Are Cybersecurity Problems a Manifestation of Technical Debt

Notes by J.H.Davenport

9 August 2022

1 Preamble

Thanks to FTAG colleagues who commented on the previous version (the rest of this paper). It seems that I haven't quite hit the mark here. An alternative approach might be on the lines of "if your MVP isn't secure by design, then don't extend it into the actual product". Does this make more sense?

ALS wrote this.

> I prefer that title, but I also think we should expand into the psychology of bad short-term decision making (I'm just going to base64 encode this password for now... after all it's just a demo environment for now).
>
> We should also acknowledge the reality that making a SaaS system secure to auditable standards can actually cost more than developing such a system in the first place. For example, most security standards will ask for cryptographic key rotation policies and procedures to be defined and implemented. Why do that if you only have 3 months of funding left before you have to shutdown your startup?

Mark Josephs wrote this.

> Yes, James, it might be better on those lines. The usual way to express this is that security needs to be built in, not bolted on. It is a point that people have been making for about 20 years; see the books, articles and talks of Gray McGraw, for example. Note that implementing security on the client (which can be bypassed) is one of the standard examples of a design flaw, as is failure to authenticate.
>
> Unfortunately, the article doesn't appear to be saying anything interesting about "technical debt".

Alan Brown:

> Anything i developed in a context, and as that context changes the debt occurs. There's a great tendency now just to release code. Which is fine if this code is throw-away.

Table 1: Top Routinely Exploited CVEs in 2020: [DHS21, Table 1].

Vendor	**CVE**	Type
Citrix	CVE-2019-19781	arbitrary code execution
Pulse	CVE 2019-11510	arbitrary file reading
Fortinet	CVE 2018-13379	path traversal
F5- Big IP	CVE 2020-5902	remote code execution (RCE)
MobileIron	CVE 2020-15505	RCE
Microsoft	CVE-2017-11882	RCE
Atlassian	CVE-2019-11580	RCE
Drupal	CVE-2018-7600	RCE
Telerik	CVE 2019-18935	RCE
Microsoft	CVE-2019-0604	RCE
Microsoft	CVE-2020-0787	elevation of privilege
Microsoft	CVE-2020-1472	elevation of privilege

4×2020; 4×2019; 2×2018; 2×2017.

2 Introduction

A recent security webinar [Mal22] claimed that the problem was the "Mountain of Technical Debt", or

> The thread that linked many of the most important cybersecurity events of 2021 was old, insecure code, but we can't patch our way to security.

This is not a new thought: see for example [GW13].

3 Third-Party Vulnerabilities

Much software that we have bought in, either as such or as part of a package such as a VPN system, contains vulnerabilities. Consider Table 1, and the warning that goes with it:

> Malicious cyber actors will most likely continue to use older known vulnerabilities, such as CVE-2017-11882 affecting Microsoft Office, as long as they remain effective and systems remain unpatched. Adversaries' use of known vulnerabilities complicates attribution, reduces costs, and minimizes risk because they are not investing in developing a zero-day exploit for their exclusive use, which they risk losing if it becomes known.

There were 20,175 new vulnerabilities published (i.e. allocated a CVE number) in 2021, and a total of 166,938 over the last ten years. For any reasonably large organisation, just keeping up with patches can be a major task.

It is tempting to think that things have improved, since the corresponding table ([DHS22, Table 1]) for 2021 shows 11×2021, 2×2020, 1×2019 and 1×2018. But in fact the 11×2021 are really only 6×2021, as ProxyLogon and ProxyShell were multiple CVEs. Sadly, CVE-2018-13379 (Path traversal in Fortinet software) and 2020-1472 (Microsoft Netlogon Remote Protocol elevation of privilege) appear in both lists.

4 Our Own Vulnerabilities

Even if we are not primarily a software house, we will often have in-house or bespoke external software. This too may well contain vulnerabilities.

5 The MVP trap

A standard definition of "Minimum Viable Product" is this [Bri16].

> Eric Ries, pioneer of the Lean Startup movement, describes a minimum viable product as: "[the] version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort".

More cynically (or possibly realistically) is this [Kam22]:

> an MVP is the smallest amount of work you can do to confirm or dispel your hypothesis.

That article goes on to quote Eric Ries (`https://techcrunch.com/2011/10/19/dropbox-minimal-viable-product/`) as describing Dropbox's initial MPV (a video) as the most powerful MVP: it persuaded funders to fund Dropbox.

Of course, the point is that these MVPs fufil the definitions above, but are *not* capable of being extended into true viable products, and hence are not viable products in the usual sense of the phrase.

6 Examples

Example 1 (Aethon Health Robots) There are apparently[1] thousands of these robots operating in hundreds of hospitals in North America, Europe and Asia. Cynerio found five serious issues with these, the most serious being CVE-2022-1070, a 9.8 vulnerability described as

> The product does not adequately verify the identity of actors at both ends of a communication channel, or does not adequately ensure the integrity of the channel, in a way that allows the channel to be accessed or influenced by an actor that is not an endpoint.

[1] Oddly, the newest item on the company's website is a holding statement (`https://aethon.com/aethon-statement-on-cyber-security-report/`) about this incident.

This means that "An unauthenticated attacker can connect to the TUG Home Base Server websocket to take control of TUG robots".

Cynerio describes the root cause as follows.

> The security components underpinning Aethon TUG devices were located in the JavaScript that was running in the browser of the user connected to their portal. This meant that all security measures in place for these devices could be bypassed, and that every action Cynerio researchers subsequently tested was not validated or checked by the system.

So what Aethon had was a viable product in the sense that it could impress potential purchasers, and would demonstrate apparent security features in that it would refuse naive unauthorised commands, but wasn't actually secure, and, given the description above, could not easily be made secure. Though it was stated that the vulnerabilities have been patched, the changes were likely more than "patches", and indede required firmware and operating system updates.

See the journalistic article in [Dav22], and the technical description in [Cyn22].

Example 2 (IoT Devices) [OER19] indicate that the design of IoT devices, especially those with a security function, may be sufficient for a minimal viable product, but be intrinsically insecure. Logically, and all to often in the implementation, the system is divided into "always-responsive" and the "on-demand" subsystems. By slectively suppressing (e.g. at the home router) the "on-demand" component, the user can be lulled into thinking that the device is working but has nothing to report.

Example 3 (Ever Surf web wallets) These wallets for the Everscale blockchain ecosystem were withdrawn (see [Gre22]) after Check Point research found a problem [Che22]. The "salt" on the web version was always '`sha256("unknown")`' and thus failed the fundamental requirement of salts.

Everscale noted that the web version of Ever Surf was "an experimental solution" that was helpful in the initial stages of the platform's development, i.e. an MVP. But it was a false MVP¡ in that it could not be extended into a product. Following [Che22], Everscale said "Our web version cannot provide a secure use of password-based KDF because of an inability to provide a unique salt such as device ID for that platform".

To make matters worse, "The company added that they don't know how many people use the web version so they are releasing information publicly to make sure no one's funds are at risk".

7 To be integrated

[Hug21, Con21, Nat16] [WAC10]

References

[Bri16] British Library. How to define a minimum viable product. https://www.bl.uk/business-and-ip-centre/articles/how-to-define-a-minimum-viable-product, 2016.

[Che22] Check Point (Alexey Bukhteyev). Check Point Research detects vulnerability in the Everscale blockchain wallet, preventing cryptocurrency theft. https://research.checkpoint.com/2022/check-point-research-detects-vulnerability-in-the-everscale-blockchain-wallet-preventing-cryptocurrency-theft/, 2022.

[Con21] Consortium for Information and Software Quality™ (CISQ™). The Cost of Poor Software Quality in the US: A 2020 Report. https://www.it-cisq.org/pdf/CPSQ-2020-report.pdf, 2021.

[Cyn22] Cynerio. JekyllBot:5 Command Center What You Need to Know about JekyllBot:5 Vulnerabilities for Aethon TUG Autonomous Robots. https://www.cynerio.com/jekyllbot-5-command-center, 2022.

[Dav22] J. Davis. 5 critical zero-days found in Aethon TUG smart robots used in global hospitals. https://www.scmagazine.com/analysis/device-security/5-critical-zero-days-found-in-aethon-tug-smart-robots-used-in-global-hospitals, 2022.

[DHS21] DHS Cybersecurity and Infrastructure Security Agency. Alert (AA21-209A) Top Routinely Exploited Vulnerabilities. https://www.cisa.gov/uscert/ncas/alerts/aa21-209a, 2021.

[DHS22] DHS Cybersecurity and Infrastructure Security Agency and peers. Alert (AA22-117A): 2021 Top Routinely Exploited Vulnerabilities. https://www.cisa.gov/uscert/ncas/alerts/aa22-117a, 2022.

[Gre22] J. Greig. Everscale blockchain wallet shutters web version after vulnerability found. https://therecord.media/everscale-blockchain-wallet-shutters-web-version-after-vulnerability-found/, 2022.

[GW13] D. Geer and C. Wysopal. For Good Measure: Security Debt. *;login:*, 38(4):62–64, 2013. URL: https://www.usenix.org/system/files/login/articles/13_geer_62-64_online.pdf.

[Hug21] O. Hughes. Developers: These botched software rollouts are costing businesses billions. https://www.techrepublic.com/article/developers-these-botched-software-rollouts-are-costing-businesses-billions/, 2021.

[Kam22] H.J. Kamps. Your MVP is neither minimal, viable nor a product. https://techcrunch.com/2022/05/03/your-mvp-is-not/, 2022.

[Mal22] Malwarebytes. Understanding the Malwarebytes 2022 Threat Review. Webinar April 12, 2022. URL: https://go.malwarebytes.com/EMEA_Threat-Intel-Webinar_0.3On-Demand.html.

[Nat16] National Institute for Standards and Technology (Paul E. Black and Lee Badger and Barbara Guttman and Elizabeth Fong). Dramatically Reducing Software Vulnerabilities. https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8151.pdf, 2016.

[OER19] T.J. O'Connor, W. Enck, and B. Reaves. Blinded and Confused: Uncovering Systemic Flaws in Device Telemetry for Smart-Home Internet of Things. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pages 140–150, 2019.

[WAC10] Jim Woodcock, Emine Gökçe Aydal, and Rod Chapman. The Tokeneer Experiments. In A.W. Roscoe, Cliff B. Jones, and Kenneth R. Wood, editors, *Reflections on the Work of C.A.R. Hoare*, pages 405–430, London, 2010. Springer. doi:10.1007/978-1-84882-912-1_17.