.

If you deny the request, the third-party will not be able to use cross-site cookies during that session. If you refresh or reload the page, the third-party may prompt you again.

From the Permissions panel for a site, you can click the X to revoke previously allowed access to cookies.

Hmmm. So website visitors, who have no idea what's going on behind the scenes, are being expected to make on-the-fly allow/deny decisions about whether or not the 3rd-party appearing at any given site should be allowed to use cross-site cookies. This is going to be interesting.

Firefox automatically allows third-party websites to use cross-site cookies on the first five or so websites you visit. For example, Amazon Pay would be able to use cookies on Old Navy, Blick, dog.com, and a handful of other sites without asking you for permission.

If a third-party continues to use cross-site cookies across multiple sites, this becomes a signal to Firefox that the third-party might be a tracker. At that point, the third-party would have to prompt you to ask for permission to use cross-site cookies.

There are other rules ("heuristics") that will make Firefox temporarily grant access to cross-site cookies to certain websites. These rules are designed to enable special use cases such as Single-Sign-On services and usually require some special interaction such as a top-level redirect or a user interaction, making it difficult for trackers to exploit them.

Wow. It's an even more tangled mess. Presumably these "heuristics" that Mozilla is talking about for Firefox are things like Firefox "knowing in advance" the domain names of large single sign-on providers so they automatically get special permission and can use cross-domain cookies without limit – because that's their legitimate business model. So that puts Mozilla in the position of deciding whose business model will and will not receive more permissive treatment under Firefox.

What's happening is that the industry is now tying itself in knots because cross-site information sharing we want and cross-site information sharing we don't want are using the same mechanisms.

Mozilla's Third-PartyTrackers explanation begs the question: In light of this, how could Chrome's Privacy Sandbox designers have possibly imagined the elimination of all cross-site 3rd-party cookies? Annoying though it may be to purists, Mozilla highlights some compelling use-cases which have arisen for needing cross-site information sharing in some form sometimes.

The W3C Finally Weighs-in

And amid all this, the World Wide Web Consortium, the W3C, has finally weighed-in on 3rd-party Cookies. Just to remind and update everyone on the W3C, Wikipedia's first paragraph says:

The World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web. Founded in 1994 and led by Tim Berners-Lee, the consortium is made up of member organizations that maintain full-time staff working together in the development of standards for the World Wide Web. As of 5 March 2023, the W3C had 462 members. The W3C also engages in education and outreach, develops software and serves as an open forum for discussion about the Web.

About a week and a half ago, on July 26th, the W3C's official statement was titled: "Third Party Cookies Must Be Removed." The article's Abstract says:

Third-party (AKA cross-site) cookies are harmful to the web, and must be removed from the web platform. This finding explains why they must be removed, and examines the challenges in removing them. We highlight some use cases that depend on third-party cookies and offer some examples of designed-for-purpose technologies that can replace them. Specification authors are expected to ensure they do not undermine the benefits of removing third-party cookies when proposing new web platform technologies.

And their introduction is interesting because they call-out Chrome. They write:

We consider privacy [to be] a core design principle and differentiator for the web platform. Many browsers have restricted third-party cookies (see: Webkit, Mozilla). Unfortunately, not all browsers have followed suit.

And they link to the article the Privacy Sandbox Group published 4 days before this which acknowledged that 3rd-party cookies would not be removed from Chrome.

TAG is the abbreviation for the W3C's Technical Architecture Group. So the W3C piece continues:

TAG calls for all browsers to drop support for third-party cookies, as this provides an opportunity to further improve the privacy preserving features of the web platform.

Removing third-party cookies from the web platform is not without complications. There are use cases for third-party cookies that need to be preserved, and pitfalls we need to be careful to avoid while doing so. This document sets out some aspects that specification editors and implementors should be aware of in order to make sure we ultimately leave the web better than we found it after third-party cookies are removed.

So this might be significant. If nothing else it likely provides some cover for any browsers that wish to move in this direction. There is now an official position from the Internet's major Web standards body unequivocally calling for the end of 3rd-party cookies.

Their "Leaving the web better than we found it" section says:

We support removing third-party cookies from the web platform, and we embrace the opportunity to improve the privacy features of the web. When we review new technologies to replace third-party cookies, we need to ensure that the replacements do not recreate the same pitfalls to privacy.

The TAG considers each new technology proposal both individually, and as they fit together with the web platform as a whole. The web must be cross-platform, so multi-implementer (multi-browser) support and developer support for privacy-related specifications is essential if they are going to achieve the goal of increasing privacy on the web. When we consider whether something makes the web platform better, we should be explicit about what the baseline for comparison is. Is a proposal better for privacy when compared to usage of third-party cookies? Or when compared with a web free from third-party cookies altogether? What about when some user agents restrict third-party cookies, but others do not?

We want to emphasize that as any replacement proposals progress, implementations should have a strong commitment toward, and reasonable time frame for, removing third-party cookies.

We are also wary of new mechanisms being introduced that could be abused together with cookies, fingerprinting surface or other tools, for greater privacy invasion. Given this context, we see an urgency to have a strict timeline for the removal of third-party cookies.

We are strongly in favor of innovations to build sustainable business models on the web platform, but an in-depth discussion of the various possibilities are outside of the scope of this document. From an architectural standpoint, web standards should avoid encoding particular business models that are available to authors, publishers, and web content creators.

In conclusion, when accommodating changes caused by the removal of third-party cookies, we should avoid introducing new technologies that, when deployed either individually or in combination, effectively preserve the status quo of harmful tracking and surveillance on the web.

I could certainly be reading too much into this. And I'm not nearly enough of an insider to be

able to venture a guess about what this might actually mean for changes in the future. But BOY does it sound good! They're using all of the right words. And this is laying out a template for the shape of the future approaches that would have a chance to succeed. They said: "*We are strongly in favor of innovations to build sustainable business models on the web platform*". We know what that means. That means that they, like Google, understand that advertising powers much of the Internet, and that targeting ads makes those ads enough more effective that it can double a site's revenue from advertising. At the same time, I lost count of the number of times the word "privacy" was used. So it's clearly not their intent for anything like the status quo to remain. But will anything change?

It's definitely worth repeating what I noted last week, which is that there are significant enterprises employing many people whose entire purpose is to deliberately and surreptitiously violate the privacy of everyone who ventures out onto the Internet with their computer. And these days who doesn't? And equally unfortunate is the fact that these enterprises have paying customers who have some use for the aggregated private data of individuals... which is to say, all of us. So this W3C action of very clearly working toward putting an end to 3rd-party cookies can be expected to see more opposition much as it appears happened with Google in Europe.

CrowdStrike Damages.

Predictably, the lawsuits and class actions have started up in the wake of CrowdStrike's extremely expensive outage. I think that some details of two of them would be of interest to our listeners.

The first is a shareholder lawsuit, blaming CrowdStrike for the entirely predictable significant drop in the company's stock price. Apparently the Plymouth County Retirement Association of Plymouth, Massachusetts now regrets having invested so heavily in CrowdStrike. The coverage of this after removing the redundant stuff we know, says:

Austin-based Cybersecurity firm CrowdStrike is facing a class action lawsuit from shareholders who claim the company defrauded them by concealing how its inadequate software testing could cause a global computer outage, resulting in a big hit to the share price and overall market value.

According to the July 30 complaint filed in the United States District Court in the Western District of Texas, CrowdStrike's Chief Executive George Kurtz characterized CrowdStrike's Falcon software as "validated, tested and certified" during a conference call on March 5.

The plaintiffs say these statements were "false and misleading" because CrowdStrike allegedly failed to properly test and update its Falcon software before rolling it out to customers.

The complaint alleges CrowdStrike "instituted deficient controls in its procedure for updating Falcon, and was not properly testing updates to Falcon before rolling them out to customers." CrowdStrike did not disclose that "this inadequate software testing created a substantial risk that an update to Falcon could cause major outages for a significant number of the Company's customers."

The complaint alleges: "Such outages could pose, and in fact ultimately created, substantial reputational harm and legal risk to CrowdStrike. As a result of these materially false and misleading statements and omissions, CrowdStrike stock traded at artificially high prices during the Class Period."

Huh? So they're upset over the terrific management, operation and reputation of CrowdStrike before the event, that had presumably made them so much money? And they're suing because what was once overly inflated is no longer. The article notes that:

Following the outage, the legal complaint says CrowdStrike's share price fell 32 percent over the next 12 days, wiping out \$25 billion of market value. As of July 31, CrowdStrike shares are worth \$231.96. They closed at \$343.05 on the day before the outage.

The lawsuit, led by the Plymouth County Retirement Association of Plymouth, Massachusetts, seeks unspecified damages for holders of CrowdStrike Class A shares between Nov. 29, 2023, and July 29, 2024. The class action also alleges that Delta Air Lines' hiring of an attorney to represent them in seeking damages from the company, along with Kurtz being called to testify before the U.S. Congress over the incident, caused CrowdStrike's share price to fall.

CrowdStrike said in a statement provided to media outlets: "We believe this case lacks merit and we will vigorously defend the company." Speaking at the time of the outage, CrowdStrike chief executive George Kurtz said: "We identified this very quickly and remediated the issue."

He added that its systems were constantly being updated to ward off "adversaries that are out there".

So that's the first of the entirely predictable legal actions that are underway. The second one surrounds the excessive degree of damage that the outage caused to Delta Airlines. As we know, whatever it was that happened to Delta kept many of their flights on the ground far longer than any of their other competitors who, like most of the rest of the planet, removed the bad file, rebooted and resumed operations. But not Delta.

Just yesterday, the publication "CIO Dive"s headline was: "CrowdStrike rebukes Delta's negligence claims in fiery letter" with the subhead: "After the airline said it was considering legal action, CrowdStrike said Delta's contract capped the cybersecurity provider's liability to "single-digit millions." The article explains:

CrowdStrike struck back forcefully against Delta Air Lines' claims of negligence and misconduct in a letter sent Sunday to the firm representing Delta, signed by attorney Michael Carlinsky. It's the latest in what has become a public dispute following recovery from the global CrowdStrike outage, which was caused by a faulty software update pushed to Windows servers on July 19.

Delta was the hardest hit major airline carrier — its disruption lasted longer and reached further than what United Airlines, American Airlines and others experienced. As the airline grappled with the scale and length of the outage, it moved to shift some of the blame publicly against the cybersecurity provider.

*Delta CEO Ed Bastian told CNBC last week the airline was considering legal action, seeking compensation for the **\$500 million in costs** the airline had endured. "We're looking to make certain that we get compensated, however they decide to, for what they cost us," Bastian said.*

CrowdStrike pushed the recovery responsibility back on Delta. The airline declined CrowdStrike's help with systems recovery according to the letter, which was shared with CIO Dive.

No one disputes CrowdStrike's ultimate culpability here. Even they don't. So what appears to be at issue is matters of degree. I found the text of CrowdStrike's Sunday letter to Delta. It says:

Dear David: I am writing on behalf of my client CrowdStrike, Inc. in response to your letter dated July 29, 2024, in which Delta Air Lines, Inc. raises issues and threatens CrowdStrike with legal claims related to the July 19, 2024 content configuration update impacting the Falcon sensor and the Windows Operating System (the "Channel File 291 incident").

CrowdStrike reiterates its apology to Delta, its employees, and its customers, and is empathetic to the circumstances they faced. However, CrowdStrike is highly disappointed by Delta's suggestion that CrowdStrike acted inappropriately and strongly rejects any allegation that it was grossly negligent or committed willful misconduct with respect to the Channel File 291 incident. Your suggestion that CrowdStrike failed to do testing and validation is contradicted by the very information on which you rely from CrowdStrike's Preliminary Post Incident Review.

CrowdStrike worked tirelessly to help its customers restore impacted systems and resume services to their customers. Within hours of the incident, CrowdStrike reached out to Delta to offer assistance and ensure Delta was aware of an available remediation. Additionally, CrowdStrike's CEO personally reached out to Delta's CEO to offer onsite assistance, but received no response. CrowdStrike followed up with Delta on the offer for onsite support and was told that the onsite resources were not needed. To this day, CrowdStrike continues to work closely and professionally with the Delta information security team.

Delta's public threat of litigation distracts from this work and has contributed to a misleading narrative that CrowdStrike is responsible for Delta's IT decisions and response to the outage. Should Delta pursue this path, Delta will have to explain to the public, its shareholders, and ultimately a jury why CrowdStrike took responsibility for its actions—swiftly, transparently, and constructively—while Delta did not. Among other things, Delta will need to explain:

- *Why Delta's competitors, facing similar challenges, all restored operations much faster.*
- *Why Delta turned down free onsite help from CrowdStrike professionals who assisted many other customers to restore operations much more quickly than Delta.*
- *That any liability by CrowdStrike is contractually capped at an amount in the single-digit millions.*
- *Every action, or failure to act, by Delta or its third-party service providers, related to the Channel File 291 incident.*
- *The design and operational resiliency capabilities of Delta's IT infrastructure, including decisions by Delta with respect to system upgrades, and all other contributory factors that relate in any way to the damage Delta allegedly suffered.*

In light of Delta's July 29 letter, CrowdStrike must also demand that Delta preserve all documents, records, and communications of any kind—including emails, text messages, and other communications—in the possession, custody, or control of Delta, its officers and directors, and employees concerning, but not limited to, the items listed below. As I am sure you can appreciate, while litigation would be unfortunate, CrowdStrike will respond aggressively, if forced to do so, in order to protect its shareholders, employees, and other stakeholders.

CrowdStrike's focus remains on its customers, including Delta. CrowdStrike hopes Delta reconsiders its approach and agrees to work cooperatively with CrowdStrike going forward, as the two sides historically have done.

Industry estimates are that the aggregate cost of the CrowdStrike outage to the Fortune 500 airlines will be around \$860 million so more than around \$143 million per airline. But it appears there's a contractual upper limit that does cap CrowdStrike's liability. And that's not in dispute.

The article then offers some interesting background about this, quoting Scott Bickley, the advisory practice lead at Info-Tech Research. Scott explained: "The standard limitation of liability (which, interestingly is known as the LOL clause) for most SaaS (software as a service) agreements caps liability at the actual funds spent on the subscription over a set period of time, usually the previous twelve months. Many enterprises will negotiate a multiple of this amount or a set capped amount." Bickley said that CrowdStrike's liability is likely to match annual spend or a multiple of annual spend if the clause was negotiated. He said: "Many large enterprises surprisingly do not negotiate these terms and default to the language in the vendors' agreements, which benefits the vendor," Bickley said. "Delta is likely going to pursue damages outside of the LOL cap and may rely on other legal arguments to bring the claim to a third-party dispute mediation or litigation."

So this is exactly what's expected. One last note is that, sadly, individual travelers who were inconvenienced by this are suing, even though travel delays and flight cancellations and rerouting is quite common. Yesterday Reuters reported:

Aug 5 (Reuters) - CrowdStrike's legal troubles from last month's massive global computer outage deepened Monday, as the cybersecurity company was sued by air travelers whose flights were delayed or canceled.

In a proposed class action filed in the Austin, Texas, federal court, three fliers blamed CrowdStrike's negligence in testing and deploying its software for the outage, which also disrupted banks, hospitals and emergency lines around the world.

The plaintiffs said that as fliers scrambled to get to their destinations, many spent hundreds of dollars on lodging, meals and alternative travel, while others missed work or suffered health problems from having to sleep on the airport floor.

They said CrowdStrike should pay compensatory and punitive damages to anyone whose flight was disrupted, after technology-related flight groundings for Southwest Airlines and other carriers in 2023 made the outage "entirely foreseeable."

So we know what effect that will have. After last years' troubles, I would be surprised to learn that the fine print in all passenger agreements doesn't already include a blanket liability waiver for any failure to fly caused by any foreseeable or unforeseeable events. And if it doesn't already it certainly will in the future. This is just the way the world is now. When you sign forms for a relatively minor procedure in a hospital the fine print explains that "We're really going to do the best we can to keep you alive, but, you know, stuff happens. So if something happens we'll be really sorry. Sign here and good luck." It's always a bit unnerving but what are you going to do?

Just as I was finishing up this podcast I received email from a listener, Rob Woodruff, informing me that as a CrowdStrike customer he had just received email from CrowdStrike's CEO and that their much anticipated "Root Cause Analysis" had been completed. I had no time to dig into it for today, and I don't plan to spend too much time on it next week since we've already given this ample time and coverage. But rest assured that I'll share anything new we may learn from it.

GRC's Email

Today's podcast filled-up before I could share any of the terrific feedback I've been receiving, in great volume, from our listeners. The incoming listener feedback system has been an utter success and I've been drowning in thoughts, notes, pointers to news and other great content. I just checked my "securitynow" inbox and we're at 1,512 pieces of email received.

I'm mentioning this because I'm torn by my inability to reply to this much email. Initially, I was trying to. But then I looked at how much time was going into creating replies and saying even not much more than "Thanks so much for sending that pointer." So I need to explain that everyone should know that even in the absence of any reply from me, if the email didn't bounce back to you, I have it, I've read it and I'm sure I appreciated it. Really. When I feel that I have to reply, I tend not to read since my seeing the feedback creates the feeling of an obligation to reply that I'm just not able to service. That's just the way I am.

So I want to sincerely thank everyone here, collectively, who has written and who will write. I really value this feedback and I am seeing what you have sent. You'll probably often sense its effect on the podcast afterwards. Bits of obscure news that I may have missed if you hadn't told me. So please do not ever imagine that because you don't receive a reply that means I never saw what you took the time to send. If you took the time to send it, I will have at least taken the time to read it, even if I was unable to say so explicitly. And, really, thanks!

One last point is that I've received a number of complaints from listeners who were forced to write to Sue or Greg at our Sales or Support email, respectively, because, they were unable to find the special "securitynow" email address anywhere on the site. The reason for that is, it's not meant for the general public and for GRC's non-podcast visitors. I want to keep this just for us. And the email address should not be too difficult to remember: Is "securitynow@grc.com". And to make it even easier, I've set the "reply-to" address of our weekly podcast mailings to that "securitynow@grc.com" address, so anyone who receives that email can simply reply to any weekly email received, and many of our listeners do just that.

How Revoking!

Just as last week's podcast was happening, my favorite certificate authority, DigiCert, announced their discovery of a mistake their systems had made during the process of Domain Control Validation or DCV. I talked about this last week with regard to Entrust, noting that SSL.COM, the Certificate Authority from whom Entrust plans to purchase interim certificates, will still need to have their own customers prove their control over their pending certificate's domain name. And I noted that there are various ways to get that done. The weakest is to use email received by and sent to the domain in question. Better ways involve adding DNS records to the domain which the CA can then remotely verify.

As we know, on the one hand we have Entrust who has for years effectively refused to revoke certificates which had been shown to be mistakenly issued to their customers. And as we'll see today, on the other hand we have DigiCert who takes the CA/Browser forum requirements to heart and immediately jumped into action when a truly insignificant mistake was discovered. But as this shows, a rule is a rule. Many of our listeners understood this difference and sent notes to me with variations of the Subject: "This is the way it's supposed to be done". Consequently, today's podcast is titled: "Here's how you revoke!"

The Hacker News explained the whole event, complete with updates as of Sunday. Their headline was: "*DigiCert to Revoke 83,000+ SSL Certificates Due to Domain Validation Oversight*" But just wait until you hear what it was. I've mixed their reporting with my own clarifications and additions. So here's that we have:

Certificate authority (CA) DigiCert has warned that it will be revoking a subset of SSL/TLS certificates within 24 hours due to an oversight in how it verified if a digital certificate is issued to the rightful owner of a domain. The company said it will be taking the step of revoking certificates that do not have proper Domain Control Validation (DCV).

DigiCert said: "Before issuing a certificate to a customer, DigiCert validates the customer's control or ownership over the domain name for which they are requesting a certificate using one of several methods approved by the CA/Browser Forum (CABF)."

One of the ways this is done hinges on the customer setting up a DNS CNAME record containing a random value provided to them by DigiCert. DigiCert then performs a DNS lookup for the domain in question to make sure that the random values are the same.

The provided CNAME record is for a subdomain of the user's domain and the random value from DigiCert is prefixed with an underscore character so as to prevent a possible collision with an actual randomly-named subdomain that might use the same random value. Note that domain names are not allowed to begin with an underscore, thus the collision prevention.

*What the Utah-based company found was that it had failed to include the underscore prefix with the random value used in **some** CNAME-based validation cases.*

Wow. Like... that could not **possibly** matter. Really. But in this CA space it needs to be that a rule is a rule, and we certainly beat up on Entrust over their flagrant violation of the rules.

DigiCert's mistake has its roots in a series of changes DigiCert enacted starting in 2019 to revamp their underlying architecture, as part of which the code to add an underscore prefix was removed and subsequently "added to some of the paths in the updated system" but not to one path that neither added it automatically nor checked if the random value had a pre-appended underscore. [Whoops.]

DigiCert said: "The omission of an automatic underscore prefix was not caught during the cross-functional team reviews that occurred before deployment of the updated system. While we had regression testing in place, those tests failed to alert us to the change in functionality because the regression tests were scoped to workflows and functionality instead of the content/structure of the random value. Unfortunately, no reviews were done to compare the legacy random value implementations with the random value implementations in the new system for every scenario. Had we conducted those evaluations, we would have learned earlier that the system was not automatically adding the underscore prefix to the random value where needed."

Subsequently, on June 11, 2024, DigiCert said it revamped the random value generation process, eliminating the manual addition of the underscore prefix within the confines of a user-experience enhancement project, but acknowledged it again failed to "compare this UX change against the underscore flow in the legacy system."

The company said it didn't discover the non-compliance issue until "several weeks ago" when an unnamed customer reached out regarding the random values used in validation, prompting a deeper review.

DigiCert noted that the incident impacts approximately 0.4% of the applicable domain validations, which, according to an update on the related Bugzilla report, affects 83,267 certificates issued across 6,807 customers.

Notified customers are recommended to replace their certificates as soon as possible by signing into their DigiCert accounts, generating a Certificate Signing Request (CSR), and reissuing them after passing DCV, Domain Control Validation.

The development prompted CISA to publish an alert, not over any danger that this incredibly minor and truly inconsequential mistake may have had, but stating that "revocation of these certificates may cause temporary disruptions to websites, services, and applications relying on these certificates for secure communication."

In a later update, DigiCert said: "DigiCert continues to actively engage with customers impacted by this incident and many of them have been able to replace their certificates. Some customers have applied for a delayed revocation due to exceptional circumstances and we are working with them on their individual situations. We are no longer accepting any applications for delayed revocation."

*These include customers who are operating critical infrastructure, who it said, "are not in a position to have all their certificates reissued and deployed in time without critical service interruptions." And DigiCert further noted that all impacted certificates, regardless of circumstances, **will** be revoked no later than August 3, 2024, 7:30 p.m. UTC.*

And true to their word, DigiCert subsequently confirmed on Sunday the 4th that *“all identified TLS certificates impacted by the CNAME-based Domain Validation incident were revoked” as of August 3, 2024, 7:30 p.m. UTC.*

So here we have an example of a company that is doing it right. Even in the face of the quite significant pain being caused to their own customers by the sudden and essentially emergency requirement to revoke their certificates.

And what’s more bracing is that there was nothing whatsoever wrong with those certificates – nor, actually, with the way their domains had been validated. This lack of a leading underscore in no way allows anyone to nefariously obtain a certificate fraudulently. It must still be the domain owner who places a CNAME record into their DNS having a long and random gibberish name. The fact that that long and random gibberish name doesn’t begin with an underscore has exactly ZERO impact upon the strength of the CNAME record’s subsequent validation. Really. Zero.

But, nevertheless, the mutually agreed upon specification clearly states that in the exercise of a ridiculous abundance of caution, and for the sake of making sure that there cannot possibly be a collision of the validating CNAME record’s name, with the name of a domain’s actual subdomain – like any domain would ever have an actual subdomain with a matching gibberish name – nevertheless, all DCV CNAME records **must** lead with an underscore character which is not, otherwise, a legal domain name.

So, yeah, it’s a worthwhile precaution to toss in. Why not? But the required response to the discovery of its absence really does seem way out of proportion to the actual impact of its absence – which wasn’t even negligible, it was zero.

So DigiCert took the hit. They informed 6,807 of their customers that due to a mistake at their end, they would be forcing the revocation of any of their mis-issued certificates even though there was nothing wrong with them – at all. It would have had to be galling for everyone involved. “We’ve got to do all of this – right now – for no real reason.” Yet it was done and if nothing else it will serve as a perfect counterpoint to Entrust who actively argued, while continuing to knowingly mis-issue certificates, that they really didn’t want to inconvenience their customers.

I’m out of time today. But I want to note that, as many of this podcast’s long-time listeners know, the topic of web server certificate revocation has long been another one of my hobby horses, much like 3rd-party cookies, because the entire web browser revocation system is a total and utter joke. It’s completely broken, it has always been broken and it’s never worked. Despite some half-hearted attempts to fix this inherently broken system, it never happened. Web browsers don’t know when the certificates they’re receiving from web servers have been revoked. If we have some time next week let’s update ourselves on in this space. In the meantime try aiming your browser at [“https://revoked.grc.com”](https://revoked.grc.com) – I bought that interesting and revealing test back to life last week in anticipation of this topic!

