runZero

KEVology

An analysis of exploits, scores, & timelines on the CISA KEV

Executive summary

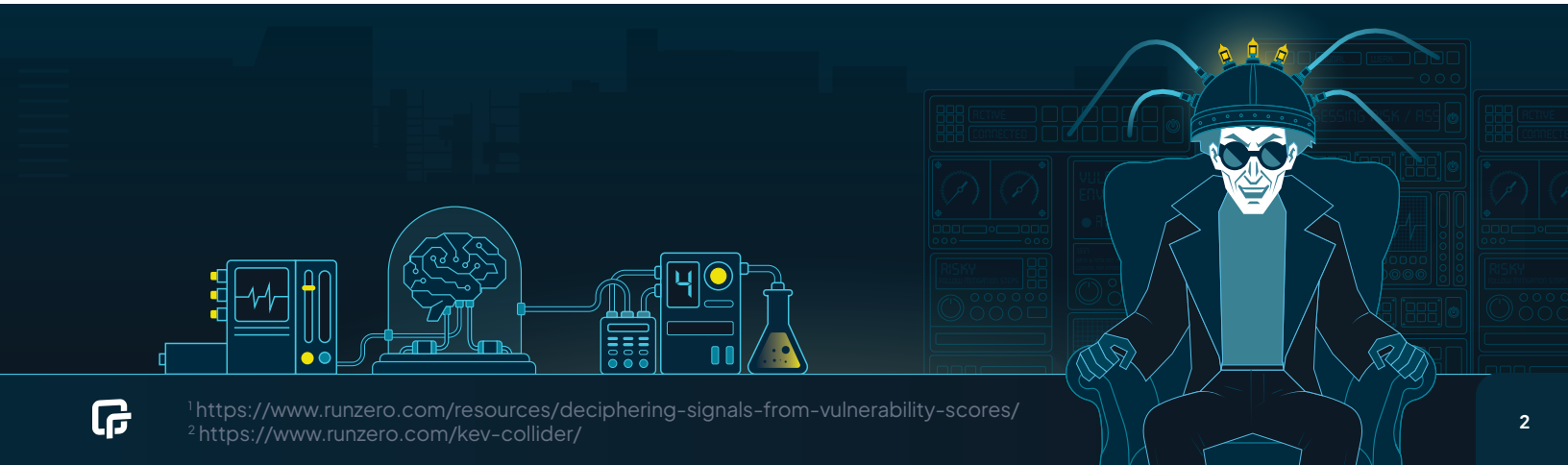
The CISA Known Exploited Vulnerabilities (KEV) catalog is often treated as a simple compliance punch list of vulnerabilities to fix, but that's not its purpose. This paper examines the KEV as an operational signal with the goal of helping cybersecurity practitioners make defensible prioritization decisions under real-world constraints. Written by Tod Beardsley, former CISA Section Chief for the KEV, this analysis draws on direct institutional experience and publicly available data to clarify what the KEV is designed to communicate, how it is produced, and how it should — and should not — be interpreted by IT defenders.

We evaluate a range of commonly used enrichment signals, including CVSS, EPSS, SSVC[1], as well as less-common signals such as public exploit tooling, MITRE ATT&CK mappings, and time-sequenced relationships, emphasizing that no single metric is sufficient on its own. Rather, value emerges from combining diverse, imperfect signals to reason about uncertainty, effort, and urgency as the KEV continues to grow in size, scope, and technological diversity.

In service of transparency, experimentation, and further analysis, the paper is accompanied by KEV Collider[2], a community-first web application and dataset that encourages readers to explore, recombine, and validate KEV enrichment data to better leverage the KEV in their daily operations.

Key contributions of this work include:

- **An insider's explanation of the KEV itself**, dispelling common myths, clarifying its inclusion criteria and limitations, and re-centering the reader on what the KEV does and does not signal.
    - **Key finding:** Only 32% of KEV-listed vulnerabilities are immediately exploitable for initial access.
- **A systematic exploration of enrichment signals**, showing how cybersecurity analysts can use public data sources such as CVSS, EPSS, exploit availability, and ATT&CK mappings in novel and practical ways to support risk-based decision-making.
    - **Key finding:** Commodity exploit tooling often predates KEV additions by days, weeks, and even years.
- **A multi-source view of vulnerability lifecycles**, demonstrating that by combining KEV, CVE, and exploit tooling timelines, defenders can develop an intuitive sense of what "normal" looks like for different classes of high-impact vulnerabilities, and recognize when a case deviates from those expectations.
    - **Key finding:** One vulnerability saw a CVE record, a KEV entry, and public commodity exploit tooling appear all on the same day.



[1] https://www.runzero.com/resources/deciphering-signals-from-vulnerability-scores/
[2] https://www.runzero.com/kev-collider/

What is the KEV?

The Known Exploited Vulnerabilities (KEV[3]) Catalog is a public list maintained by the U.S. Cybersecurity and Infrastructure Security Agency (CISA[4]), an agency within the United States Department of Homeland Security (DHS[5]). The catalog is in the public domain, is freely available worldwide, and is intended to guide cybersecurity prioritization by identifying specific vulnerabilities for which exploitation has been observed in the real world.

The KEV is often misunderstood as a government-curated list of the most severe vulnerabilities ever discovered, or as a catalog of hyper-critical remote code execution flaws actively being used by foreign adversaries against U.S. government systems. This casual interpretation is incorrect on several counts. While KEV-listed vulnerabilities do represent confirmed exploitation, the catalog exists primarily as an operational prioritization tool rather than as a comprehensive inventory of exploited vulnerabilities.

The KEV emerged following the release of Binding Operational Directive 22-01 (BOD 22-01[6]) in November, 2021. That directive requires U.S. Federal Civilian Executive Branch (FCEB) agencies to remediate vulnerabilities listed in the catalog within defined deadlines. The motivation was pragmatic: repeated, high-impact incidents demonstrated that public agencies and organizations were failing to patch well-known vulnerabilities, even after exploitation was widely understood.

Although BOD 22-01 is binding only on federal civilian agencies, the KEV itself is public and has since been widely adopted by state, local, tribal, and territorial governments (SLTTs), critical infrastructure (CI) operators, foreign governments, and private-sector defenders. In practice, the KEV has become a broadly reused signal for "things that matter right now," even outside its original regulatory scope.

When examined in isolation, KEV entries appear as a simple punch list. A patch for every vuln, a vuln for every enterprise. However, when KEV-listed vulnerabilities are analyzed in combination with other sources of vulnerability intelligence, richer patterns emerge. These patterns can inform more nuanced prioritization decisions, particularly when new vulnerabilities are added to the catalog. This form of analysis is useful well beyond federal environments and applies to enterprises across all sectors, regardless of their relationship with the U.S. government.

[3] https://www.cisa.gov/known-exploited-vulnerabilities-catalog
[4] https://cisa.gov/
[5] https://dhs.gov/
[6] https://www.cisa.gov/news-events/directives/bod-22-01-reducing-significant-risk-known-exploited-vulnerabilities

Inclusion criteria

According to BOD 22-01, inclusion in the KEV Catalog is gated by four explicit conditions, and all must be satisfied.

First, the vulnerability must have an assigned Common Vulnerabilities and Exposures (CVE) identifier. KEV entries are not free-form descriptions; they are anchored to CVEs so that downstream users can map them cleanly to vulnerability scanners, asset inventories, and patch management workflows.

Second, there must be a reasonable mitigation. In most cases this means a vendor patch or update. In some cases, configuration changes or compensating controls are acceptable. Occasionally, the only viable option is decommissioning affected software entirely. This means that vulnerabilities with no realistic path to mitigation will not reach the KEV. As an extreme example, an unpatched core Microsoft Windows service with no straightforward workaround won't end up being recommended by CISA to be decommissioned, since that would leave millions of federal users without a functioning workstation. (At the time of this writing, this is exactly the reason why CVE-2022-21894, aka BlackLotus, is still not on the KEV, though there is a comprehensive guide from the National Security Agency[7] (NSA) on how to mitigate it with some effort and expertise.)

Third, there must be evidence that the vulnerability is, or was, actually being exploited. This exploitation must be observed by CISA, either directly or through trusted reporting channels. CISA does not rely solely on public proof-of-concept code or theoretical exploitability. The bar is observed exploitation, not hypothetical risk.

Fourth, the vulnerability must be relevant to U.S. federal civilian interests. This criterion is broader than it may first appear. It includes not only federal systems themselves, but also technologies commonly incorporated by those systems, critical infrastructure, and, in some cases, infrastructure operated by key allied governments, SLTTs, or other special-interest organizations critical to the continued safety and security of the United States.

Information sources and reporting realities

Most KEV additions are informed by shared intelligence rather than unilateral observation. Information flows in from security vendors, incident responders, infrastructure operators, and other government entities, spanning dozens to hundreds of organizations. The catalog is a shining example of sustained, trust-based public-private collaboration.

Vendor behavior is uneven. Some vendors are quick to disclose and share details because they want customers to patch quickly and decisively, and view KEV as a strong motivator for this desirable behavior. Other vendors are less forthcoming, particularly when disclosure is perceived as a source of reputational risk. In rare cases, confidential government observations and reporting contribute to KEV decisions, even though the catalog itself is of course unclassified and public.

[7] https://media.defense.gov/2023/Jun/22/2003245723/-1/-1/0/CSI_BlackLotus_Mitigation_Guide.PDF

The KEV's operational constraints

The KEV is not a list of all exploited vulnerabilities. Many vulnerabilities are exploited in the wild but never appear in the catalog, and many exploits are developed and shared that are never reported as having been used in an offensive attack. A vulnerability can fail to meet the criteria for inclusion in a number of ways, but broadly, it's almost always because:

- The vulnerability lacks a CVE (often merely a delay in listing)
- No viable mitigation exists (common for very new vulnerabilities)
- Exploitation has not yet been corroborated by CISA analysts
- The affected software is not considered relevant to FCEB interests

As a result, the KEV should be understood as a deliberately constrained set of vulnerabilities that satisfy these specific operational criteria.

Non-CISA KEVs

As mentioned earlier, the CISA KEV is a free, public-domain, license-unencumbered, high-quality, low-noise resource for tracking exploited vulnerabilities that can surface in enterprise networks the world over. However, the CISA KEV is not the only list of known exploited vulnerabilities.

Notably, the private cybersecurity company VulnCheck produces the VulnCheck KEV[8], which is widely regarded as being more expansive than the CISA KEV. It often adds vulnerabilities to its KEV in closer-to-real-time as exploitation evidence surfaces, sometimes beating the CISA KEV as first to publish exploitation notifications. This is unsurprising, since VulnCheck's KEV has different operational goals than CISA's KEV. In brief:

- VulnCheck prioritizes speed of exploitation evidence[9] intelligence integration in order to detect malicious behavior, while CISA prioritizes speed of exploitation evidence intelligence integration in order to enable remediation of affected systems. Specifically, VulnCheck has no requirement for patch availability or other reasonable mitigation, while CISA does. VulnCheck also has no directive to be specifically relevant to U.S. federal agencies, which of course is unsurprising for a generally available product.

- VulnCheck KEV customers are free to choose to incorporate (or ignore) vulnerability intelligence in their cybersecurity defense programs. Conversely, CISA KEV-listed vulnerabilities **must** be actioned by federal agencies, as defined by CISA's statutory authorities[10]. In other words, every vulnerability listed on the CISA KEV is a million-dollar bug with regulatory knock-on effects, since inclusion triggers action among over 100 federal agencies[11], all of which fall under CISA's authorities. Customers of cybersecurity products rarely operate under such directives from their solution providers. Understandably, an accidental, or even questionable, inclusion of a vulnerability on the KEV can have serious ramifications for CISA's reliability and reputation, not to mention the cost to the American taxpayer of such an error.

[8] https://www.vulncheck.com/kev
[9] https://www.vulncheck.com/blog/state-of-exploitation-2026
[10] https://uscode.house.gov/view.xhtml?req=(title:6%20section:652)
[11] https://www.cisa.gov/news-events/directives/federal-civilian-executive-branch-agencies-list

Throughout this paper, when we reference "the KEV," we mean "the KEV operated by CISA." Since the methodologies and strategies explored in this paper are likely generalizable to most exploited vulnerabilities, though, future work in this area may extend to VulnCheck KEV and other lists of known exploited vulnerabilities.

Finally, this paper should not be construed as endorsing the CISA KEV over the VulnCheck KEV, or vice versa, for any commercial purpose. Both KEVs have their own uses, strengths, audiences, and limitations, and comparing them as competitive or somehow "better" or "worse" tends to misunderstand the purposes of both.

What kinds of vulnerabilities are KEV-listed?

As stated earlier, one of the most common misconceptions of the KEV is that it lists the worst of the worst vulnerabilities ever known. Oh, if only that were the case, for the KEV would be much, much shorter!

When delving even shallowly into the KEV, a casual observer will notice that the vulnerabilities there are not all unauthenticated, remotely exploitable, initial intrusion vulnerabilities. Take, for example, the last dozen vulnerabilities added to the KEV in December, 2025:

CVE ID	Vulnerability Name	CWE	Date Added	Due Date
CVE-2025-14847	MongoDB and MongoDB Server Improper Handling of Length Parameter Inconsistency	CWE-130	2025-12-29	2026-01-19
CVE-2023-52163	Digiever DS-2105 Pro Missing Authorization	CWE-862	2025-12-22	2026-01-12
CVE-2025-14733	WatchGuard Firebox Out of Bounds Write	CWE-787	2025-12-19	2025-12-26
CVE-2025-20393	Cisco Multiple Products Improper Input Validation	CWE-20	2025-12-17	2025-12-24
CVE-2025-40602	SonicWall SMA1000 Missing Authorization	CWE-250	2025-12-17	2025-12-24
CVE-2025-59374	ASUS Live Update Embedded Malicious Code	CWE-506	2025-12-17	2026-01-07
CVE-2025-59718	Fortinet Multiple Products Improper Verification of Cryptographic Signature	CWE-347	2025-12-16	2025-12-23
CVE-2025-14611	Gladinet CentreStack and Triofox Hard Coded Cryptographic	CWE-798	2025-12-15	2026-01-05
CVE-2025-43529	Apple Multiple Products Use-After-Free WebKit	CWE-416	2025-12-15	2026-01-05
CVE-2018-4063	Sierra Wireless AirLink ALEOS Unrestricted Upload of File with Dangerous Type	CWE-434	2025-12-12	2026-01-02
CVE-2025-14174	Google Chromium Out of Bounds Memory Access	CWE-119	2025-12-12	2026-01-02
CVE-2025-6218	RARLAB WinRAR Path Traversal	CWE-22	2025-12-09	2025-12-30

Table 1: The last twelve vulnerabilities of 2025 added to the KEV. The Twelve Days of KEVmas, as it were.

Just eyeballing this list reveals at least a couple that aren't very good for initial access. The Google Chromium bug, CVE-2025-14174, is clearly a browser bug. That means the attacker would probably need to trick someone into visiting a malicious website, or insert malvertising Javascript into the target's favorite subreddit, or do something else to set the trap and wait for the user to trigger the exploit. Apple WebKit bugs tend to (but not always!) fall into the same requires-user-interaction category. While initial access could be gained from such attacks, these attacks cannot be launched at will by attackers targeting specific victims. They require patience, some luck, and importantly, some victim action.

SonicWall and Fortinet bugs often indicate high criticality, since they tend to be found on those edge devices that govern access to the internal network. But without more context, it's hard to say what any of these vulnerabilities actually mean for a given enterprise. With just a little cross-referencing against readily available data, though, we can find out a lot more about how worried we should be about these vulnerabilities, as shown in the following table and discussed below.

One handy shorthand for quickly evaluating CVEs by CVSS is the notion of "straight shot RCE," where "RCE" is common hacker jargon for "remote code execution." The criteria for this special combination of CVSS vectors are:

- Access Vector of "Network" (as opposed to "Adjacent," "Local," or "Physical")
- Privileges Required of "None" (as opposed to "Low" or "High")
- User Interaction of "None" (as opposed to "Required")
- Integrity Impact of "High" (as opposed to "None" or "Low")

If all of these are true, we can consider a vulnerability as a straight shot RCE bug, regardless of all the other CVSS vectors. These are the vulnerabilities that listen on an internet socket, don't require a login, don't require the victim to act, and the attacker ends up with total control over the affected system.

CVE ID	Vendor	CVSS Vector String	CVSS Severity	Straight Shot RCE
CVE-2025-14847	MongoDB	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N	HIGH	FALSE
CVE-2023-52163	Digiever	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	HIGH	FALSE
CVE-2025-14733	WatchGuard	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	CRITICAL	TRUE
CVE-2025-20393	Cisco	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H	CRITICAL	TRUE
CVE-2025-40602	SonicWall	CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:U/C:H/I:H/A:H	MEDIUM	FALSE
CVE-2025-59374	ASUS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	CRITICAL	TRUE
CVE-2025-59718	Fortinet	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	CRITICAL	TRUE
CVE-2025-14611	Gladinet	CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:L/I:N/A:N	MEDIUM	FALSE
CVE-2025-43529	Apple	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	HIGH	FALSE
CVE-2018-4063	Sierra Wireless	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	HIGH	FALSE
CVE-2025-14174	Google	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	HIGH	FALSE
CVE-2025-6218	RARLAB	CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	HIGH	FALSE

Table 2: Cross-referenced CVSS vectors and severities among the last 12 KEVs of 2025.

Applying this rubric to the last twelve vulnerabilities added to the KEV in 2025, what immediately jumps out from this view of the KEV is that only four of these twelve recent vulnerabilities (33%) appear to be useful for attackers to gain immediate, remote initial access to an enterprise network.

One final glance at the reported CVSS-based severity ratings shows that while the common severity estimation of these bugs tend toward the high side, not all are coming in at hair-on-fire levels. We can see that of the twelve, the four we found to be straight-shot RCE are indeed Critical, while the remaining are High and Medium.

So, it's not just initial access vulnerabilities that get KEV attention. Among all 1,488 catalogued KEVs as of January 14, 2026, **only 483, or 32%, are useful for immediate initial access**, based on this CVSS filter. This leaves over a thousand KEVs for other activities that criminals and spies exploit vulnerabilities for.

Finally, not all KEV-listed bugs affect only one product, or even one class of product. Some vulnerabilities listed on the KEV that affect fundamental libraries, components, and protocols that are present in many hundreds to thousands of products are particularly notorious for their resistance to easy operational identification by vulnerability management software. They include:

- CVE-2021-44228, Apache Log4j2 Remote Code Execution (aka Log4Shell)
- CVE-2023-44487, HTTP/2 Rapid Reset Attack
- CVE-2014-0160, OpenSSL Information Disclosure (aka Heartbleed)

Log4Shell and Heartbleed affect very common software components that, respectively, provide logging services for Java applications, and cryptographic services for basically everything. The HTTP/2 Rapid Reset vulnerability affects web servers that implement HTTP/2 (which is every modern production web server).

These three vulnerabilities are likely to persist for years and years in the wild, since identification and remediation with today's tooling is spotty at best, and no amount of federally mandated deadlines is going to change this fact. Heartbleed, today, can be detected fairly trivially, but does tend to require active exploitation to do so. Log4Shell can be detected similarly, but exploitation evidence often pops up on systems different from the initial target (this is a classical example of the "Scope change" vector discussed in CVSS). Rapid Reset can be detected and demonstrated with some standard DDoS techniques, but these are beyond the scope of authority for typical IT security professionals.

All of these **are** detectable through carefully maintained Software Bills of Materials (SBOMs) or other intrusive, on-system endpoint detection and response (EDR) solutions. Without some unusually tight controls on client systems, or very careful active exploitation, these vulnerabilities will escape the notice of auditors and assessors alike. The silver lining, of course, is that attackers face the same challenges in targeting. But this silver lining is faint, for attackers often have the luxury of "spray-and-pray" techniques to see which, if any, exploits actually produce results, especially in networks lacking intrusion and anomaly detection controls.

So, where does this brief jaunt into correlating KEV data with other data sources take us? It is clear that, unless you're responsible for a federal agency and are aiming to hit those deadlines mandated by BOD 22-01, you should not treat all KEV vulnerabilities at the same level of urgency. Of course, just writing that down doesn't do the lone IT security professional much good, so the remainder of this paper will explore some methodologies and techniques that can make sense of your own KEV-listed exposures. It also introduces some free online tooling by runZero that makes this process more natural, efficient, and dare I say, more fun.

Our approach to enriching KEV data

There are several rich, authoritative sources of vulnerability intelligence, also generally free to use, that can inform priorities and strategies for detecting and remediating vulnerabilities. We can take a look at not only the KEV record itself, but also their associated CVE records, CVSS, Stakeholder-Specific Vulnerability Categorization (SSVC), and Exploit Prediction Scoring System (EPSS) scores, Common Weakness Enumeration (CWE) identifiers, and the MITRE ATT&CK framework mappings provided by the Center for Threat-Informed Defense (CTID). On the darker side, we can also inspect working exploits provided by the open source community projects Metasploit Framework[12] and ProjectDiscovery Nuclei[13].

Rather than exhaustively detail every data point and the whys and wherefores, we'll take a quick tour through each of these data sources and document some immediately interesting findings. Of course, if you're the sort who wants to do their own research, I invite you to skip directly to the Methodology appendix at the end of this paper, or swing by KEV Collider[14], hosted by runZero, where you can mix and match your favorite criteria to produce an actionable, prioritized list of vulnerability management starting points. The data driving KEV Collider is sourced directly from this paper, and resides in an issues-tracked GitHub repository, so if you have ideas for further investigations and data sources, pull requests are most certainly accepted!

KEV entries

KEV entries are structured to be fairly sparse, getting directly to the what and the when of vulnerability management across the U.S. federal government. A typical JSON object at CISA's GitHub repository, https://github.com/cisagov/kev-data, looks like this:


```
{
     "cveID": "CVE-2025-14847",
     "vendorProject": "MongoDB",
     "product": "MongoDB and MongoDB Server",
     "vulnerabilityName": "MongoDB and MongoDB Server Improper Handling of Length Parameter
Inconsistency Vulnerability",
     "dateAdded": "2025-12-29",
     "shortDescription": "MongoDB Server contains an improper handling of length parameter
inconsistency vulnerability in Zlib compressed protocol headers. This vulnerability may
allow a read of uninitialized heap memory by an unauthenticated client.",
     "requiredAction": "Apply mitigations per vendor instructions, follow applicable BOD
22-01 guidance for cloud services, or discontinue use of the product if mitigations are
unavailable.",
     "dueDate": "2026-01-19",
     "knownRansomwareCampaignUse": "Unknown",
     "notes": "This vulnerability could affect an open-source component, third-party
library, protocol, or proprietary implementation that could be used by different products.
For more information, please see: https:\/\/jira.mongodb.org\/browse\/SERVER-115508 ;
https:\/\/nvd.nist.gov\/vuln\/detail\/CVE-2025-14847",
     "cwes": [
         "CWE-130"
     ]
```


[12] https://github.com/rapid7/metasploit-framework
[13] https://github.com/projectdiscovery/nuclei
[14] https://www.runzero.com/kev-collider/

About 12 lines long, we can easily pick out the vendor and the product, an authoritative reference or two, and a reminder to fix this bug per BOD 22-01. Unfortunately, sometimes the product is "multiple products." Nary a Common Platform Enumeration (CPE) string or Package URLs (PURL) are to be found, at least not here. That said, there is more we can glean from KEV by analyzing data on timings and ransomware vs. non-ransomware.

Time-based metrics

Glenn Thorpe of GreyNoise Intelligence presented many interesting findings based almost entirely on KEV-supplied data at BSides Las Vegas 2024[15] and an accompanying blog post, Unveiling Vulnerability Insights from the CISA KEV Catalog[16]. So let's replicate one of his findings, now a year later, focusing on the pair of markers, `dateAdded` and `dueDate`, and seeing just how often the standard, 21-day deadline is offered by CISA to its FCEB clients.

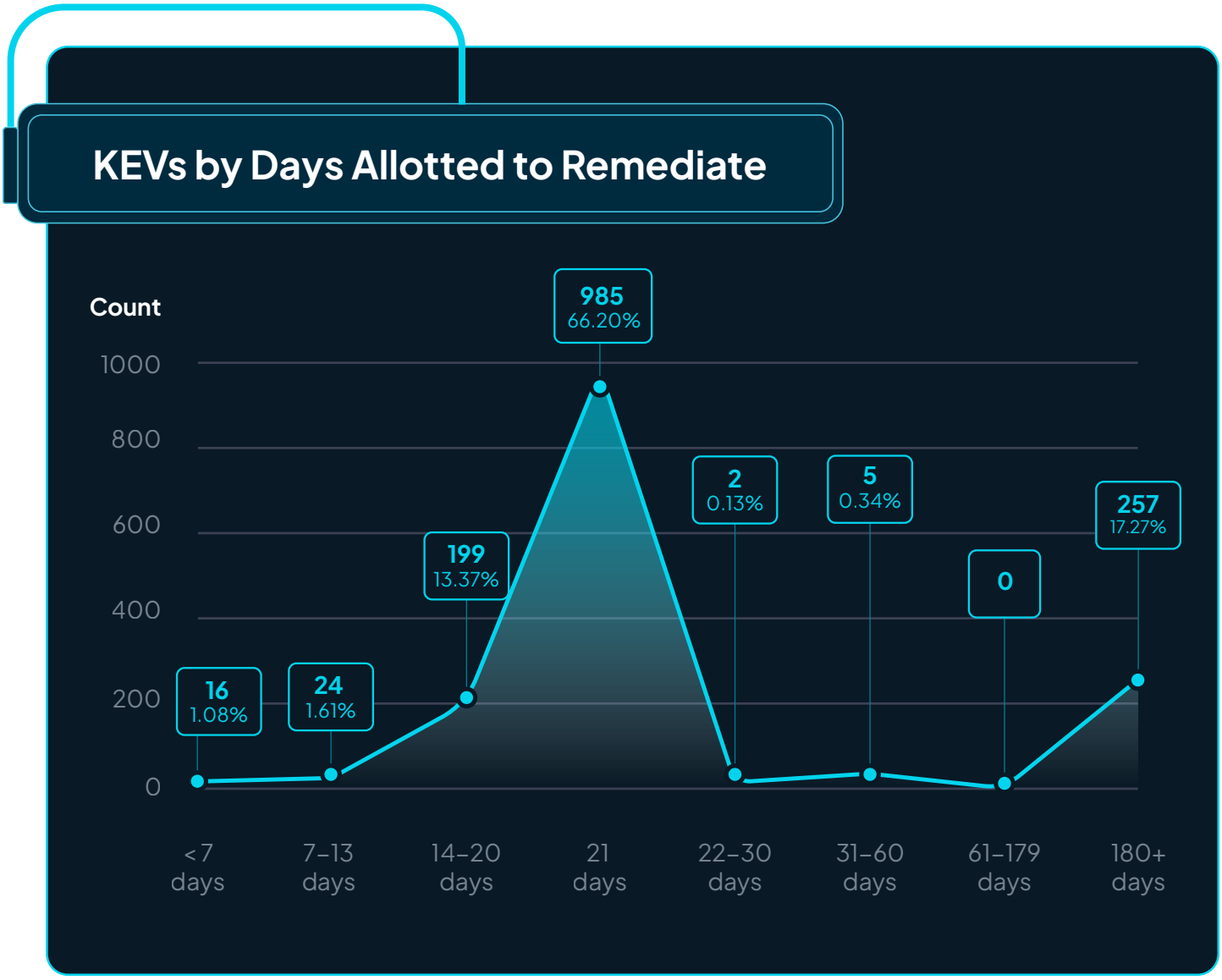


Figure 1: KEV count by days allotted

[15] https://www.youtube.com/live/y6ABaBO3hLE?t=15717s
[16] https://www.greynoise.io/blog/unveiling-vulnerability-insights-from-the-cisa-kev-catalog-at-bsideslv

Here, we can see that only 16 KEVs were listed with a due date of less than seven days hence. Those vulnerabilities are detailed in the table below.

CVE ID	Date Added	Days Allotted	Vendor	CVSS Severity	Straight Shot RCE
CVE-2025-20333	2025-09-25	1	Cisco	CRITICAL	FALSE
CVE-2025-20362	2025-09-25	1	Cisco	MEDIUM	FALSE
CVE-2025-49704	2025-07-22	1	Microsoft	HIGH	FALSE
CVE-2025-49706	2025-07-22	1	Microsoft	MEDIUM	FALSE
CVE-2025-53770	2025-07-20	1	Microsoft	CRITICAL	TRUE
CVE-2025-5777	2025-07-10	1	Citrix	CRITICAL	TRUE
CVE-2025-7775	2025-08-26	2	Citrix	CRITICAL	TRUE
CVE-2024-21893	2024-01-31	2	Ivanti	HIGH	TRUE
CVE-2023-36845	2023-11-13	4	Juniper	CRITICAL	TRUE
CVE-2023-36844	2023-11-13	4	Juniper	MEDIUM	FALSE
CVE-2023-36846	2023-11-13	4	Juniper	MEDIUM	FALSE
CVE-2023-36847	2023-11-13	4	Juniper	MEDIUM	FALSE
CVE-2023-36851	2023-11-13	4	Juniper	MEDIUM	FALSE
CVE-2023-20273	2023-10-23	4	Cisco	HIGH	FALSE
CVE-2023-20198	2023-10-16	4	Cisco	CRITICAL	TRUE
CVE-2022-26134	2022-06-02	4	Atlassian	CRITICAL	TRUE

Table 3: All KEVs which were listed with a due date less than a week into the future, with CVSS base severities, and highlighted if they are "straight-shot RCE."

Some very interesting patterns emerge here immediately, as Thorpe noted in his findings from 2024. First off, a sub-week deadline is very unusual, since there were only ever 16 of them (accounting for just 1.07% of all KEVs).

Friday additions are also unusual over the history of the KEV, accounting for only 11.16% of all KEVs — unusual enough to warrant extra attention. All things being equal, and weekends excluded, we should expect about 20% of a five-day week, if distributed randomly. Here is the day-of-the-week breakdown of all KEV additions:

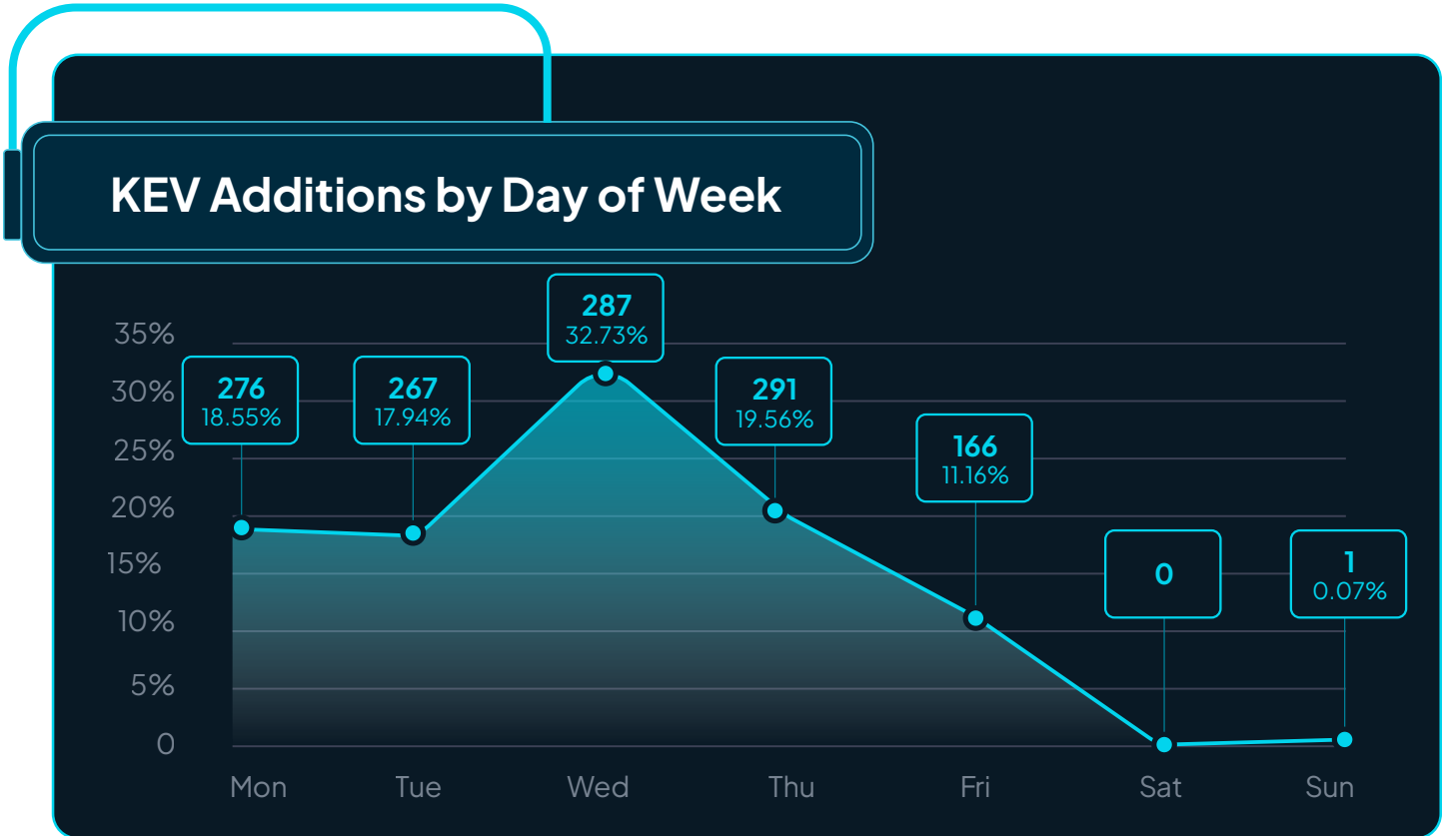


Figure 2: KEVs additions by day of week

Wednesdays are clearly over-represented, at 32.73% (12 points above a uniform distribution). By contrast, the Friday figure of 11.16% is noticeably below the expected 20% random spread, suggesting that human factors influence KEV additions on particular days. One possibility is that newsworthy KEVs are less likely to be added on Fridays, perhaps because announcements late in the week are more likely to be overlooked. Another is that adding an urgent KEV on a Friday could delay follow-up actions until Monday, with the intervening weekend potentially softening the immediate response. These patterns align with human scheduling and media considerations affecting KEV publication.

Yet exceptional cases can break this weekday pattern. Since the original research from GreyNoise, only one vulnerability, CVE-2025-53770 affecting Microsoft SharePoint servers, has been added on a weekend. The combination of its short deadline and the very wide SharePoint installation base implies significant operational urgency. Such rare out-of-cycle additions suggest that, while most KEVs follow predictable weekday patterns, urgent vulnerabilities can push publication into otherwise avoided time slots, like weekends. (We're also tracking holidays, but so far, no KEV has been added on a holiday observed in the U.S.)

Ransomware-based metrics

Vulnerabilities tagged in KEV as `knownRansomwareCampaignUse` are, as the tag would imply, identified as being part of a ransomware campaign and reported to CISA as such.

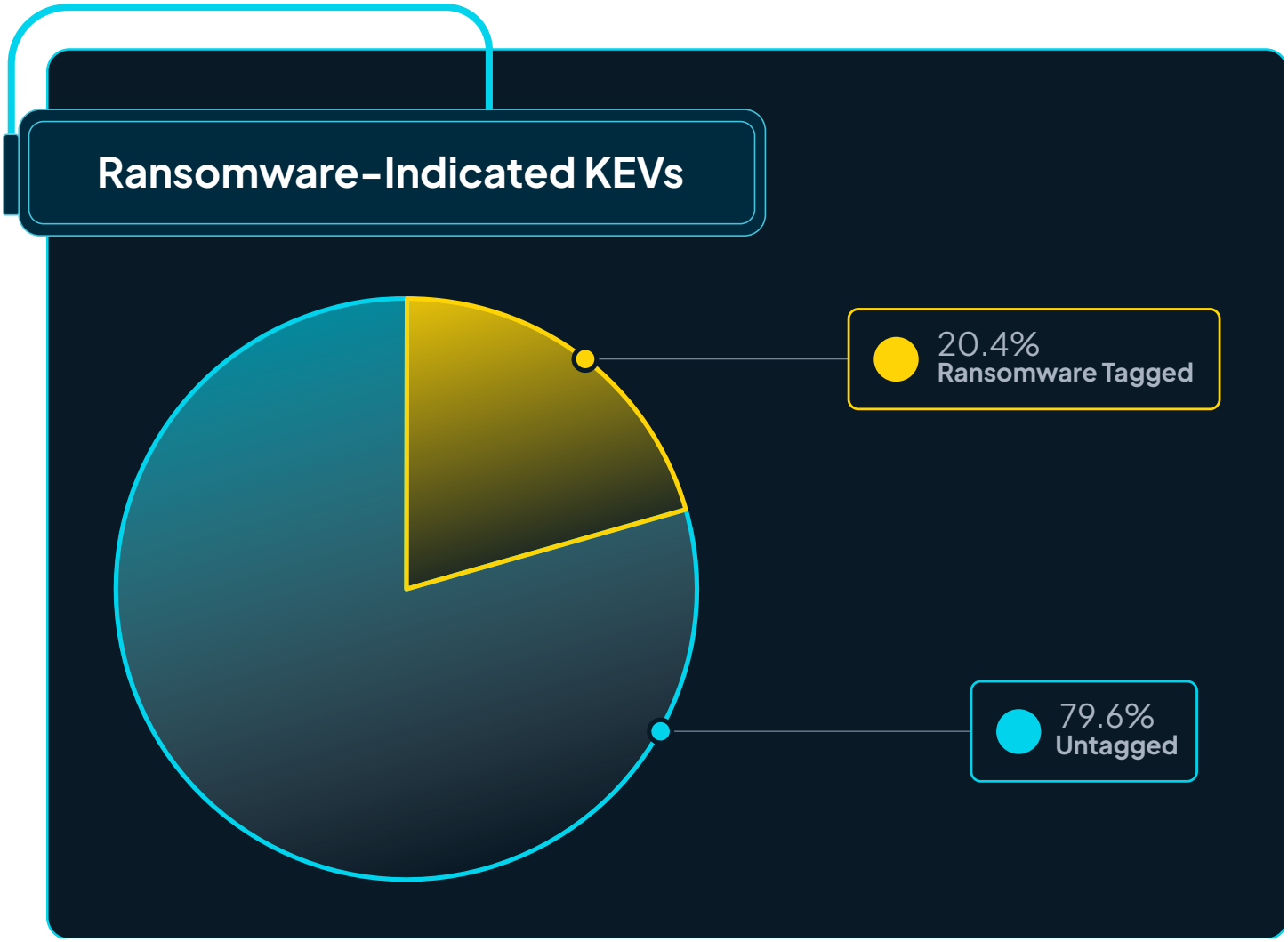


Figure 3: Ransomware vs non-ransomware tagged KEVs

A quick survey reveals that about 20% of all KEVs are tagged as being involved in ransomware campaigns. Often, this implies that commodity exploits are in circulation among at least the criminal elements of the internet, which, in turn, means that there are usually pretty good detections available. Ransomware operators are rarely particularly stealthy when compared to international espionage attackers. After all, the whole point of a ransomware campaign is to be loud and obvious to catch the attention of the victim enterprise, and the dwell time for ransomware operators tends to be pretty short, hovering around 10 to 11 days, according to the 2025 Google Mandiant M-Trends report.

All of this is delightfully fun to experiment with, but it's all also contained entirely inside the KEV. Let's move on to the next layer of data integration: information from the CVE records.

CVE records

A CVE record is all the data collected and published about a CVE-identified vulnerability. The structure is, perhaps surprisingly, very complex, having evolved over the last 26 years with plenty of input from the wider vulnerability intelligence community. One example CVE record, CVE-2025-49704[17] for the aforementioned Microsoft SharePoint vulnerability, weighs in at 188 lines (rendered normally, that would be about five pages long). As one might expect, CVE records (usually abbreviated to just "CVEs") can provide a wealth of useful information.

CVSS

The Common Vulnerability Scoring System is central to runZero's prior paper, Divining Risk: Deciphering Signals from Vulnerability Scores[18], and is the de facto standard vulnerability scoring system against which all other scoring systems are measured. Readers are encouraged to read that paper to get up to speed on what CVSS does and doesn't provide.

Previously, we touched on the notion of a "straight-shot RCE" combination of CVSS metrics (Access Vector: Network, Privileged Required: None, User Interaction: None, Integrity Impact: High), and how only about 32% of KEVs fit this criteria. But what can we tell about the rest?

First, we can measure the distribution of criticality labels conveyed by CVSS:

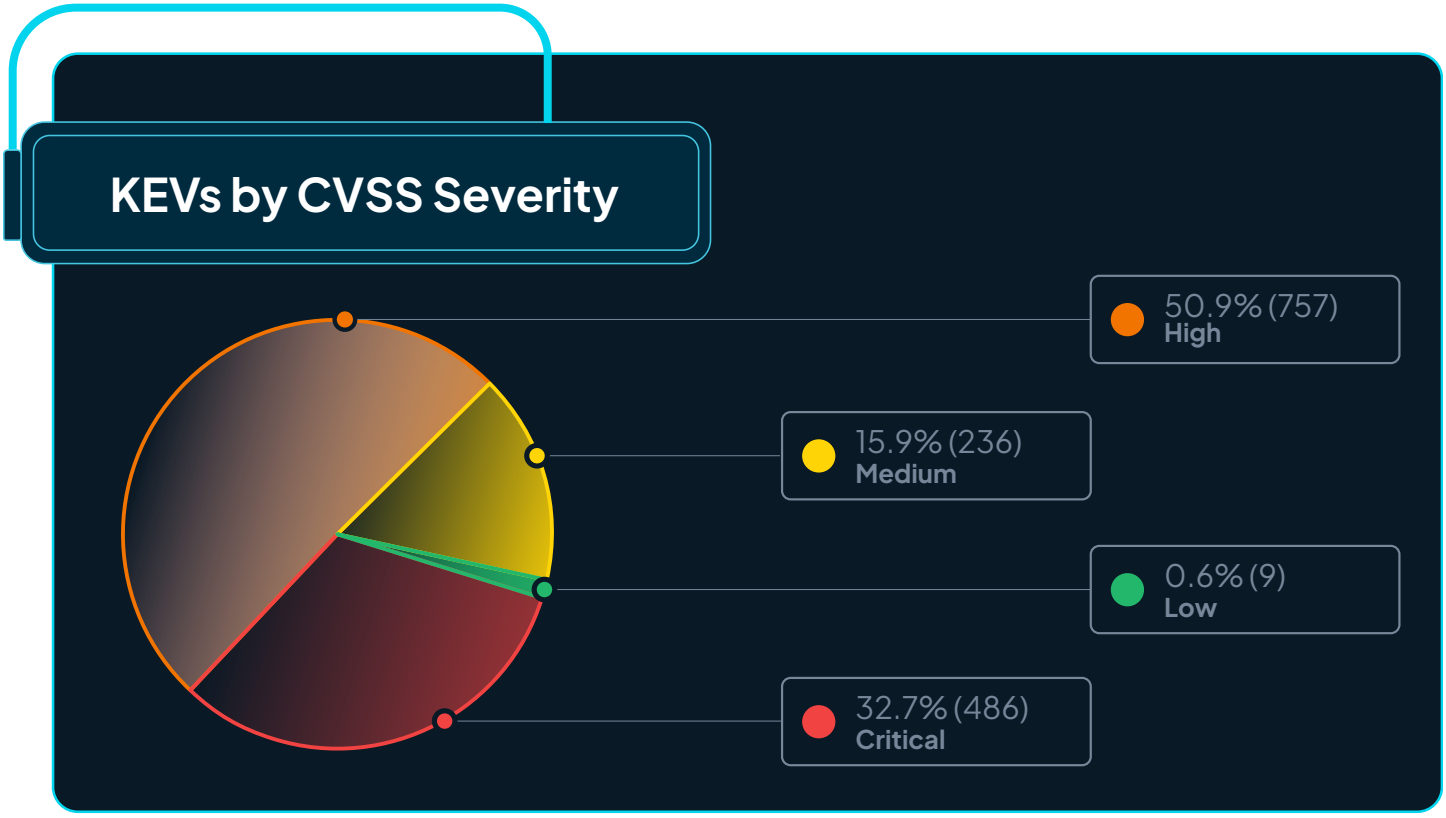


Figure 4: Distribution of CVSS severity among the KEV

[17] https://cveawg.mitre.org/api/cve/CVE-2025-49704
[18] https://www.runzero.com/resources/deciphering-signals-from-vulnerability-scores/

About half of all KEVs rate High severity, and as expected, zero percent rate as None. Over 80% of KEV-listed CVEs rate either High or Critical, leaving about 16% as Medium and less than 1% as Low. This underscores the fact that KEV-listed definitely does not equate "super-critical," but KEV entries do contain **only** those vulnerabilities with an identified exploitation event and an identified, specific target system, so even lowly Low severity bugs can make the list.

There are 24 vulnerabilities on the KEV that are purely Denial-of-Service (DoS) oriented, according to the CVSS impact metrics of Confidentiality: None, Impact: None, Availability: High. These vulnerabilities don't offer control over affected systems, but merely kick them offline, sometimes permanently. One might expect that these vulnerabilities would be used exclusively, and extensively, by ransomware operators… but you would be wrong! Only three of the CVEs tagged as involved in ransomware campaigns are also only DoS-oriented:

CVE ID	Vendor	CVSS Vector
CVE-2017-10271	Oracle	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H
CVE-2019-2725	Oracle	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H
CVE-2021-41379	Microsoft	CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H

Table 4: The three KEVs involved in ransomware campaigns that are purely Denial-of-Service

Sidebar: recasting CVSS scores

For the purpose of this research, CVSS scores were recast from version 4.0 or 3.0 when those scores were available, but version 3.1 scores were not, specifically for comparative analysis purposes. In no way should this be read as an endorsement of version 3.1 over 4.0, but simply due to the fact that casting down to 3.1 from 4.0 is mechanically trivial with the following conversion logic:

CVSS 4.0	CVSS 3.1
Access Vector	Unchanged
Attack Complexity	Unchanged
Privileges Required	Unchanged
User Interaction	If "Any" or "Partial," convert to "Required"
Vulnerable System Impacts	Becomes Impacts
Any Subsequent System Impacts of High or Low	Marks Scope as Changed

Table 5: Conversion table for CVSS v4.0 to CVSS v3.1

The reverse, casting up to 4.0, would be much more involved, since figuring out the details on the new fields in CVSS v4.0 tends to require rather extensive knowledge of the vulnerability as well as the systems targeted.

CWE IDs

Common Weakness Enumeration IDs are a rich taxonomy describing the "kind" of vulnerability a particular CVE record describes. When used correctly, CWEs can be a very useful mechanism to flag those vulnerabilities that are both commonly understood and have well-known implications for exposure criticality.

Among the KEV, there are 168 unique CWEs mentioned. Below is a chart of the most commonly mentioned CWEs.

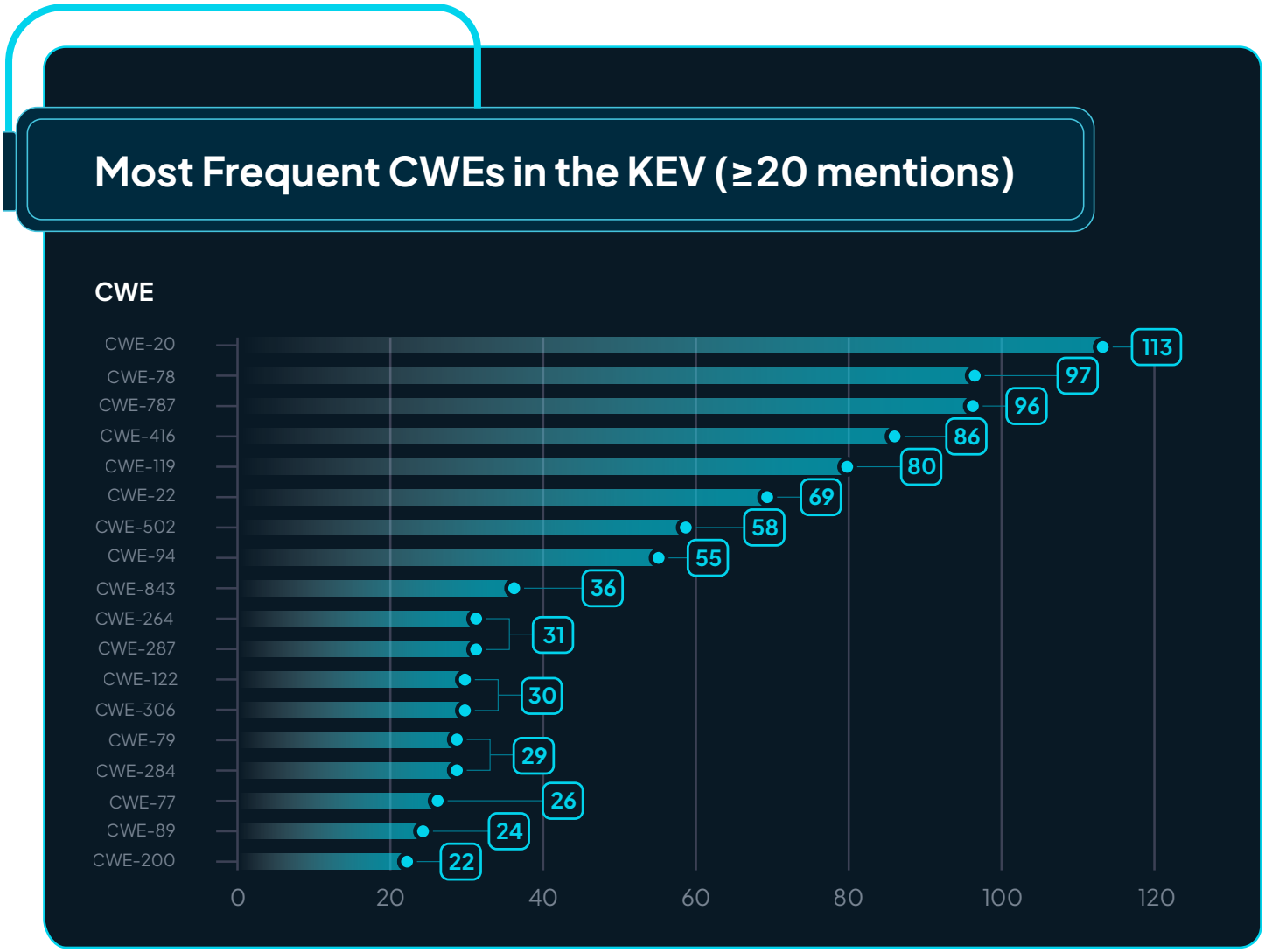


Figure 5: The most frequent CWEs in the KEV with at least 20 mentions

Experienced CWE-scryers will notice that CWE-20 is at the very top of the list, and that represents "CWE-20: Improper Input Validation." Its use is discouraged by the vulnerability disclosure community, since it's so general, and effectively means, "something bad happened when reading input." CWE-200 is also overbroad, as "CWE-200: Exposure of Sensitive Information to an Unauthorized Actor" just means "we lost control of your private data, somehow." Ultimately, what this means is that while every KEV listing does have a CWE identifier, 10% off the bat have obviously useless CWEs.

This is a case where the less common indicators can be more valuable. At the bottom of the list, there are 73 CWEs that are mentioned exactly once, and about 300 CWE IDs that are mentioned 5 times or fewer. The singletons include:

CVE ID	Vulnerability	CWE ID
CVE-2019-18988	TeamViewer Desktop Bypass Remote Login	CWE-521
CVE-2022-28810	Zoho ManageEngine ADSelfService Plus Remote Code Execution	CWE-259
CVE-2023-45249	Acronis Cyber Infrastructure (ACI) Insecure Default Password	CWE-1393

Table 6: Three sample KEVs with unique CWE IDs

All of which are password related (according to their CWE IDs), and probably pretty easy to remediate even without a patch in hand. More importantly, since the CWEs are so unusual, it's a safe bet that whoever assigned these CWE IDs didn't take the lazy route. Instead of just picking CWE-20 or CWE-200, they took the care to precisely identify the vulnerability, so this can be a moderately strong signal that the rest of the relevant CVE data is also accurate and actionable.

Sidebar: Vulnrichment

Unfortunately, CWE identification, CVSS scores, and most other elements one might expect in a well-formed CVE record are not required fields in the CVE record standard JSON[19]. But, since 2024, the CISA Authorized Data Publisher (CISA-ADP) has been hard at work filling in the gaps on all new and recent CVEs, paying particular attention to those CVEs that land on the KEV. Because of this Vulnrichment effort[20], we can rest assured that at least some effort has been made by a reasonably neutral assessor to ensure that at least some assertions about KEV-listed CVEs exist at all.

Vulnrichment ensures that every CVE record will have at least an SSVC assessment, which itself is a pretty useful data point for first-pass triage. As of issue #208[21], all KEV entries also have CVSS and CVE assertions, even if the original CVE Numbering Authority failed to provide them.

Since the project began in 2024, the CISA-ADP has ensured that many thousands of CVEs now have CVSS CWE data, if it's possible to surmise it from the provided CVE description and references.

[19] https://cveproject.github.io/cve-schema/schema/docs/#oneOf_i0_containers_cna
[20] https://www.cisa.gov/news-events/news/unlocking-vulnrichment-enriching-cve-data
[21] https://github.com/cisagov/vulnrichment/issues/208

EPSS

EPSS is discussed at length in our 2024 paper, Divining Risk[22]. Briefly, it's a machine-learning (ML)-based approach to measuring the "threatiness" of the entire CVE corpus, daily. EPSS produces two measurements for every CVE. The score, measured from 0 to 1, is a prediction of the change of "exploitation activity" involving that vulnerability in the next 30 days, while the percentile is the rank within the universe of all CVEs.

EPSS scores can change every day as new information is integrated into EPSS, so for the purposes of this exploration, we're looking at merely today's score and percentile. Let's take a look at KEV-listed vulnerabilities to see where today's top ten EPSS-rated vulnerabilities land.

CVE ID	Vendor	Vulnerability	EPSS Today	EPSS Percentile
CVE-2024-27198	JetBrains	JetBrains TeamCity Authentication Bypass	0.94579	1.00000
CVE-2023-23752	Joomla!	Joomla! Improper Access Control	0.94517	1.00000
CVE-2017-1000353	Jenkins	Jenkins Remote Code Execution	0.94508	1.00000
CVE-2018-7600	Drupal	Drupal Core Remote Code Execution	0.94489	0.99999
CVE-2018-1000861	Jenkins	Jenkins Stapler Web Framework Deserialization of Untrusted Data	0.94485	0.99999
CVE-2021-22986	F5	F5 BIG-IP and BIG-IQ Centralized Management iControl REST Remote Code Execution	0.94485	0.99998
CVE-2023-35078	Ivanti	Ivanti Endpoint Manager Mobile Authentication Bypass	0.94482	0.99998
CVE-2018-13379	Fortinet	Fortinet FortiOS SSL VPN Path Traversal	0.94475	0.99998
CVE-2019-3396	Atlassian	Atlassian Confluence Server and Data Center Server-Side Template Injection	0.94471	0.99997
CVE-2019-2725	Oracle	Oracle WebLogic Server, Injection	0.9447	0.99997

Table 7: Top 10 KEVs, by EPSS, on January 14, 2026

[22] https://www.runzero.com/resources/deciphering-signals-from-vulnerability-scores/

While one might expect everything on the KEV to have these kinds of very high scores and percentile placements, we can see the bottom 10 KEVs are actually quite low, all under 1%:

CVE ID	Vendor	Vulnerability	EPSS Today	EPSS Percentile
CVE-2023-32373	Apple	Apple Multiple Products WebKit Use-After-Free	0.00018	0.0363
CVE-2023-23529	Apple	Apple Multiple Products WebKit Type Confusion	0.00045	0.13726
CVE-2025-43529	Apple	Apple Multiple Products Use-After-Free WebKit	0.00046	0.1403
CVE-2023-42916	Apple	Apple Multiple Products WebKit Out-of-Bounds Read	0.00047	0.14639
CVE-2023-28204	Apple	Apple Multiple Products WebKit Out-of-Bounds Read	0.00048	0.15218
CVE-2021-30762	Apple	Apple iOS WebKit Use-After-Free	0.00049	0.15336
CVE-2023-37450	Apple	Apple Multiple Products WebKit Code Execution	0.00054	0.17173
CVE-2023-42917	Apple	Apple Multiple Products WebKit Memory Corruption	0.00068	0.21164
CVE-2025-24201	Apple	Apple Multiple Products WebKit Out-of-Bounds Write	0.00068	0.21267
CVE-2024-23296	Apple	Apple Multiple Products Memory Corruption	0.00082	0.24507

Table 8: Bottom 10 KEVs, by EPSS, on January 14, 2026

To belabor the point of the KEV: we know, for certain, that all of these vulnerabilities have been exploited by criminals or spies. And yet, the chances of them getting exploited again in the future appears quite low, and this comports with a seasoned analyst's instincts about these bugs.

Notice that all ten of these bottom ten are bugs in Apple devices, mostly Apple iPhones, iPads, iWatches, and the like. Eight of ten are in WebKit, the web content rendering engine that ships with these devices. WebKit usually (but not always[23]) requires user interaction, but attacks involving iPhones tend to be highly targeted, usually affecting journalists and civil rights activists in repressive parts of the world. So, unless you're in that special class of "high value, individually targeted iPhone user," you're unlikely to be affected directly by these bugs. Finally, bugs described by Apple as CVEs tend to already be patched by the time they're documented, and iPhone updates can be difficult to avoid, even on purpose. You generally needn't lose much sleep over these bugs.

[23] https://github.com/cisagov/vulnrichment/issues/200

EPSS distributions

Now that we know the floor and ceiling of EPSS scores, we can take a look at the distribution of EPSS scores among KEVs.

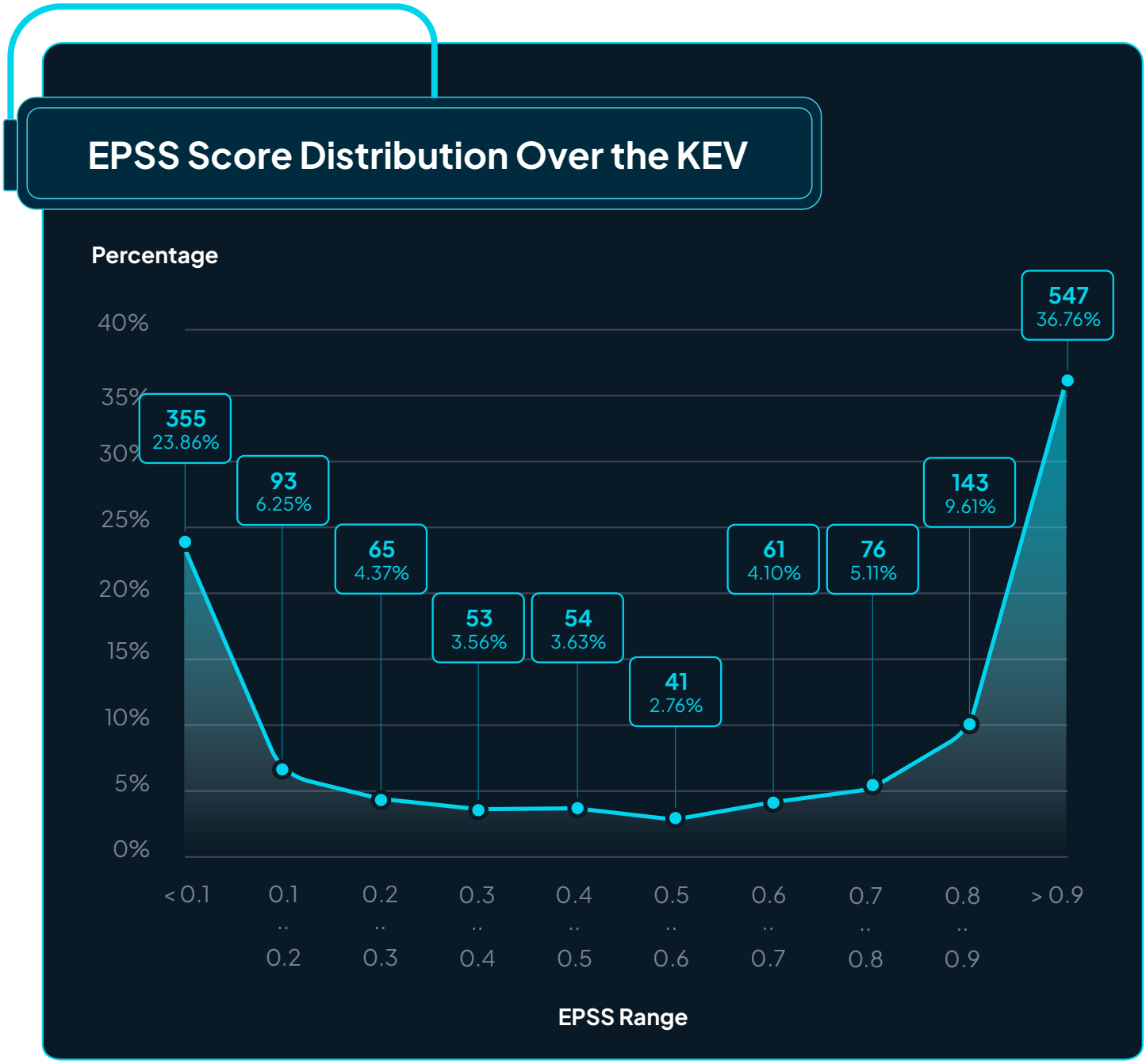


Figure 6: EPSS score bands (0.0 to 1.0), split into 10 ranges

The data show us two pretty solid anchors, at the very top and bottom of the EPSS scores represented by KEV. Most KEVs come with **the highest** possible scores, over 90% chance, of seeing exploitation activity in the wild over the next 30 days. But the next most common score band is the under 10% chance, meaning that they're **unlikely** to be exploited again any time soon. This seems like a fine mechanism to sort KEVs, if you trust EPSS scoring. At the time of writing, there are 545 KEVs that feature these very high scores, and 353 in the sub-10% band.

Taking a look at EPSS percentiles tells a different story, however:

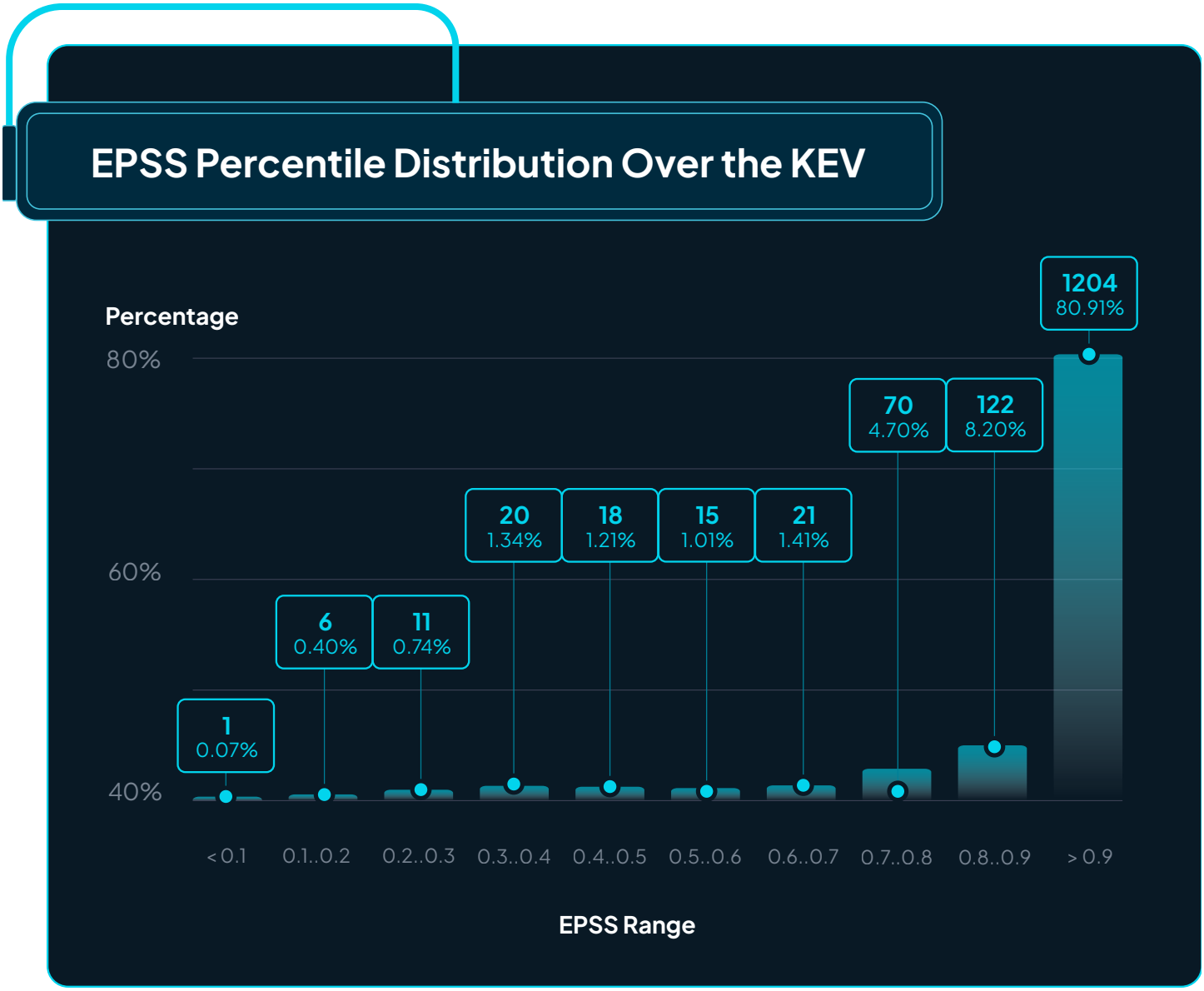


Figure 7: EPSS percentile bands (0.0–1.0), split into 10 ranges

Here, we can see that EPSS scores rise sharply from the sub-90% mark to the 90% and over band. This tells us that among the entire universe of CVEs, EPSS does seem to believe that 1,204 (80.91%) of all KEVs are, indeed, among the highest-scoring EPSS-rated vulnerabilities. Paradoxically, this also makes sense: we know that the vast, vast majority of CVE-identified vulnerabilities never see any detectable exploitation at all.

These EPSS score distributions are quite fascinating, and one area of future work is to collect the historical EPSS scores of vulnerabilities taken the day before and the day after they were added to the KEV. Doing so would, for once and for all, provide a pretty good measurement of how much "KEV tax" is being levied to EPSS scores at the time of their addition. Tracking these over sliding 30–day windows may inform how long it takes for that KEV tax to decay to get a better sense of "but for KEV, what are the true EPSS scores for KEV-listed vulnerabilities." However, the effect of KEV addition seems to be permanently inflationary; even years after being added to the KEV, scores remain high. More study on this effect is advised.

32% of KEVs are "Metasploitable"

The Metasploit Framework has spent over 20 years at the forefront of practical exploitation for the purpose of ramping up defender skills. Funded by Rapid7 since 2009, it's still going strong today as an open source project with contributions from around the world. Cybersecurity expert Josh Corman stated, "casual attacker power grows at the rate of Metasploit," in a 2011 blog post, Intro to HDMoore's Law[24], and that law still appears to hold today.

What can Metasploit, a collection of exploits and other tricks for gaining, enhancing, and maintaining access to compromised systems, tell us about the KEV? Well, out of the gate, we can tell that there are **464 KEVs associated with at least one Metasploit module**. This means that just about a third of all KEVs are trivially exploitable today, as Metasploit modules are free, easy to use, and well-understood by attackers and defenders alike.

If that weren't enough, we could dig just a little deeper and see what kinds of Metasploit modules we're talking about. But, knowing nothing else, I'd recommend patching up or otherwise mitigating those issues yesterday, or more accurately, yesteryear. The complete list of "Metasploitable" KEVs is in an appendix at the end of this paper.

That said, we can in fact tell a little bit more about the nature of these Metasploit modules.

444 exploit modules exist in Metasploit for KEVs. Exploit modules are the bread and butter of Metasploit, and most tend to provide that coveted initial access starting from zero. A few are of the lie-in-wait variety, and some require some setup, but the vast majority are simply point, click, pwn.

97 auxiliary modules exist for KEVs. Auxiliary modules are simply those modules that don't immediately end with a Meterpreter shell on the target. Instead, they tend to be the kind of module you need to use when what you're after is more of an information leak, a default web application password that can get you access to a configuration console, or something else that's not immediate and total control on the underlying operating system. And, from what we've learned of the KEV in this paper, most KEVs are exactly that kind of vulnerability, so this is a reasonably healthy representation in the Metasploit corpus. Recall that Metasploit is designed with popping root shells in mind; auxiliary modules are more of a bonus.

7 are post-exploitation modules. These tend to be used almost exclusively for privilege escalation bugs and local code execution once a shell is obtained. This is a somewhat surprising finding, as I would expect to see more of this sort in Metasploit, since Metasploit is pretty good at session management and ensuring further exploitation on an already-compromised system, and given there are a somewhat surprising number of KEVs that describe vulnerabilities like these. I suppose the usual Metasploit module contributor is more interested in those tasty remotely exploitable bugs.

Note that just because it's **not** in Metasploit doesn't mean commodity exploits **don't** exist. There are many one-off exploits that show up. There are many private exploits. And of course, there's a whole other common, popular exploit development platform: Nuclei, from ProjectDiscovery.

[24] https://blog.cognitivedissidents.com/2011/11/01/intro-to-hdmoores-law/

27% of KEVs are “Nucleiable”

If you’re too young to remember when The Matrix was released in theaters, Nuclei may be more your speed. Nuclei is the application testing framework from ProjectDiscovery, and unlike Metasploit, Nuclei started as a web application assessment platform. Since so many KEVs involve a web application vector, we took a look at the crossover between KEV and the body of Nuclei templates. As it happens, there are **398** Nuclei templates suitable for testing KEV vulnerabilities.

One interesting combined stat that can help motivate mitigation and remediation would be pinpointing all those KEVs that are exploited in **both** Nuclei and Metasploit. After all, in Divining Risk[25], we (re)discovered that three events seem to juice EPSS scores quite a bit for CVEs:

- Getting added to the KEV
- Having a Metasploit module
- Having a Nuclei template

We can confirm this by looking at EPSS creator Jay Jacobs’ slide from VulnCon 2025, reproduced below from the YouTube stream[26] of that event:

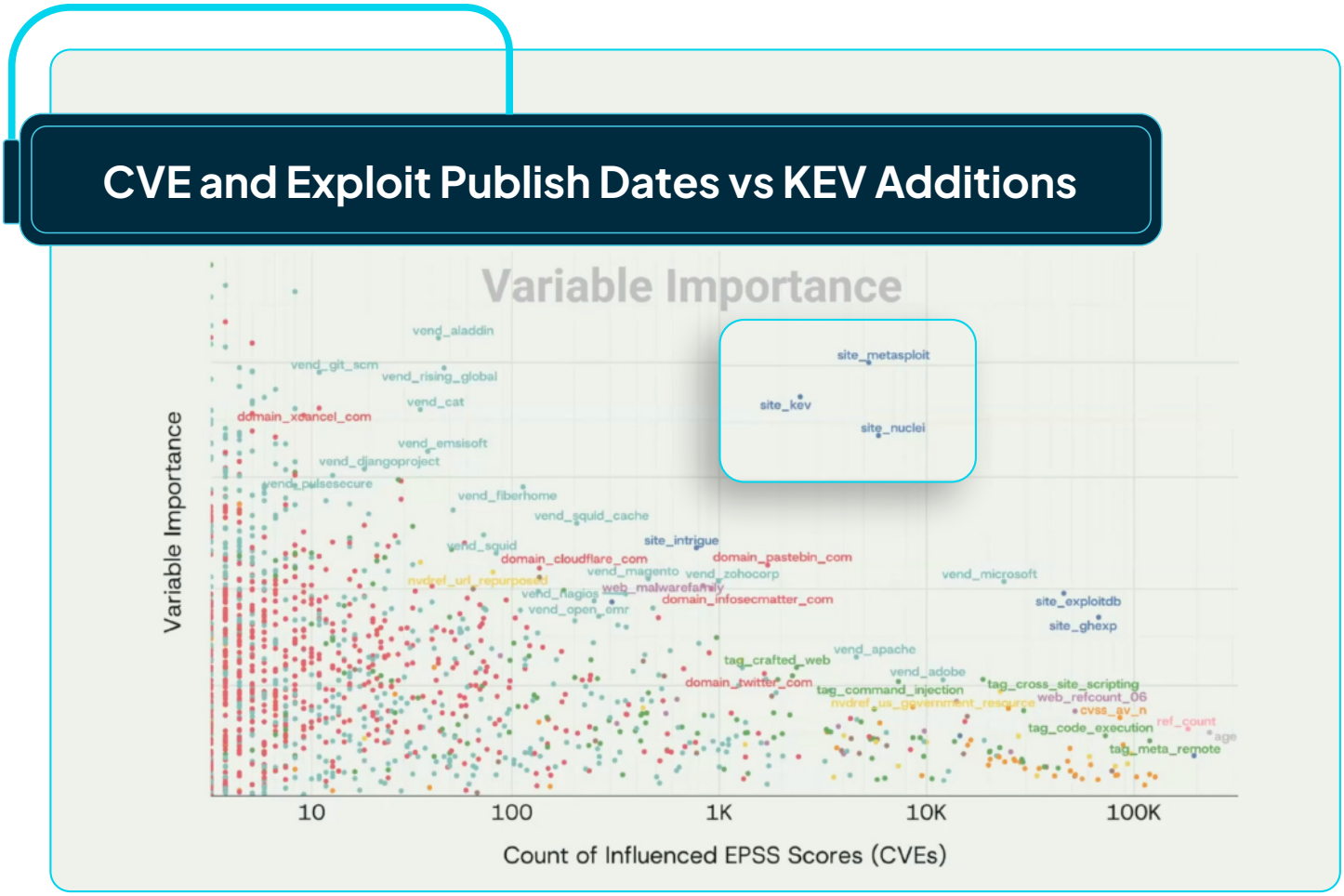


Figure 8: EPSS variable importance from Jay Jacobs

[25] https://www.runzero.com/resources/deciphering-signals-from-vulnerability-scores/
[26] https://youtu.be/o1XKTgX1JeE?si=-wI9m4FMDOh4rBok&t=2282

Since we can now directly measure all three events, daily, we can reverse out the most interesting (and most severe according to EPSS) vulnerabilities. The final number is only 235 KEVs as of this writing, which is quite comfortable for a first pass set of things to check, patch, and monitor.

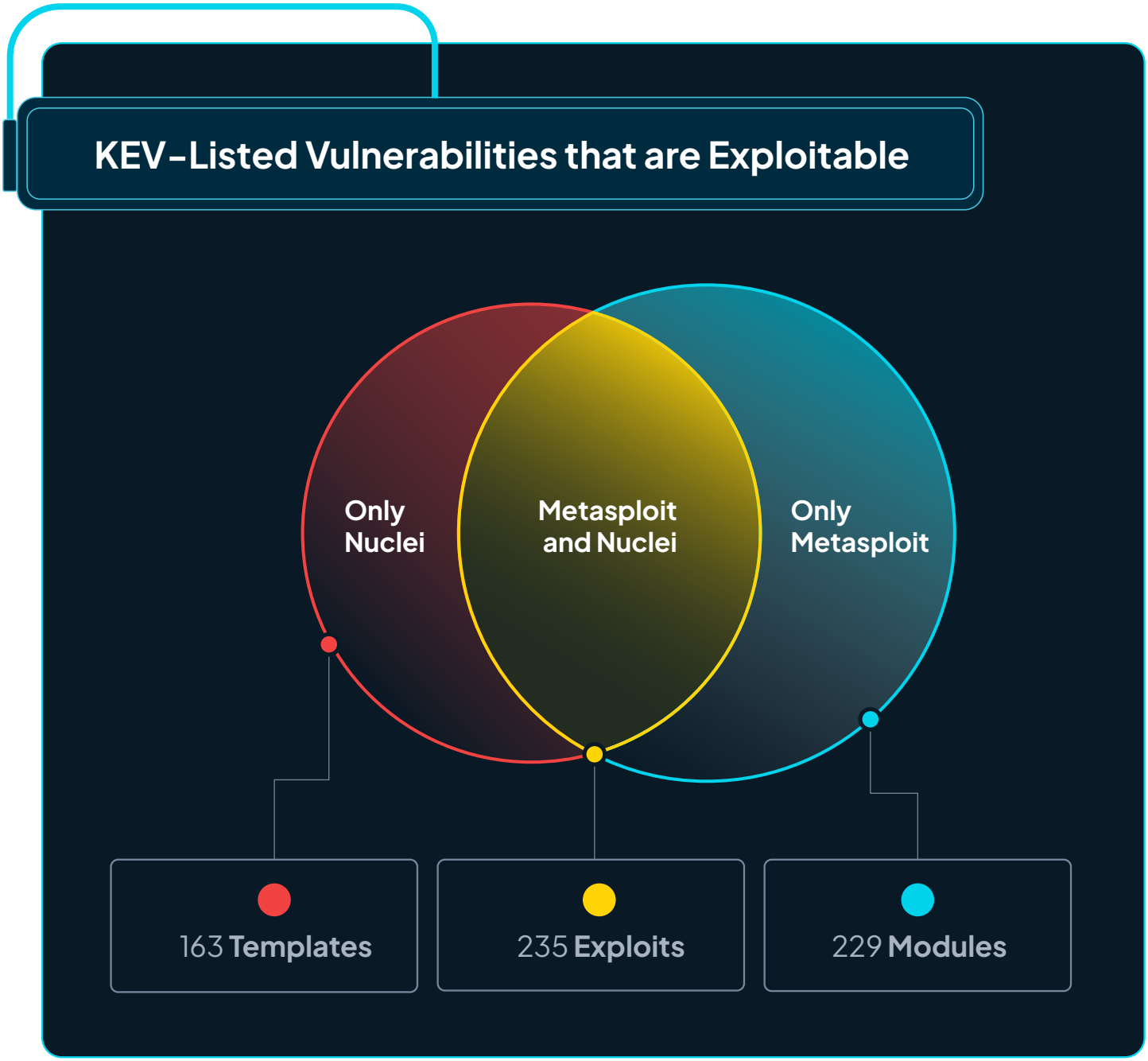


Figure 9: KEV-listed vulnerabilities that are exploitable from Metasploit, Nuclei, or both

Again, as with Metasploit, there is no reason to believe that if an exploit is "missing" from Nuclei, that exploit does not truly exist. There are many, many other sources for commodity exploits. Nuclei and Metasploit, though, are special — the exploits implemented in these robust, expert-reviewed platforms are well-vetted and are much less likely to be AI-generated "PoC slop," or broken exploits that have no hope of executing successfully.

MITRE ATT&CK mappings

MITRE ATT&CK is a behaviorally oriented knowledge base that describes adversary tactics, techniques, and procedures (TTPs) observed in real-world intrusions. Unlike scoring systems such as CVSS or EPSS, ATT&CK is not predictive. Instead, it provides a common language for describing how exploitation and post-exploitation activity unfolds once an attacker is active in an environment.

The ATT&CK mappings used in this paper are sourced from the Center for Threat-Informed Defense, via their public mappings explorer project. These mappings represent expert analysis that links CVE-identified vulnerabilities to ATT&CK techniques based on observed or inferred attacker behavior. As such, they should be interpreted as descriptive and contextual, rather than as a complete or mechanistic model of exploitation.

Only 424 of the 1,488 KEVs currently have ATT&CK mappings, making this the sparsest enrichment dimension analyzed in this paper.

As of the most recent CTID update (July, 2025), intersecting with the most recent KEV (January, 2026), these 424 KEV-listed vulnerabilities map to:

- 1,228 total ATT&CK mapping entries
- 174 unique ATT&CK techniques
- 155 enterprise techniques
- 19 mobile techniques

Despite ATT&CK's sparsity (owing largely to a lack of updates since July, 2025), this breadth alone reinforces a key theme of this paper: KEV-listed vulnerabilities are not narrowly concentrated in a small set of attacker behaviors, but instead span a wide range of exploitation and post-exploitation activity. Importantly, the number of ATT&CK techniques associated with a vulnerability should not be interpreted as a measure of severity or likelihood of exploitation. Rich ATT&CK mappings often reflect the availability of public reporting and post-incident analysis, not necessarily broader attacker adoption.

Mapping types

CTID classifies ATT&CK mappings into several high-level categories:

- Exploitation Technique: how the vulnerability is initially exploited
- Primary Impact: the immediate post-exploitation effect
- Secondary Impact: follow-on behaviors enabled by exploitation

These distinctions are important. A single KEV often maps to multiple ATT&CK techniques across different phases of an intrusion, and the presence of a secondary impact mapping does not imply inevitability, only plausibility.

As Enterprise ATT&CK techniques dominate the KEV, the top 20 techniques are detailed in the table below:

ATT&CK ID	ATT&CK Mapping	Mapping Type	Count
T1190	Exploit Public-Facing Application	exploitation_technique	154
T1059	Command and Scripting Interpreter	primary_impact	141
T1068	Exploitation for Privilege Escalation	exploitation_technique	40
T1078	Valid Accounts	exploitation_technique	34
T1203	Exploitation for Client Execution	exploitation_technique	32
T1204.002	Malicious File	exploitation_technique	31
T1068	Exploitation for Privilege Escalation	primary_impact	27
T1005	Data from Local System	primary_impact	24
T1105	Ingress Tool Transfer	primary_impact	23
T1133	External Remote Services	exploitation_technique	23
T1005	Data from Local System	secondary_impact	22
T1189	Drive-by Compromise	exploitation_technique	21
T1059	Command and Scripting Interpreter	secondary_impact	16
T1505.003	Web Shell	secondary_impact	15
T1486	Data Encrypted for Impact	secondary_impact	14
T1496	Resource Hijacking	secondary_impact	14
T1055	Process Injection	exploitation_technique	14
T1059.004	Unix Shell	primary_impact	13
T1059	Command and Scripting Interpreter	exploitation_technique	13
T1003	OS Credential Dumping	secondary_impact	12

Figure 9: KEV-listed vulnerabilities that are exploitable from Metasploit, Nuclei, or both

This spread tells a very specific story about the makeup of this area of the KEV that was analyzed by the CTID. Most obviously, initial access via exposed services (T1190) is the most common technique asserted with these KEVs. This is also the class of vulnerability most amenable to repeatable, unauthenticated exploitation, which likely explains its overlap with public exploit tooling. Cross-referencing these ATT&CK techniques with the corpus of Metasploit modules and Nuclei templates, we can prove that hypothesis quickly and see that there does appear to be a relationship:

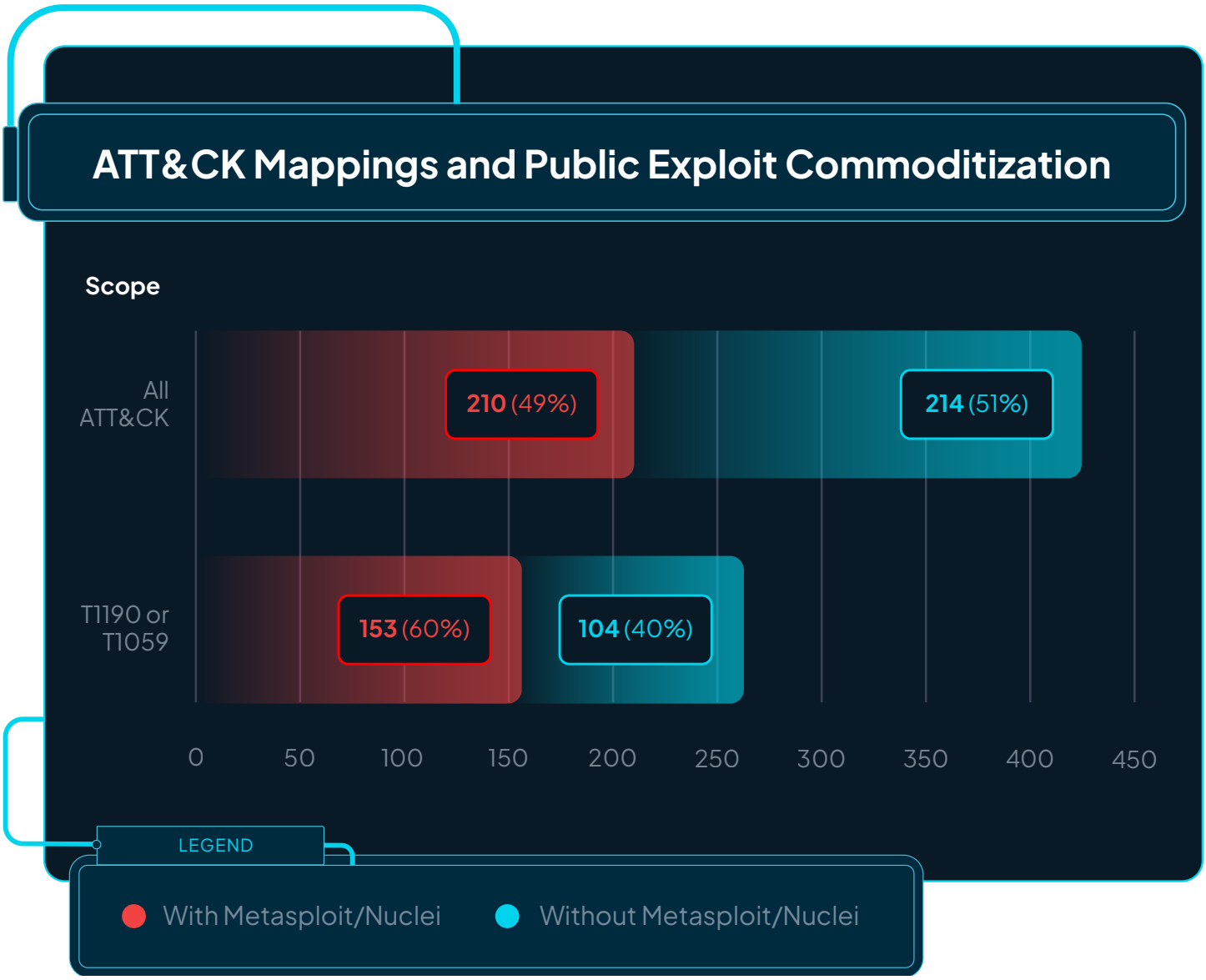


Figure 10: ATT&CK mappings and public exploit commoditization

Figure 10, above, shows the relationship between ATT&CK techniques and public exploit commoditization. Of the 424 KEVs with ATT&CK mappings, approximately half (210) have at least one Metasploit module or Nuclei template. When focusing on the two most common techniques, T1190 (Exploit Public-Facing Application) and T1059 (Command and Scripting Interpreter), this proportion increases to roughly 60% of this subset of 257 KEVs, indicating that vulnerabilities associated with these behaviors disproportionately attract public exploit development.

Mobile ATT&CK mappings are far less common. Mobile exploits are rarely reported, as they're both expensive to develop and difficult to detect by typical exposure management or incident response programs. In fact, most mobile exploitation tends to be first reported by specialized, elite incident responders or the vendors themselves, and often only after an incident. The table below describes all the available mobile ATT&CK techniques and impacts mapped today:

ATT&CK ID	ATT&CK Mapping	Mapping Type	Count
T1658	Exploitation for Client Execution	exploitation_technique	7
T1456	Drive-By Compromise	exploitation_technique	5
T1623	Command and Scripting Interpreter	primary_impact	3
T1544	Ingress Tool Transfer	primary_impact	3
T1660	Phishing	exploitation_technique	2
T1658	Exploitation for Client Execution	primary_impact	2
T1629	Impair Defenses	secondary_impact	2
T1575	Native API	secondary_impact	2
T1623	Command and Scripting Interpreter	secondary_impact	2
T1406	Obfuscated Files or Information	secondary_impact	2
T1638	Adversary-in-the-Middle	secondary_impact	2
T1437	Application Layer Protocol	primary_impact	1
T1404	Exploitation for Privilege Escalation	primary_impact	1
T1437.001	Web Protocols	primary_impact	1
T1631	Process Injection	exploitation_technique	1
T1642	Endpoint Denial of Service	primary_impact	1
T1404	Exploitation for Privilege Escalation	exploitation_technique	1
T1629	Impair Defenses	primary_impact	1
T1458	Replication Through Removable Media	secondary_impact	1
T1461	Lockscreen Bypass	exploitation_technique	1

Table 10: Mobile ATT&CK mappings (one vulnerability may have several mappings)

These mappings align closely with the EPSS findings discussed earlier, in that many mobile KEVs correspond to highly targeted exploitation campaigns rather than broad, repeatable attacks. This effect is demonstrated in the table below:

CVE ID	EPSS Score	EPPS Percentile
CVE-2025-24201	0.00068	0.21267
CVE-2025-21479	0.00083	0.24666
CVE-2025-6558	0.00094	0.26988
CVE-2025-43200	0.00323	fff7d9
CVE-2025-6554	0.00369	0.58251
CVE-2025-5419	0.01353	0.79693
CVE-2025-31200	0.01731	0.82003
CVE-2021-38000	0.0251	0.84954
CVE-2025-31201	0.04376	0.88622
CVE-2022-38181	0.24547	0.95942
CVE-2025-24085	0.28136	0.96325
CVE-2025-24200	0.43616	0.97404
CVE-2025-27363	0.76675	0.98901
CVE-2023-4863	0.94083	0.99901

Table 11: KEVs with ATT&CK Mobile mappings, with mostly low EPSS scores and mid-range percentiles

Here, we can see that these mobile-flavored vulnerabilities tend toward the extremely low side of EPSS predictions, with a median chance of only about 2.12% of being exploited in the next 30 days. (As covered in the EPSS section of this paper, though, we can also see that these vulnerabilities, like all KEV-listed vulnerabilities, tend to rank rather high in overall percentile rankings among all CVEs, at a median of about 83.46%. This effect is almost certainly by dint of being listed on the KEV at all.)

In both the Enterprise and Mobile realms, ATT&CK mappings are most valuable after exploitation is assumed, helping defenders reason about detection coverage, telemetry gaps, and response readiness once a KEV-listed vulnerability has been exploited. In this sense, ATT&CK complements EPSS and KEV deadlines by addressing a different question: “What happens next?”

Research findings and KEV insights

Throughout this paper, we've established a natural narrative, by examining several aspects of KEV vulnerabilities in turn:

- **KEV:** What the vulnerability is and how urgent CISA believes it is
- **CVSS, CWE, and SSVC:** What the vendor, researcher, or CISA believes about the vulnerability
- **EPSS:** What background trends about the vulnerability can predict theoretically
- **Metasploit and Nuclei:** How exploitation works practically
- **MITRE ATT&CK:** What responses to exploitation are appropriate and worthwhile

The real power of vulnerability data analysis, however, lies in how these data sources can be integrated and synthesized with each other in new and novel ways, with the goal of defensibly prioritizing which vulnerabilities are worth an IT security analyst's time and effort today, which can be deferred until tomorrow, and which can safely wait until the next change control window.

We've done a little of this already previously in this paper, such as examining which vulnerabilities are exploitable with both Metasploit and Nuclei, and thus, implying the vulnerability is ultra-commoditized and likely to show up on our intrusion detection alerting with some frequency. We've also connected ATT&CK mappings with Metasploit and Nuclei exploit development, showing that those vulnerabilities known to be associated with either T1190 (Exploit Public-Facing Application) and T1059 (Command and Scripting Interpreter) are more likely to attract the attention of public exploit developers.

But what if we took a look at four different vectors, all anchored in time?

Time-based analysis across vulnerability identification & exploitation

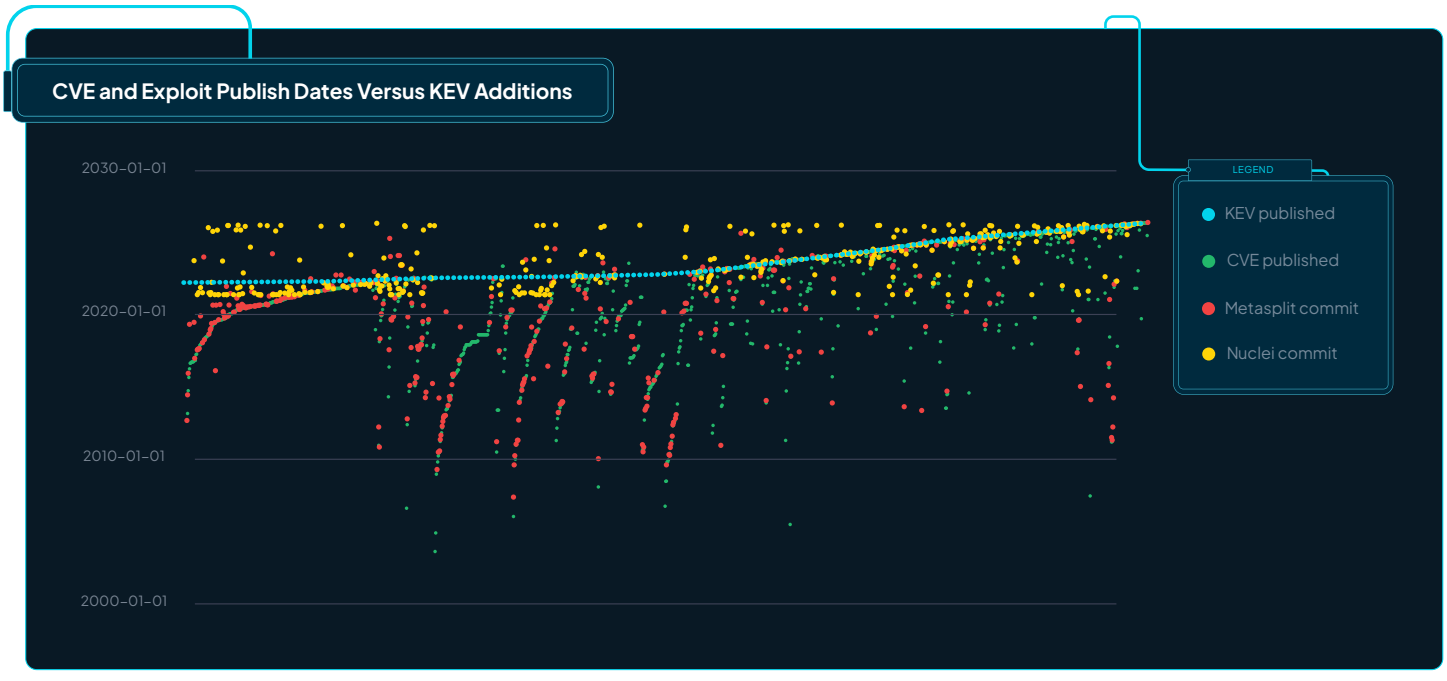


Figure 11: Time relationships between CVE publish dates, KEV publish dates, and commodity exploit publish dates

Figure 11, above, describes how we, as physical entities hurtling forward through time at nearly the speed of light, experience the notification of exploitation evidence through the KEV, the documentation of the vulnerability through CVE, and the development of public commodity exploits (proxied by Metasploit and Nuclei first commits). Ideally, this kind of cross-domain analysis can help inform future decisions when a new vulnerability reaches the KEV. What's likely already happened by the time a KEV is announced by CISA, what's likely to happen next, and when?

First off, KEVs nearly always lag against the original CVE publish dates, which is unsurprising since KEVs require CVEs as a gate to publishing. This is reflected in the graph by a consistent green "dripping" of CVE records below the (blue) point at which the KEV was announced. We can also see that (red and purple) public commodity exploits usually — but not always — appear on the scene well before the KEV. Somewhat routinely, these exploits can appear years ahead of KEVs.

Oddly, two issues are extreme outliers. These two CVEs were published well after their listing on the KEV:

- CVE-2019-8720, a RedHat issue, was added to the KEV on 2022-05-23, but wasn't published until 2023-03-06.
- CVE-2022-26486, a Mozilla issue, was added to the KEV on 2022-03-07, but wasn't published until 2022-12-22.

While it's somewhat common for KEV entries to be published a few days before the full CVE record is published (this effect happens routinely with Apple vulnerabilities, much to the annoyance of vulnerability researchers everywhere), it's weird for the KEV to front-run a CVE ID nine months ahead of time.

We can also see there's often a very tight turnaround between KEV listings and commodity exploit development. Public exploits do tend to pre-date KEV — sometimes by years — but lately, it would appear that there's a trend toward tighter coupling between a KEV addition and the introduction of a Metasploit module or Nuclei template.

It would be tempting to blame the mere act of publishing the KEV for this, but I'm not so sure. After all, KEVs are generated in response to confirmed exploitation in the wild, and public exploit developers often share many of the same intelligence sources as CISA (namely, infosec social media and tech news outlets). While we cannot say precisely what drives public exploit development (they're/we're an odd bunch), I would be willing to bet that it's the publicly noticed exploitation activity that drives both the KEV addition and public exploit development.

Finally, this chart demonstrates that, lately, there's a growing disconnect between Nuclei templates and Metasploit modules. We can see that, recently, of the (purple) Metasploit modules that significantly predate KEV additions, their Nuclei counterparts tend to hew much closer to KEV publish dates. It's unclear why this might be, and more research is needed to find out.

There are many other interesting relationships, based on time, between the CVE, the KEV, and the exploits that come along. For example, there are a few CVEs, KEVs, and exploits that are suspiciously close to one another:

CVE	KEV vulnerability	KEV and CVE published	Exploit first commit	Exploit source	KEV to Exploit Delta
CVE-2025-64446	Fortinet FortiWeb Path Traversal	2025-11-14	2025-11-14	Metasploit	0
CVE-2025-64446	Fortinet FortiWeb Path Traversal	2025-11-14	2025-11-14	Nuclei	0
CVE-2021-44228	Apache Log4j2 Remote Code Execution	2021-12-10	2021-12-11	Nuclei	1
CVE-2024-0012	Palo Alto Networks PAN-OS Management Interface Authentication Bypass	2024-11-18	2024-11-19	Metasploit	1
CVE-2024-0012	Palo Alto Networks PAN-OS Management Interface Authentication Bypass	2024-11-18	2024-11-19	Nuclei	1
CVE-2024-9474	Palo Alto Networks PAN-OS Management Interface OS Command Injection	2024-11-18	2024-11-19	Metasploit	1
CVE-2024-9474	Palo Alto Networks PAN-OS Management Interface OS Command Injection	2024-11-18	2024-11-19	Nuclei	1
CVE-2023-20198	Cisco IOS XE Web UI Privilege Escalation	2023-10-16	2023-10-18	Nuclei	2
CVE-2021-44228	Apache Log4j2 Remote Code Execution	2021-12-10	2021-12-13	Metasploit	3
CVE-2023-35078	Ivanti Endpoint Manager Mobile Authentication Bypass	2023-07-25	2023-07-28	Nuclei	3
CVE-2024-21893	Ivanti Connect Secure, Policy Secure, and Neurons Server-Side Request Forgery (SSRF)	2024-01-31	2024-02-03	Nuclei	3
CVE-2025-53770	Microsoft SharePoint Deserialization of Untrusted Data	2025-07-20	2025-07-23	Metasploit	3
CVE-2025-53770	Microsoft SharePoint Deserialization of Untrusted Data	2025-07-20	2025-07-23	Nuclei	3
CVE-2025-58034	Fortinet FortiWeb OS Command Injection	2025-11-18	2025-11-21	Metasploit	3
CVE-2024-3400	Palo Alto Networks PAN-OS Command Injection	2024-04-12	2024-04-16	Nuclei	4
CVE-2024-3400	Palo Alto Networks PAN-OS Command Injection	2024-04-12	2024-04-17	Metasploit	5
CVE-2022-37042	Synacor Zimbra Collaboration Suite (ZCS) Authentication Bypass	2022-08-11	2022-08-17	Nuclei	6
CVE-2024-21893	Ivanti Connect Secure, Policy Secure, and Neurons Server-Side Request Forgery (SSRF)	2024-01-31	2024-02-06	Metasploit	6

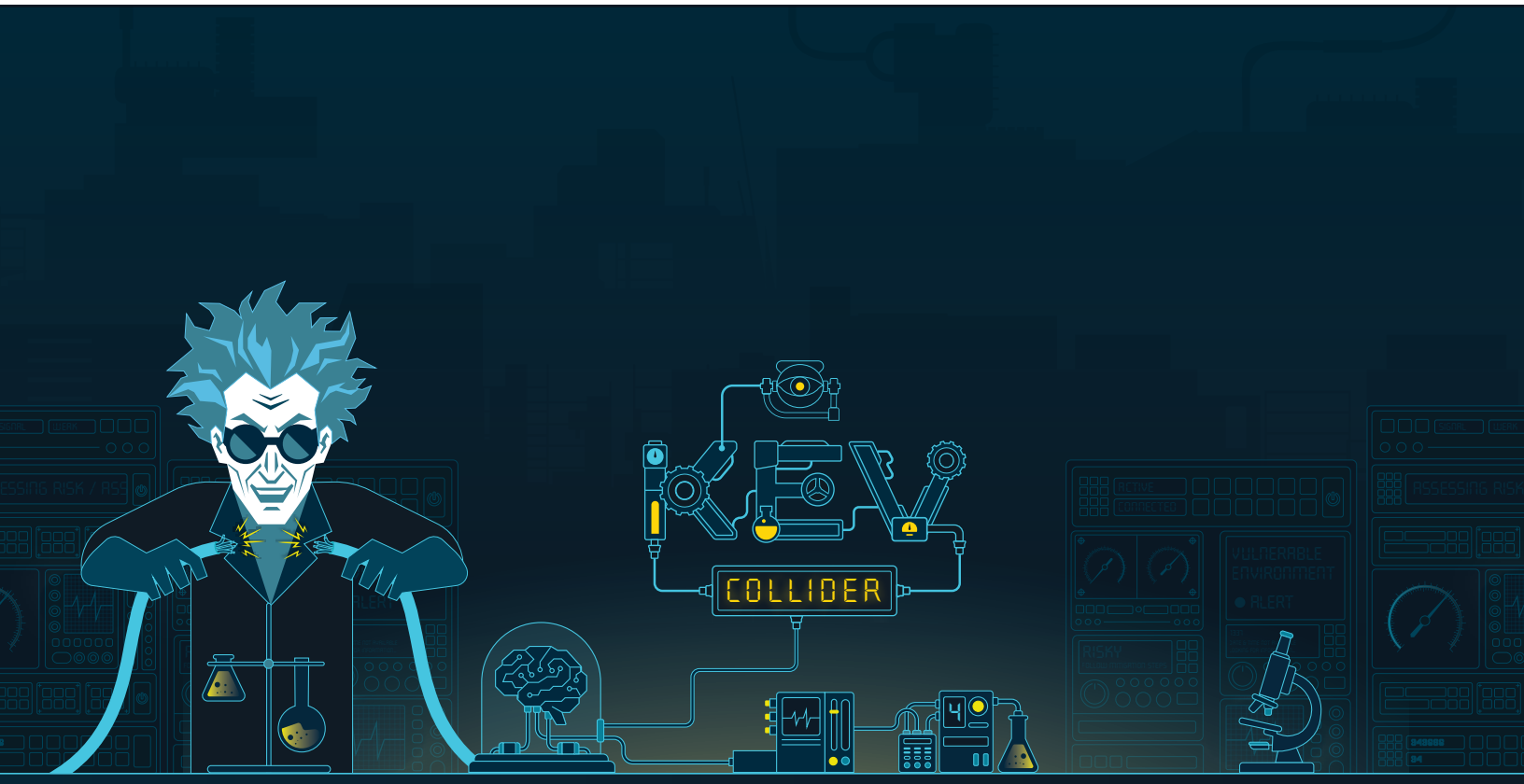
Table 12: Commodity exploits appearing within a week of KEVs where CVEs were published the same day as KEVs

This table builds a conspiracy theory, and illustrates analyst bias in action: correlations that invite narrative but do not, on their own, establish coordination or intent. According to this query, we can find exactly one vulnerability in the history of the KEV where we saw a KEV entry, a CVE record, and public exploit tooling all published on the same day. A credulous reader might infer that in this Fortinet case, CVE-2025-64446 (undoubtedly a numerologically significant CVE ID!) was being actively coordinated between Fortinet, the US government, and the hacker community. These kinds of coincidences certainly warrant investigation, but I wouldn't break out the string-and-snapshot corkboard without first starting that investigation.

Sidebar: cheating time

One possibility that we considered while collecting this data was the issue of time travel. When someone performs a git commit, the commit message is entirely dependent on the local system time. A clever, if underhanded, Metasploit or Nuclei committer could fairly easily forge a first commit in order to backdate a claim of being first to publish simply by setting their own clock back. No flux capacitance would be required.

Now, as someone who used to serve as the technical manager for Metasploit, I don't recall this ever happening, nor have I seen any Metasploit drama around this possibility since then. But it's conceivable that someone might try to cheat history, especially if they feel like they're in a race against another exploit developer for whatever reason. Such edits could be subtle, by mere minutes, or ridiculous, by predating the invention of the Internet. For this reason, exploit commit dates should be treated as approximate indicators of commoditization timing, not cryptographically secure time signatures.



Limitations and biases

This paper, like all other research papers, suffers from several forms of bias. Unlike many other research papers, we'd like to take a moment to address those biases in turn.

1. **Selection bias:** This paper is focused entirely on the CISA KEV, not VulnCheck KEV or other sources of known exploitation. This is because CISA KEV is free, in the public domain, central to cybersecurity regulations, and is itself designed with explicit inclusion criteria documented in BOD 22-01. The findings in this paper are not intended to be generalized across all exploited vulnerabilities.
2. **Reporting bias:** runZero, Inc. is the publisher of this paper, and is also in the business of providing solutions for exposure management, vulnerability management, and asset management, across IT, OT, cloud, and mobile platforms. As such, this may bias this paper toward seeking out attributes that are mechanically observable and actionable well before exploitation activity and incident response tends to happen.
3. **Measurement bias:** As discussed throughout the paper, scoring systems like CVSS and SSVC are (currently) largely human-driven, and thus, can reflect the biases of the humans providing those scores. The biases of individual CVSS scorers are explored in-depth in Shedding Light on CVSS Scoring Inconsistencies: A User-Centric Study on Evaluating Widespread Security Vulnerabilities[27] by Wunder et al. But in short, vendors of software tend to depress CVSS scores and understate CWE IDs, while researchers tend to inflate CVSS scores and over-index on hyper-specific CWE IDs. As LLMs increasingly take on the role of calculating CVSS, SSVC, and CWEs, we don't expect this measurement bias to disappear. Rather, we do expect it to be more subtle and more difficult to pinpoint.
4. **Tooling bias:** Metasploit and Nuclei are used as stand-ins for the general concept of exploit commoditization. When an exploit exists in one of these common toolsets, it's a strong signal of commoditization, but an exploit's absence from one of these tools should not be interpreted as "we're safe from this vulnerability." It is entirely possible, even common, to find exploits in regular use among criminals, intelligence agencies, and penetration testers.
5. **Temporal bias:** EPSS scores specifically are capable of changing over time, sometimes radically, as new evidence about particular exploitation indicators is ingested by the models used by EPSS. For more on tracking the bounciness of EPSS, please visit runZero's EPSS Pulse[28]. The values referenced in this paper are fixed as of its release in February 2026. Furthermore, KEV policies and operational practices do occasionally change over time. For example, in 2024, it was very uncommon to add a KEV on a Friday, but starting in the middle of 2025, it became more common, and thus, this specific marker has degraded as an unintentional signal of urgency.
6. **Analyst bias:** Several findings in this paper rely on the interpretation of correlated signals and proxy measures rather than direct mechanistic evidence. Inferences drawn from KEV deadlines, ATT&CK mappings, and the presence or absence of public exploitation tooling should be understood as suggestive indicators, not causal determinations.

[27] https://arxiv.org/pdf/2308.15259
[28] https://www.runzero.com/epss-pulse/

Conclusion: practical uses of KEVology

One of the recurring themes throughout this paper is that perfect vulnerability coverage is an increasingly unrealistic goal, particularly when organizations are constrained by finite tooling, staffing, or budget. This is even true when the focus is narrowed to merely the CISA KEV catalog. While the KEV is intentionally curated by experts, and far smaller than the universe of disclosed vulnerabilities, it has grown substantially since its introduction in 2021. Reasons for this expansion include greater cooperation between public and private sectors, better observance of exploitation across the industry, and, unfortunately, more active attacks that involve or are relevant to U.S. federal and critical infrastructure interests.

When the KEV contained fewer than 300 vulnerabilities, it was at least plausible for a well-resourced organization to claim near-total coverage using a single discovery and vulnerability management platform. At that scale, most KEVs affected traditional enterprise IT assets such as general purpose operating systems, widely deployed applications, and network services that were already well modeled by asset inventories and scanners. Coverage gaps certainly existed, but they were narrow and well understood.

That assumption no longer holds. As of this writing, the KEV contains nearly 1,500 vulnerabilities spanning a much broader range of technologies and environments, including traditional IT systems, operational technology (OT), mobile platforms, embedded devices, cloud services, and vendor-managed infrastructure. Many KEVs now affect assets that are difficult to inventory, difficult to scan, or difficult to patch using conventional enterprise tooling. In many cases, the vulnerable component may not even be directly owned or administered by the organization responsible for remediation.

This expansion in scope has important practical consequences. No single cybersecurity product today credibly claims 100% KEV coverage. This is not an indictment of any specific product, but rather, a recognition that different classes of assets require different discovery techniques, telemetry sources, and validation methods. Network-based discovery excels at finding exposed services, but will miss deeply embedded components visible only to locally authenticated users (and locally authenticated attackers). Authenticated scanners can provide rich vulnerability detail, but only where credentials and agents are deployable. Passive monitoring, cloud APIs, mobile device management, and vendor advisories all cover different slices of the problem space.

As a result, organizations attempting to operationalize KEV remediation should assume that a "total" solution will involve multiple products from multiple vendors, each contributing partial visibility. This is especially true in environments that include OT networks, managed service providers, or mobile bring-your-own device (BYOD) fleets. In these cases, the challenge is not merely identifying KEV-affected assets, but reconciling overlapping, incomplete, and sometimes contradictory data from disparate sources.

The enrichment signals explored throughout this paper — CVSS, EPSS, exploit availability, ATT&CK mappings, and time-based relationships — are best understood in this context. Their value lies not in delivering a single authoritative answer, but in helping practitioners reason about uncertainty and prioritize effort when full coverage is unattainable. For example, knowing that a KEV is associated with highly commoditized exploitation tooling may justify additional compensating controls when patching is delayed. Conversely, recognizing that a KEV affects a narrowly deployed technology with low observed exploit reuse may inform scheduling decisions when time is short. In other words, unless you are a state-employed veterinary care provider, you probably don't need to worry too much about CVE-2021-44207 (Acclaim USAHERDS, added to the KEV on December 23, 2024).

Importantly, these practical constraints do not negate the policy requirements associated with the KEV. For federal agencies and contractors subject to Binding Operational Directive 22-01, every KEV must ultimately be addressed. For the rest of us, the reality of modern enterprise environments means that addressing all KEVs simultaneously is often impossible. In practice, organizations must decide how to sequence remediation, where to apply detection and monitoring first, and when to escalate resource allocation to meet particularly aggressive deadlines.

In this sense, KEVology is less about prediction and more about decision support. They help translate an expanding and heterogeneous KEV catalog into actionable plans that reflect real operational and human limitations. As the KEV continues to grow in size and diversity, this kind of multi-signal reasoning is likely to become not just useful, but practically mandatory for any organization attempting to manage known exploited vulnerabilities at scale.

Smashing it all together: KEV Collider

All of the analysis in this paper is built on data that is publicly available and reproducible. To make this exploration easier and more fun, runZero has open sourced the datasets used in this research and launched a companion web application, KEV Collider, that allows readers to interact with the data directly.

KEV Collider exposes the same underlying KEV enrichment data used throughout this paper, including CVSS metrics, EPSS scores, exploit tooling indicators, ATT&CK mappings, and time-based attributes. The application is intentionally designed for experimentation: readers can sort, filter, and combine attributes in arbitrary ways to explore questions that go beyond the specific analyses presented here. Want to see which KEVs combine low EPSS scores with aggressive remediation deadlines? Curious which ATT&CK techniques appear most often alongside publicly available exploits? KEV Collider makes those kinds of ad hoc investigations trivial.

Importantly, KEV Collider is not a black box. All source JSON files used by the application, including the KEV records themselves and the derived enrichment fields, are available for public inspection. Readers are encouraged to download and validate the data, and build their own analyses. Nothing in this paper relies on proprietary scoring or unpublished datasets, and no conclusions require trusting an opaque analytical process.

One of the recurring arguments in this work is that vulnerability prioritization benefits from mixing and matching signals rather than relying on a single metric. KEV Collider is meant to support exactly that mindset. It is a sandbox in the classical sense, full of toy tools, models, and the occasional cat turd, for asking "what if?" questions about vulnerabilities listed on the KEV.

If you uncover an interesting pattern, edge case, or analytic approach using the data, we would love to hear from you[29]. This is doubly true if you find something that challenges or refines the conclusions in this paper. The KEV continues to evolve, and the most useful insights often come from many independent perspectives interrogating the same shared data.

[29] https://github.com/runZeroInc/kev-collider-data

Appendix: methodology

All analyses presented in this paper are derived from publicly available data sources and are intended to be independently verifiable. To support transparency and reproducibility, the datasets, schemas, and derived artifacts referenced throughout this work are published in a public GitHub repository.

This repository is actively maintained as part of the KEV Collider project. As a result, readers should expect that commit hashes, KEV counts, enrichment coverage, and derived outputs will evolve over time. Where reproducibility is required, specific commits and snapshot dates should be used.

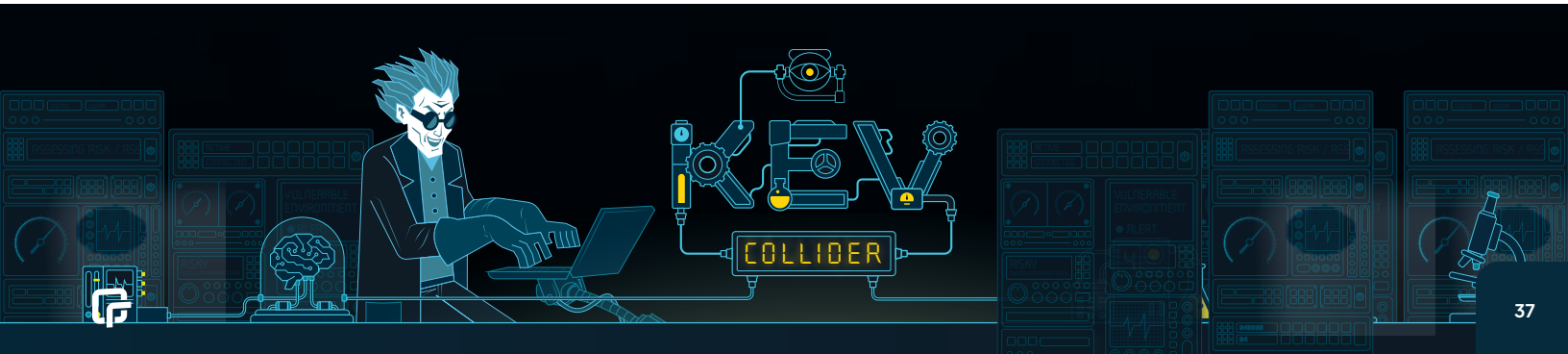
GitHub repositories

The development repository is organized as a lightweight orchestration layer that pulls in upstream data sources as git submodules, preserving their native structure and history. This approach minimizes data duplication and makes upstream provenance explicit. The commit hashes of the relevant git submodules used to generate the source JSON files that informed this paper are listed below.

```
https://github.com/cisagov/kev-data.git
5ea79fa5e4a7805dc0fb988000264092c3155d9d (develop)

https://github.com/CVEProject/cvelistV5.git
f0d57590828ac0b1cc67b8dda52544a61ae0824d (main)

https://github.com/rapid7/metasploit-framework.git
d8bef9bd7b7298eb1aab27d3a86be38601e7e6a8 (master)

https://github.com/projectdiscovery/nuclei-templates.git
74c6ba4beed70188a407a4ee08e86b094599c482 (main)

https://github.com/center-for-threat-informed-defense/mappings-explorer
a14e25183a23bc26c0c4bada27ee5ff0eb9710d7 (main)
```

Commit hashes of the relevant GitHub repositories

The resultant JSON files that are built against all this data are published at https://github.com/runZeroInc/kev-collider-data/, and are the source files for the KEV Collider web application. Issues in the data, or the web application, should be filed there.



EPSS data

One notable exception to this GitHub-first approach is EPSS, which does not appear to be available authoritatively from GitHub. Instead, EPSS scores and percentiles for "today" and "yesterday" are pulled via the public API at `https://epss.empiricalsecurity.com/epss_scores-{date}.csv.gz`

Jupyter Notebook

Finally, a Jupyter Notebook containing several exploratory and analytical approaches to the JSON datasets is published at vulnerability-mechanics, alongside the JSON data files. In an act that will undoubtedly distress data scientists everywhere, this notebook uses the Ruby 3 (`iruby`) kernel rather than the far more common, and objectively more sensible, Python 3 (`ipython`) kernel.

This was a deliberate choice, driven not by best practices or even runZero company style, but by authorial habit: the author is, regrettably, still a systems-level Rubyist first. Readers are welcome to become contributors and translate interim analysis into Python, R, or anything else that brings them joy; the standalone JSON data and schemas are provided precisely so that no one is required to endure Ruby idiom unless they truly want to. Stay crunchy, bacon.

Appendix 1: Metasploitable KEVs

The CVEs listed in this section are known to be connected to at least one KEV-listed vulnerability. This list is likely to fall out of date quickly, as public exploit development is ongoing.

CVE-2005-2773	CVE-2010-3765	CVE-2012-4969	CVE-2013-7331	CVE-2015-0313
CVE-2007-3010	CVE-2010-3904	CVE-2012-5076	CVE-2014-0160	CVE-2015-1130
CVE-2007-5659	CVE-2010-3962	CVE-2013-0074	CVE-2014-0322	CVE-2015-1187
CVE-2008-2992	CVE-2010-4344	CVE-2013-0422	CVE-2014-0497	CVE-2015-1427
CVE-2009-0927	CVE-2010-4345	CVE-2013-0431	CVE-2014-100005	CVE-2015-1635
CVE-2009-1151	CVE-2011-0609	CVE-2013-0632	CVE-2014-1761	CVE-2015-1701
CVE-2009-3129	CVE-2011-0611	CVE-2013-1347	CVE-2014-1812	CVE-2015-2051
CVE-2009-3953	CVE-2011-2462	CVE-2013-1690	CVE-2014-3120	CVE-2015-2426
CVE-2009-3960	CVE-2011-3402	CVE-2013-2251	CVE-2014-3153	CVE-2015-3035
CVE-2009-4324	CVE-2011-3544	CVE-2013-2423	CVE-2014-4113	CVE-2015-3043
CVE-2010-0188	CVE-2012-0158	CVE-2013-2465	CVE-2014-4114	CVE-2015-3113
CVE-2010-0232	CVE-2012-0391	CVE-2013-2551	CVE-2014-4404	CVE-2015-4495
CVE-2010-0738	CVE-2012-0507	CVE-2013-3163	CVE-2014-6271	CVE-2015-4852
CVE-2010-0840	CVE-2012-0754	CVE-2013-3346	CVE-2014-6278	CVE-2015-5119
CVE-2010-1297	CVE-2012-1535	CVE-2013-3660	CVE-2014-6287	CVE-2015-5122
CVE-2010-1428	CVE-2012-1723	CVE-2013-3893	CVE-2014-6324	CVE-2015-7450
CVE-2010-1871	CVE-2012-1823	CVE-2013-3896	CVE-2014-6332	CVE-2015-7755
CVE-2010-2568	CVE-2012-1889	CVE-2013-3897	CVE-2014-6352	CVE-2016-0040
CVE-2010-2861	CVE-2012-3152	CVE-2013-3906	CVE-2014-8361	CVE-2016-0099
CVE-2010-2883	CVE-2012-4681	CVE-2013-3918	CVE-2015-0016	CVE-2016-0189
CVE-2010-3333	CVE-2012-4792	CVE-2013-6282	CVE-2015-0311	CVE-2016-0752

CVE-2016-10033
CVE-2016-10174
CVE-2016-11021
CVE-2016-1555
CVE-2016-20017
CVE-2016-3088
CVE-2016-3235
CVE-2016-4117
CVE-2016-4437
CVE-2016-4655
CVE-2016-4656
CVE-2016-4657
CVE-2016-6277
CVE-2016-6366
CVE-2016-6415
CVE-2016-9079
CVE-2017-0143
CVE-2017-0144
CVE-2017-0145
CVE-2017-0146
CVE-2017-0147
CVE-2017-0148
CVE-2017-0199
CVE-2017-1000353
CVE-2017-1000486
CVE-2017-10271
CVE-2017-11317
CVE-2017-11882
CVE-2017-12149
CVE-2017-12617
CVE-2017-15944
CVE-2017-16651
CVE-2017-17562
CVE-2017-18368
CVE-2017-3881
CVE-2017-5521
CVE-2017-5638
CVE-2017-5689
CVE-2017-6334
CVE-2017-7269
CVE-2017-7494
CVE-2017-8464
CVE-2017-9791
CVE-2017-9805
CVE-2017-9822
CVE-2018-0296
CVE-2018-0824
CVE-2018-1000861
CVE-2018-11138
CVE-2018-11776
CVE-2018-14847
CVE-2018-14933
CVE-2018-15133
CVE-2018-15811
CVE-2018-15961
CVE-2018-17463
CVE-2018-18325
CVE-2018-20062
CVE-2018-20250
CVE-2018-2628
CVE-2018-7600
CVE-2018-8120
CVE-2018-8440
CVE-2018-8453
CVE-2018-9276
CVE-2019-0708
CVE-2019-0808
CVE-2019-0841
CVE-2019-1003029
CVE-2019-10068
CVE-2019-10149
CVE-2019-11043
CVE-2019-11510
CVE-2019-11539
CVE-2019-11580
CVE-2019-1322
CVE-2019-13272
CVE-2019-1405
CVE-2019-1458
CVE-2019-15107
CVE-2019-15752
CVE-2019-15949
CVE-2019-16278
CVE-2019-1652
CVE-2019-1653
CVE-2019-16759
CVE-2019-17558
CVE-2019-17621
CVE-2019-18935
CVE-2019-18988
CVE-2019-19781
CVE-2019-20085
CVE-2019-2215
CVE-2019-2725
CVE-2019-3010
CVE-2019-3396
CVE-2019-3929
CVE-2019-4716
CVE-2019-5418
CVE-2019-5786
CVE-2019-5825
CVE-2019-6340
CVE-2019-7192
CVE-2019-7194
CVE-2019-7195
CVE-2019-7256
CVE-2019-7609
CVE-2019-8394
CVE-2019-9082
CVE-2019-9621
CVE-2019-9670
CVE-2020-0618
CVE-2020-0646
CVE-2020-0688
CVE-2020-0787
CVE-2020-0796
CVE-2020-10189
CVE-2020-10199
CVE-2020-1054
CVE-2020-1147
CVE-2020-11651
CVE-2020-11652
CVE-2020-11738
CVE-2020-11978
CVE-2020-13927
CVE-2020-1472
CVE-2020-14750
CVE-2020-14871
CVE-2020-14882
CVE-2020-14883
CVE-2020-15505
CVE-2020-16846
CVE-2020-17496
CVE-2020-17519
CVE-2020-17530
CVE-2020-1938
CVE-2020-25213
CVE-2020-25223
CVE-2020-2555
CVE-2020-2883
CVE-2020-28949
CVE-2020-3153
CVE-2020-3433
CVE-2020-3950
CVE-2020-3952
CVE-2020-4427
CVE-2020-4428
CVE-2020-5410
CVE-2020-5722
CVE-2020-5741
CVE-2020-5847
CVE-2020-5849
CVE-2020-5902
CVE-2020-6207
CVE-2020-6287
CVE-2020-6418
CVE-2020-7247
CVE-2020-7961
CVE-2020-8260
CVE-2020-8644
CVE-2020-8655
CVE-2020-8657
CVE-2020-8816
CVE-2020-9934
CVE-2021-1497
CVE-2021-1498
CVE-2021-1675
CVE-2021-1732
CVE-2021-21220
CVE-2021-21551
CVE-2021-21972
CVE-2021-21975
CVE-2021-21985
CVE-2021-22005
CVE-2021-22204
CVE-2021-22205
CVE-2021-22502
CVE-2021-22555
CVE-2021-22986
CVE-2021-25296
CVE-2021-25297
CVE-2021-25298
CVE-2021-26084
CVE-2021-26855
CVE-2021-27065
CVE-2021-27876
CVE-2021-27877
CVE-2021-27878
CVE-2021-30657
CVE-2021-31166
CVE-2021-31207
CVE-2021-3129
CVE-2021-3156
CVE-2021-34473
CVE-2021-34523
CVE-2021-34527
CVE-2021-3493
CVE-2021-35464
CVE-2021-35587
CVE-2021-3560
CVE-2021-36260
CVE-2021-36934
CVE-2021-36942
CVE-2021-38647
CVE-2021-38648
CVE-2021-39144
CVE-2021-4034
CVE-2021-40444
CVE-2021-40449
CVE-2021-40539
CVE-2021-41773
CVE-2021-42013
CVE-2021-42237
CVE-2021-42258
CVE-2021-42321
CVE-2021-43798
CVE-2021-44077
CVE-2021-44228
CVE-2021-44529
CVE-2021-45046
CVE-2022-0543
CVE-2022-0847
CVE-2022-1388
CVE-2022-20699
CVE-2022-21587
CVE-2022-21882
CVE-2022-21999
CVE-2022-22947
CVE-2022-22948
CVE-2022-22954
CVE-2022-22960
CVE-2022-22963
CVE-2022-22965
CVE-2022-24112
CVE-2022-24706
CVE-2022-24990
CVE-2022-26134
CVE-2022-26318
CVE-2022-26352
CVE-2022-26904
CVE-2022-26923
CVE-2022-27925
CVE-2022-28810
CVE-2022-29464
CVE-2022-30190
CVE-2022-30333
CVE-2022-30525
CVE-2022-33891
CVE-2022-35405
CVE-2022-35914
CVE-2022-36804
CVE-2022-37042
CVE-2022-40684
CVE-2022-41040
CVE-2022-41082
CVE-2022-41352
CVE-2022-43769
CVE-2022-43939
CVE-2022-44877
CVE-2022-46169
CVE-2022-47966
CVE-2023-0386
CVE-2023-0669
CVE-2023-20198
CVE-2023-20273
CVE-2023-20887
CVE-2023-21839
CVE-2023-22515
CVE-2023-22518
CVE-2023-22527
CVE-2023-22952
CVE-2023-23752
CVE-2023-24955
CVE-2023-26360
CVE-2023-27350
CVE-2023-27524
CVE-2023-28252
CVE-2023-28432
CVE-2023-28771
CVE-2023-29357
CVE-2023-32315
CVE-2023-33246
CVE-2023-34362
CVE-2023-3519
CVE-2023-36845
CVE-2023-36874
CVE-2023-38035
CVE-2023-38831
CVE-2023-40044
CVE-2023-42793
CVE-2023-43208
CVE-2023-45249
CVE-2023-46604
CVE-2023-46747
CVE-2023-46805
CVE-2023-48788
CVE-2023-49103
CVE-2023-4911
CVE-2023-4966
CVE-2023-7028

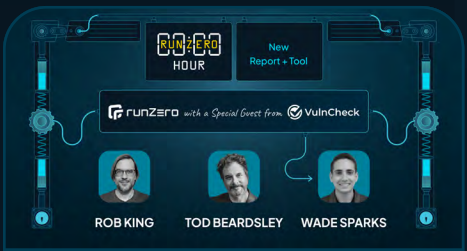
CVE-2024-0012
CVE-2024-11680
CVE-2024-1212
CVE-2024-12356
CVE-2024-1709
CVE-2024-20767
CVE-2024-21887
CVE-2024-21893
CVE-2024-23692
CVE-2024-23897
CVE-2024-24919
CVE-2024-27198
CVE-2024-27348
CVE-2024-28987
CVE-2024-28995
CVE-2024-29824
CVE-2024-32113
CVE-2024-3400
CVE-2024-34102
CVE-2024-35250
CVE-2024-36401
CVE-2024-38856
CVE-2024-4040
CVE-2024-4358
CVE-2024-4577
CVE-2024-47575
CVE-2024-51378
CVE-2024-51567
CVE-2024-55956
CVE-2024-56145
CVE-2024-57727
CVE-2024-5910
CVE-2024-6670
CVE-2024-7593
CVE-2024-9474
CVE-2025-14847
CVE-2025-22457
CVE-2025-24016
CVE-2025-24813
CVE-2025-24893
CVE-2025-30406
CVE-2025-32433
CVE-2025-32463
CVE-2025-3248
CVE-2025-33053
CVE-2025-37164
CVE-2025-4427
CVE-2025-4428
CVE-2025-47812
CVE-2025-49704
CVE-2025-49706
CVE-2025-53770
CVE-2025-54236
CVE-2025-55182
CVE-2025-57819
CVE-2025-58034
CVE-2025-58360
CVE-2025-59287
CVE-2025-64446

Appendix 2: Nucleiable KEVs

The CVEs listed in this section are known to be connected to at least one KEV-listed vulnerability. This list is likely to fall out of date quickly, as public exploit development is ongoing.

CVE-2007-3010
CVE-2009-1151
CVE-2010-2861
CVE-2012-1823
CVE-2013-2251
CVE-2014-3120
CVE-2014-6271
CVE-2014-6287
CVE-2015-1427
CVE-2015-1635
CVE-2015-3035
CVE-2015-7450
CVE-2016-10033
CVE-2016-1555
CVE-2016-3088
CVE-2016-4437
CVE-2016-6277
CVE-2016-8735
CVE-2017-1000353
CVE-2017-1000486
CVE-2017-10271
CVE-2017-12149
CVE-2017-12615
CVE-2017-12617
CVE-2017-12637
CVE-2017-15944
CVE-2017-17562
CVE-2017-18362
CVE-2017-3506
CVE-2017-3881
CVE-2017-5521
CVE-2017-5638
CVE-2017-5689
CVE-2017-7269
CVE-2017-9791
CVE-2017-9805
CVE-2017-9822
CVE-2018-0171
CVE-2018-0296
CVE-2018-1000861
CVE-2018-10562
CVE-2018-11138
CVE-2018-11776
CVE-2018-1273
CVE-2018-13379
CVE-2018-14933
CVE-2018-15811
CVE-2018-15961
CVE-2018-18325
CVE-2018-18809
CVE-2018-19410
CVE-2018-20062
CVE-2018-2628
CVE-2018-6530
CVE-2018-6882
CVE-2018-6961
CVE-2018-7600
CVE-2018-7602
CVE-2018-7841
CVE-2019-0193
CVE-2019-0604
CVE-2019-10068
CVE-2019-10758
CVE-2019-11510
CVE-2019-11580
CVE-2019-11581
CVE-2019-12989
CVE-2019-15107
CVE-2019-16057
CVE-2019-16278
CVE-2019-1653
CVE-2019-16759
CVE-2019-16920
CVE-2019-17558
CVE-2019-19781
CVE-2019-20085
CVE-2019-2616
CVE-2019-2725
CVE-2019-3396
CVE-2019-3398
CVE-2019-3929
CVE-2019-4716
CVE-2019-5418
CVE-2019-5544
CVE-2019-5591
CVE-2019-6340
CVE-2019-7192
CVE-2019-7194
CVE-2019-7195
CVE-2019-7238
CVE-2019-7256
CVE-2019-7481
CVE-2019-7609
CVE-2019-9621
CVE-2019-9670
CVE-2019-9874
CVE-2019-9978
CVE-2020-0618
CVE-2020-0646
CVE-2020-0796
CVE-2020-10148
CVE-2020-10189
CVE-2020-10199
CVE-2020-10987
CVE-2020-11738
CVE-2020-11978
CVE-2020-12641
CVE-2020-13927
CVE-2020-14644
CVE-2020-14750
CVE-2020-14864
CVE-2020-14882
CVE-2020-14883
CVE-2020-15415
CVE-2020-15505
CVE-2020-16846
CVE-2020-17463
CVE-2020-17496
CVE-2020-17519
CVE-2020-17530
CVE-2020-1938
CVE-2020-1956
CVE-2020-25078
CVE-2020-25213
CVE-2020-25223
CVE-2020-25506
CVE-2020-2551
CVE-2020-26919
CVE-2020-2883
CVE-2020-29583
CVE-2020-3452
CVE-2020-3580
CVE-2020-3952
CVE-2020-4427
CVE-2020-5410
CVE-2020-5847
CVE-2020-5902
CVE-2020-6207
CVE-2020-6287
CVE-2020-7247
CVE-2020-7961
CVE-2020-8193
CVE-2020-8515
CVE-2020-8644
CVE-2020-8657
CVE-2020-9054
CVE-2021-1497
CVE-2021-1498
CVE-2021-20021
CVE-2021-20038
CVE-2021-20090
CVE-2021-20123
CVE-2021-20124
CVE-2021-21311
CVE-2021-21315
CVE-2021-21972
CVE-2021-21973
CVE-2021-21975
CVE-2021-21985
CVE-2021-22005
CVE-2021-22205
CVE-2021-22502
CVE-2021-22986
CVE-2021-25296
CVE-2021-25297
CVE-2021-25298
CVE-2021-26084
CVE-2021-26085
CVE-2021-26086
CVE-2021-26855
CVE-2021-27561
CVE-2021-27877
CVE-2021-28799
CVE-2021-30116
CVE-2021-3129
CVE-2021-3156
CVE-2021-31755
CVE-2021-32030
CVE-2021-32648
CVE-2021-33044
CVE-2021-33045
CVE-2021-33766
CVE-2021-34473
CVE-2021-35395
CVE-2021-35464

CVE-2021-35587
CVE-2021-36260
CVE-2021-36380
CVE-2021-37415
CVE-2021-38647
CVE-2021-39144
CVE-2021-39226
CVE-2021-40438
CVE-2021-40539
CVE-2021-40655
CVE-2021-40870
CVE-2021-41277
CVE-2021-41773
CVE-2021-42013
CVE-2021-42237
CVE-2021-42258
CVE-2021-43798
CVE-2021-44077
CVE-2021-44228
CVE-2021-44515
CVE-2021-44529
CVE-2021-45046
CVE-2021-45382
CVE-2022-0543
CVE-2022-1040
CVE-2022-1388
CVE-2022-21587
CVE-2022-22536
CVE-2022-22947
CVE-2022-22954
CVE-2022-22963
CVE-2022-22965
CVE-2022-23131
CVE-2022-23134
CVE-2022-24086
CVE-2022-24112
CVE-2022-24682
CVE-2022-24706
CVE-2022-24816
CVE-2022-24990
CVE-2022-26134
CVE-2022-26138
CVE-2022-26143
CVE-2022-26352
CVE-2022-27593
CVE-2022-27924
CVE-2022-27926
CVE-2022-29303
CVE-2022-29464
CVE-2022-30525
CVE-2022-33891
CVE-2022-35405
CVE-2022-35914
CVE-2022-36537
CVE-2022-36804
CVE-2022-37042
CVE-2022-40684
CVE-2022-41352
CVE-2022-42475
CVE-2022-43769
CVE-2022-43939
CVE-2022-44877
CVE-2022-46169
CVE-2022-47966
CVE-2022-47986
CVE-2023-0669
CVE-2023-1389
CVE-2023-1671
CVE-2023-20198
CVE-2023-20887
CVE-2023-21839
CVE-2023-22515
CVE-2023-22518
CVE-2023-22527
CVE-2023-22952
CVE-2023-23752
CVE-2023-24489
CVE-2023-25280
CVE-2023-25717
CVE-2023-26360
CVE-2023-27350
CVE-2023-27524
CVE-2023-28432
CVE-2023-29298
CVE-2023-29300
CVE-2023-29357
CVE-2023-32315
CVE-2023-33246
CVE-2023-34048
CVE-2023-34192
CVE-2023-34362
CVE-2023-35078
CVE-2023-35082
CVE-2023-3519
CVE-2023-36844
CVE-2023-36845
CVE-2023-37580
CVE-2023-38035
CVE-2023-38203
CVE-2023-38205
CVE-2023-38950
CVE-2023-40044
CVE-2023-41265
CVE-2023-41266
CVE-2023-41763
CVE-2023-42793
CVE-2023-43208
CVE-2023-45249
CVE-2023-46604
CVE-2023-46747
CVE-2023-46805
CVE-2023-47246
CVE-2023-48788
CVE-2023-49103
CVE-2023-4911
CVE-2023-4966
CVE-2023-6549
CVE-2023-7028
CVE-2024-0012
CVE-2024-11680
CVE-2024-1212
CVE-2024-12356
CVE-2024-12987
CVE-2024-13159
CVE-2024-13160
CVE-2024-13161
CVE-2024-1709
CVE-2024-20439
CVE-2024-20767
CVE-2024-21887
CVE-2024-21893
CVE-2024-23692
CVE-2024-23897
CVE-2024-24919
CVE-2024-27198
CVE-2024-27348
CVE-2024-27443
CVE-2024-28987
CVE-2024-28995
CVE-2024-29059
CVE-2024-29824
CVE-2024-32113
CVE-2024-3272
CVE-2024-3273
CVE-2024-3400
CVE-2024-34102
CVE-2024-38475
CVE-2024-4040
CVE-2024-40711
CVE-2024-41713
CVE-2024-42009
CVE-2024-45195
CVE-2024-4577
CVE-2024-47575
CVE-2024-4885
CVE-2024-50603
CVE-2024-50623
CVE-2024-51378
CVE-2024-51567
CVE-2024-53704
CVE-2024-55591
CVE-2024-55956
CVE-2024-56145
CVE-2024-57727
CVE-2024-58136
CVE-2024-5910
CVE-2024-6670
CVE-2024-7593
CVE-2024-8963
CVE-2024-9463
CVE-2024-9465
CVE-2024-9474
CVE-2025-0108
CVE-2025-0282
CVE-2025-11371
CVE-2025-12480
CVE-2025-20281
CVE-2025-20362
CVE-2025-22457
CVE-2025-24016
CVE-2025-24813
CVE-2025-24893
CVE-2025-25257
CVE-2025-2746
CVE-2025-2747
CVE-2025-2775
CVE-2025-2776
CVE-2025-27915
CVE-2025-30406
CVE-2025-31161
CVE-2025-31324
CVE-2025-32433
CVE-2025-3248
CVE-2025-34028
CVE-2025-4008
CVE-2025-4427
CVE-2025-4632
CVE-2025-47812
CVE-2025-48703
CVE-2025-49706
CVE-2025-5086
CVE-2025-53770
CVE-2025-54309
CVE-2025-55182
CVE-2025-57819
CVE-2025-58360
CVE-2025-59287
CVE-2025-61757
CVE-2025-61884
CVE-2025-6204
CVE-2025-6205
CVE-2025-64446
CVE-2025-9242



KEVology 101: Observing exploit trajectories in the KEV Collider

Tod Beardsley, Rob King, and guest Wade Sparks (CISA & VulnCheck KEV veteran) teach the science of KEVology.

Watch the Episode

Determine true impact for your organization with KEV Collider

KEV Collider layers CISA's KEV catalog with daily open-source metadata to turn theoretical risk into actionable intelligence.

Launch KEV Collider

Experiment with the runZero Platform

Uncover the unknowns with runZero – start your free trial in minutes. Fully functional up to 100,000 assets.

Try runZero Free

runZero provides a single source of truth for exposure management across your total attack surface: internal, external, IT, OT, IoT, mobile, and cloud.

Providing the most complete and accurate visibility into every asset and exposure, runZero helps you mitigate risks faster, meet compliance requirements, and ensure you continuously discover the assets and exposures that others miss.