



Mystical Vulnerabilities:

Exploring the Oddities in Cybersecurity

```
$ whoami
```

```
Massimo Morello - Associate Information  
Security Specialist @ Deutsche Börse Group
```

```
$ whoami --version
```

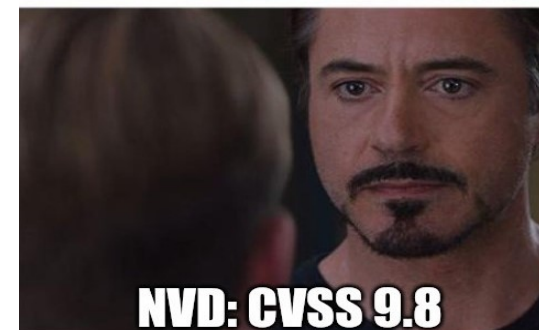
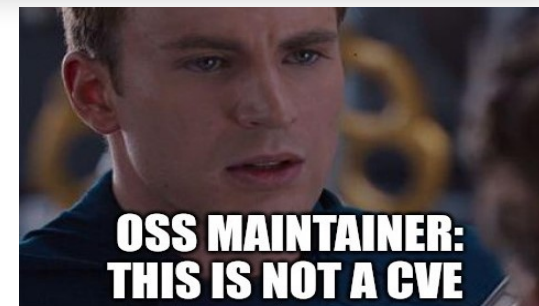
```
27.6
```

What Makes a CVE Mystical?

“**Mystical**” CVEs: vulnerabilities that carry disputed claims by their vendors, against the security researchers who filed them. They always result in debates due to conflicting interpretations, highlighting the complexity of proper categorization.

Additional criterion is the rarity of the vulnerability, based on:

- Unconventional methods of exploitation
- Unpredictable impact



Research Relevance



Purpose of the exploration

- Foster collaboration, knowledge sharing, and discussions among security professionals, researchers, and vendors (e.g. information exchange forums)
- Explore dynamics between security researchers, vendors, and vuln. databases in handling such CVEs
- Develop strategies for evaluating validity and significance of these CVEs in Risk Management frameworks



Why bothering?



These CVEs will show it's not hard to challenge the traditional notions of vulnerability and severity.

They demand a more nuanced and granular evaluation approach.

Databases Inaccuracy: Some Numbers

Guo et al [1] investigated 133.639 vulnerability reports in the CVE Database over the past 20 years: 56% miss the vulnerability type, 85% the root cause, 38% the attack vector, and 28% the attacker type

Giannakopoulos and Konstantinos [2]: nearly 1/3 of all CVEs don't have any CWEs assigned to them or have generic CWEs ("NVD-CWE-Other") that does not offer any information about that vulnerability, to then be linked back to a threat

VulnCheck researchers [3] took a look XSS and CSRF vulnerabilities in the NVD. XSS and CSRF always require User Interaction (present in the CVSS vector). Result: in the NVD the UI:R is often forgotten, (111 times for XSS vulnerabilities, and 59 for CSRF)

Spring et al. [4]: the CVSS formula is unjustified. Plus, many compliance bodies wrongly recommend to use it as a risk score:

- Federal Civilian Departments and Agencies via NIST guidance (*e.g. Special Publications 800-115, page 7-4 and 800-40r3, page 4*)
 - PCI Data Security Standard (in the regulation on *Approved Scanning Vendors v3*)
-

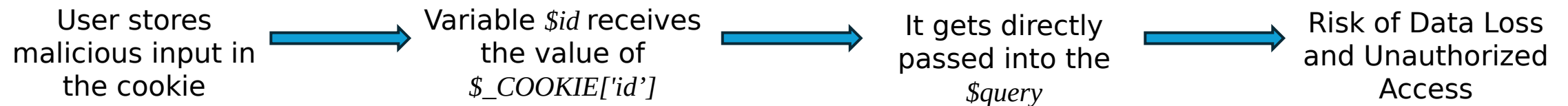
Case Study 1: CVE-2023-39848

Non-vulnerability (bad intentions)



Affected Product: DVWA 1.0

SQL injection vulnerability via the *id* parameter at *blind\source\high.php*



Why unusual?

DVWA stands for “*Damn Vulnerable Web App*” → security flaws intentionally implemented for learning purposes.

Aid for security professionals (mainly web developers) and students to test their skills and tools in a legal environment.

Status

Initially disputed, got **rejected** later on

Case Study 2: CVE-2020-21469

Non-vulnerability (good intentions)



PostgreSQL

PostgreSQL 12.2 allows attackers to cause a DoS via repeatedly sending *SIGHUP* signals

Why unusual?

The CVE was filed without the prior knowledge of the PostgreSQL Security Team, and got immediately disputed by them → untrusted users cannot send *SIGHUP* signals, i.e. they need to have an account that is explicitly granted elevated privileges, including:

- A PostgreSQL superuser (*postgres*)
- A user that was granted permission to execute *pg_reload_conf* by a PostgreSQL superuser
- Access to a privileged Operating System user (the *postgres* account or the root account)

Status

Initially disputed, got definitely **rejected**

Case Study 3: CVE-2014-0160

Vulnerability, yes, but unpredictable



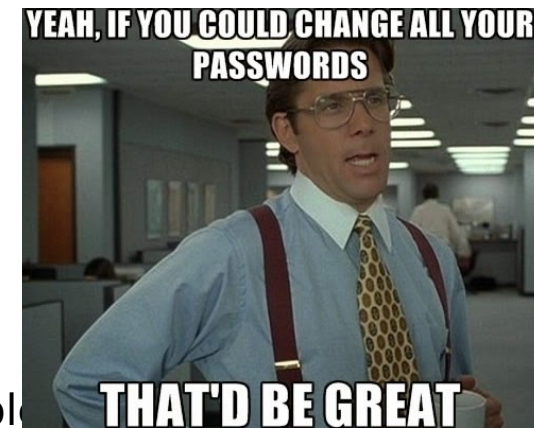
Known as **Heartbleed**, OpenSSL allowed attackers to leak sensitive data from the memory of a targeted system

Why unusual?

- 5.0 score, but had an unfathomable impact: between 30-70% of the Internet
- Then moved to a 7.5 score
- Large amount of private keys exposed to the Internet for a long time
- Ease of exploitation (Metasploit module) and attacks leaving no trace
- Unconventional exploitation method, leveraging TLS Heartbeat extension, enabled by default

Status

In November 2020, SANS counted more than 200,000 machines to be still vulnerable





Solutions and Best Practices

To sift through the noise

Goal → Contribute to a stronger filter-oriented mindset

Means → Alternatives to the traditional solutions + Complementary solutions

Breakdown → Best practices placed into four categories: Culture, Scoring Systems, Decision-making, and Sources/Databases



Culture

Becoming a CNA (CVE Numbering Authority)

- “Having a foot in the door”: vendors able to control their issues and reject non-vulnerabilities (CVEs for their products are immediately passed to them);
- Recently, also Linux kernel joined the CNA program. More Open Source projects are doing the same.

Overall new culture for enhancing the validation, reporting, and resolution:

- Incentivize people to validate and fix vulnerabilities [6], and not just to find them. Result: improved quality and completeness of new submissions.
 - Army metaphor: Bug Bounties recruited a global army of bug finders. We need also a complementary global army of **Bug Validators**.
 - Contrasting Bug Bounties [7]
-

Scoring System

We almost need a scoring system for the scoring system



EPSS (Exploit Prediction Scoring System): ML model estimating (daily) the likelihood that a CVE gets exploited ITW (score 0-1).

- Estimation on: attack complexity, likelihood of discovery, potential impact, current threat landscape.
- Meticulous maintainers: they seek for exploitation activity ITW before and after a Metasploit module is added.
- Not an alternative to the CVSS, but complementary solution (additional capabilities).

ESS (Exploit Scoring System): risk prioritization scoring system leveraging real-time monitoring and dynamic scoring.

- Early source of truth to evaluate which risks to prioritize **before** an incident occurs.
- AI + LLM scan descriptions of newly released CVEs, compare them to previously published ones. Result:
 - *EAP (Exploit Availability Probability)*: likelihood that an exploit will be made publicly available
 - *EUP (Exploit Usage Probability)*: likelihood that an attacker will actually use an exploit to execute a large-scale attack.
 - Example: CVE with high EAP and a low EUP = an exploit exists.

Scoring System



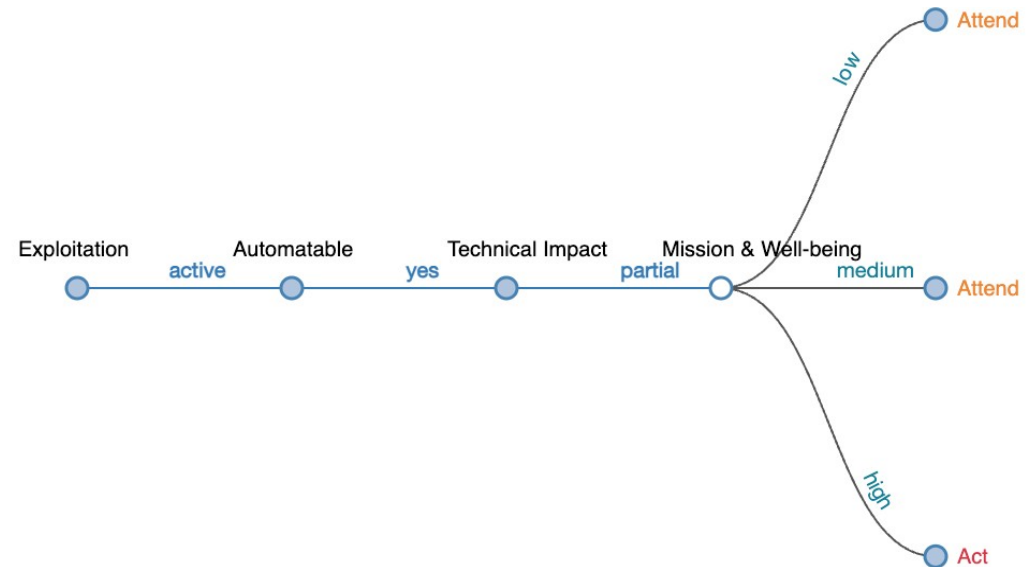
Tenable VPR (Vulnerability priority rating): score considering the impact on CIA following exploitation of a vulnerability, and the threat level (both recent and potential future threat activities)

- Calculated on: public PoC research, reports of exploitation on social media, emergence of exploit code in exploit kits, references to exploitation on the dark web and forums, presence of malware hashes ITW
- Goal: answering *“what is the appropriate level of near-term threat for a vulnerability based on the latest available data?”*
- VPR showed to be more efficient than CVSSv3 at predicting vulnerabilities under threat:
 - For CVEs with emerging exploitations ITW, remediating the top 1,500 VPR scores is as effective as remediating the top 33,000 CVSSv3 scores: 22 times more efficient;
 - 54% VPR Critical vulnerabilities have public exploits available, compared to 15% of CVSS Critical vulnerabilities;
 - VPR is better at identifying CVEs with escalating exploit code maturity, rating 4/8 vulnerabilities as Critical compared to two for CVSS in the 28 days following VPR scoring.

Decision-making

SSVC (Stakeholder-Specific Vulnerability Categorization): methodology for prioritizing vulnerabilities based on the specific needs of different stakeholders involved in the VM process.

- **Cost-benefit analysis** instead of numerical scores, employs decision trees for decision-making, but the representation is open to customization
- Can be **integrated** with other methodologies, like scores (e.g., using their vectors for technical impact) and EPSS (to inform decisions about exploitation likelihood)
- **CISA** implemented SSVC [8] in 2020, creating a decision tree with four possible outcomes: Track, Track*, Attend, and Act, based on factors like exploitation status, technical impact, automation potential, mission prevalence, and public well-being



Sources and Databases



GSD (Global Security Database): project built by the Cloud Security Alliance with the goal of creating an Open Source community for improving the quality and usability of vulnerability databases.

OSV.dev initiative: vulnerability database consisting of user-friendly data format. Maps precisely to Open Source versioning schemes, and a reference infrastructure that ingests vulnerability data from databases supporting the OVS schema.

It overcomes the difficulty of matching CVEs to package names and set of versions in an automated way.

ID	Packages	Summary	Published ↓	Attributes
CVE-2020-10370	github.com/rpi-distro/bluez-firmware	See record for full details	10 hours ago	Fix available
DLA-3949-1	Debian:11/ruby-saml	ruby-saml - security update	10 hours ago	Fix available
USN-7099-1	Ubuntu:20.04:LTS/openjdk-21 Ubuntu:22.04:LTS/openjdk-21 Ubuntu:24.10/openjdk-21 Ubuntu:24.04:LTS/openjdk-21	openjdk-21 vulnerabilities	10 hours ago	Fix available

Sources and Databases



GUAC (Graph for Understanding Artifact Composition): Open Source project launched in 2022 by Google with the goal of changing how the industry perceives software Supply Chains

The screenshot displays the GUAC Visualizer interface. At the top, it shows the title '*Experimental GUAC Visualizer v0.2.0' and navigation links for 'GitHub', 'GUAC Docs', and 'Community'. Below the title, there are four search filters: 'Package Type' (set to 'golang'), 'Package Namespace' (set to 'cloud.google.com'), 'Package Name' (set to 'go'), and 'Package Version' (set to 'v0.100.2[]'). There are 'Back', 'Forward', and 'Reset' buttons below these filters. A tip states: 'TIP: Use click and scroll to adjust graph. Right clicking a node displays more information.' On the left, there are 'Highlight Nodes' controls with toggle switches for 'Artifacts', 'Vulnerabilities', 'SBOM', and 'Builder'. Below these is a 'Query vulnerability' section with a text input 'Enter vuln ID here...' and a 'Search' button. The main area is a dependency graph with nodes like 'git', 'github.com/aws', 'aws-sdk-go-v2', 'pkg.s3shared', 'github.com/aws/aws-sdk-go-v2/service/internal', 'golang', 'github.com/aws/aws-sdk-go', 'pkg.s3', 'vt.12.0', and 'vt.5.2'. A 'depends on' relationship is shown between 'pkg.s3shared' and 'vt.12.0'. At the bottom, there is a breadcrumb trail: 'github.com/aws/aws-sdk-go...' >> 'pkg.s3shared'.

Sources and Databases



Attackerkb: Rapid7 forum for sharing insights and views through all the (over or under) hype and chaos. Not a proper vulnerability database, but CVEs there are real security issues.



CVE-2024-34102

Disclosure Date: June 13, 2024 (last updated July 10, 2024)

Adobe Commerce versions 2.4.7, 2.4.6-p5, 2.4.5-p7, 2.4.4-p8 and earlier are affected by an Improper Restriction of XML External Entity Reference ('XXE') vulnerability that could result in arbitrary code execution. An attacker could exploit this vulnerability by sending a crafted XML document that references external entities. Exploitation of this issue does not require user interaction.

Attack Vector: Network Privileges: None User Interaction: None

👁️ 3 💬 2

Exploit

PoCs that have not been added by contributors directly have been sourced from: [nomi-sec/PoC-in-GitHub](#). A PoC added here by the AKB Worker must have at least 2 GitHub stars.

[CVE-2024-34102 \(https://github.com/th3gokul/CVE-2024-34102\)](https://github.com/th3gokul/CVE-2024-34102) (Added by AKB Worker)

[CVE-2024-34102 \(https://github.com/bigb0x/CVE-2024-34102\)](https://github.com/bigb0x/CVE-2024-34102) (Added by AKB Worker)

[CVE-2024-34102 \(https://github.com/Chocapikk/CVE-2024-34102\)](https://github.com/Chocapikk/CVE-2024-34102) (Added by AKB Worker)

[CVE-2024-34102 \(https://github.com/0x0d3ad/CVE-2024-34102\)](https://github.com/0x0d3ad/CVE-2024-34102) (Added by AKB Worker)

[CVE-2024-34102 \(https://github.com/EQSTSeminar/CVE-2024-34102\)](https://github.com/EQSTSeminar/CVE-2024-34102) (Added by AKB Worker)

Record complemented by a technical analysis, and comment section where security professionals enrich the thread.

Complementary tool to better face the daily CVE storm.

Conclusions

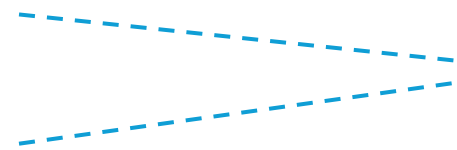
Lack of data quality in security databases

Security researchers exploiting poor oversight

Trust-but-Verify approach also towards global industry standards

Even in **Academia** the amount of inaccurate or bogus NVD entries is being studied

Interesting **aids** out there, to integrate with the existing security culture, scoring system, and databases



Novel challenges for traditional vulnerability management practices

The *Mystical* Stage is All Yours!

CVE-1999-0526: *the Great Worm*



An X server's access control is disabled (e.g. through "xhost +" command), allowing anyone to connect to it.

Bizarre because:

1. First Major Worm: one of the earliest instances of a worm spreading across the Internet
2. Creator: written by a Cornell graduate student, it was an experiment to test the size of the Internet
3. Unintended Impact: it replicated excessively (widespread network congestion and internet services disruption)
4. Legal Value: First case of conviction under the Computer Fraud & Abuse Act: 3 years of probation, fine

CVE-2020-10148: *SolarWinds Supply Chain Attack*

Sophisticated Supply Chain attack targeting SolarWinds' Orion platform. Insertion of malicious code into legitimate software updates, which were then distributed to SolarWinds customers. The updates contained a backdoor: thousands of organizations, including several U.S. government agencies and private companies, were affected.

Bizarre because: scale of the attack, level of sophistication, undetected for several months.

CVE-2021-44228: *Log4Sh*



This CVE was highly impactful, affecting the widely used Apache Log4j library. **Bizarre because:**

1. Wide Adoption: foundational in Java-based apps, making the vulnerability widespread across industries.
2. Severe Impact: remote attackers could execute arbitrary code, compromising entire systems and sensitive data.
3. Rapid Exploitation: attackers quickly exploited the CVE, highlighting its urgency for patching it.
4. Complex Mitigation: Fixing the vulnerability required identifying and updating all instances of the library.

CVE... (Choose your fighter)





Q&A Time!



References

[1] Guo, Hao & Chen, Sen & Xing, Zhenchang & Li, Xiaohong & Bai, Yude & Sun, Jiamou. (2022). Detecting and Augmenting Missing Key Aspects in Vulnerability Descriptions. ACM Transactions on Software Engineering and Methodology. 31. 10.1145/3498537.

[2] Giannakopoulos, Thrasyvoulos & Maliatsos, Konstantinos. (2024). On the Usage of NLP on CVE Descriptions for Calculating Risk. 10.1007/978-3-031-54204-6_6

[3] National Vulnerability Database, "CVE-2022-36446 Detail," [Online]. Available: [<https://nvd.nist.gov/vuln/detail/CVE-2022-36446>]

[4] Spring, Jonathan & Hatleback, Eric & Householder, Allen & Manion, Art & Shick, Deana. (2021). Time to Change the CVSS?. IEEE Security & Privacy. 19. 74-78. 10.1109/MSEC.2020.3044475.

[5] "CVSS Score: A Heartbleed by Any Other Name," AT&T Cybersecurity, [Online]. Available: [<https://cybersecurity.att.com/blogs/security-essentials/cvss-score-a-heartbleed-by-any-other-name>]

[6] "CVE/NVD Doesn't Work for Open Source and Supply Chain Security," CrashOverride, [Online]. Available: [<https://crashoverride.com/blog/cve-nvd-doesnt-work-for-open-source-and-supply-chain-security-2>]

[7] "Beg Bounties", [Online]. Available: [<https://www.troyhunt.com/beg-bounties/>]

[8] "Stakeholder-Specific Vulnerability Categorization (SSVC)," CISA, [Online] : [<https://www.cisa.gov/stakeholder-specific-vulnerability-categorization-ssvc>]