On Definition of Vulnerability Discovery

Simo Huopio and Erno Pasanen
Finnish Defence Research Agency, Riihimäki, Finland

simo.huopio@mil.fi
erno.pasanen@mil.fi

Abstract: The adaptation and exploitation of security vulnerabilities in military operations is a complex, multidisciplinary problem that calls for precise foundational definitions. This paper establishes such groundwork by discussing the principal elements of the topic from the perspective of security vulnerability research. We first characterize the sources and details of vulnerability and weakness information, describe common collection and presentation practices, and identify the principal consumers of this data. Next, we distinguish between vulnerability research and vulnerability discovery, explaining the processes that generate vulnerability information and highlighting the subtle but important differences between these activities. Finally, we examine principles governing the operational use of the vulnerabilities in military context and assess the respective contributions of research and discovery to achieving operational objectives. Within cyber domain operational planning we elaborate on (principle of) non-forceability. The primary hypothesis of non-forceability is that in cyber domain an adversary cannot induce an adverse effect on the logical components of a target system without proof of an exploitable logical vulnerability. This claim requires a precise explanation of the logical part of a system, which we define to include software, data, configurations, and formally defined user procedures and training; it explicitly excludes underlying processing platforms (hardware) and human actors (users and administrators). Under these definitions, external adverse effects on the logical system necessarily imply the proof of a vulnerability. Finally, the role of proof-of-concept (PoC) code in the operational utilization of software vulnerabilities has to be elaborated. We argue that the utility of vulnerability information is highly related on intended use and the level of technical detail provided. For defensive purposes, identification of a vulnerable component and its triggering mechanism is typically sufficient to trigger defensive actions. By contrast offensive operations require validation of adversaries' defensive capabilities before PoC code baseline capabilities, further development objectives and possible actions of objectives (impact) can be evaluated as an operational course of action.

Keywords: Vulnerability discovery, Vulnerability research, Cyber operations, Vulnerability exploitation, Proof of concept code

1. Background

Vulnerability discovery is a process of acquiring the technical knowledge of the existence and technical details of exploitable vulnerability on a target system. It is an enabler of cyber operations, which are systematic and planned actions that either are used to actively protect own systems by patching vulnerabilities and using security controls or leverage and exploit one or more of vulnerabilities of the target system in order to reach the overall operational goals.

The difference from the operations in physical domain is that if the target is properly protected in cyber domain, the defences cannot be forced even with significant superiority within the same domain. The exceptions to this logic are the human and physical elements of the protection, which deliver the effect outside the cyber domain. *Within the cyber domain, external actor cannot have any adverse effect to a logical part of the target system without knowing exploitable logical vulnerability.*

Non-technical security vulnerabilities in target system can be considered to have an effect in the logical world of cyber domain. As an example, inadequate security related routines of cyber system workforce are a security vulnerability if they enable the possibility of adversarial impact. Furthermore, even as the logical world of cyber domain is built on physical things there usually exists a possibility for the adversary to circumvent the elements of physical protection and humans with force superiority. For example, breaking physical barriers and forcing the human operators to violate the given security policy. These categories of vulnerabilities are not addressed in this paper.

2. Vulnerability information

2.1 Definitions Regarding Vulnerabilities

Dictionaries categorize *vulnerability* as an object either open to attack or damage ('vulnerable', 2024a) or being able to be easily hurt, influenced or attacked ('vulnerable', 2024a). According to MITRE (2025a), a vulnerability is an instance of one or more weaknesses in a system, that can be exploited, causing a negative impact to confidentiality, integrity, or availability. *Therefore, a security vulnerability is an exploitable weakness that circumvents security controls and is shown to have a negative effect when exploited.*

A (security) *weakness* is a condition in a software, firmware, hardware, or service component that, under specific circumstances, could contribute to the introduction of vulnerabilities (MITRE 2024a) by allowing the violation of an explicit or implicit security policy (MITRE 2024b). The conditions are usually called bugs, errors or flaws, and commonly used interchangeably regarding different severities; As an example, a typo can be called a bug, a logic mistake in code an error, and deeper mistake in software logic a flaw. Weakness is a specific class of error or flaw which turns to a vulnerability when instantiated in a system. Most common weaknesses are referred in *Common Weakness Enumeration (CWE)* -database (MITRE 2024c), a community-developed list of weaknesses maintained by MITRE.

There are many public vulnerability databases (Kekül, H, Ergen, B, and Arslan H, 2022) that provide *vulnerability descriptions*. NIST maintains one of the largest public databases, NVD (National Vulnerability Database) (NIST 2019). The vulnerabilities listed in these kinds of databases include typically *CVE (Common Vulnerabilities and Exposures) record* information. The CVE Record, issued by *CVE Numbering Authorities (CNAs)*, include the CVE ID, brief description of the vulnerability and number of references to vulnerability reports and advisories. Typically, the description includes also codified rating *The Common Vulnerability Scoring System (CVSS)*, which is created to help the software vendors and users comparing the severity of the vulnerability.

Proof of Concept (PoC) code is package of information with which one can reproduce the vulnerability on the targeted system, for example scripts, source code, or e.g. instructions for manipulating an user interface. The PoC should provide ways to validate triggering of vulnerability, for example information on specific error message or memory and register values that can be validated with a debugger. A PoC code covers typically a single vulnerability or a chain of vulnerabilities and generally does not impact the system otherwise than showing that system is vulnerable.

An exploit (code) makes use of the vulnerability to produce an effect in the target system. The effect can be a denial of service (DoS) where functionality of the target system is fully or partially disabled or e.g., remote code execution (RCE) where it is possible to deliver and execute attacker defined code in the target system. Exploit codes incorporate usually an elaborate combination of techniques to evade security controls. It is common that an exploit code targeting a software running modern operating system uses exploitation of multiple vulnerabilities.

2.2 Discussion About Vulnerability Information

Many parties contribute to publicly available vulnerability data. A large portion of it gets published by white hat (Shea, S. 2019) security research organizations. Vulnerability disclosures are typically preceded by a coordinated collaboration period with software vendors, often facilitated by one or more CERTs (Computer Emergency Response Team). During this period vendors can fix the vulnerability and prepare, test, and distribute the necessary updates which leads to end-users having patched systems when vulnerability is disclosed. This process is called "*Responsible Disclosure*" while some actors prefer "*Full Disclosure"* which includes releasing all the details to public without giving the software vendor any time to fix the bug beforehand. Some actors, like Google Project Zero, strike a kind of middle ground, giving a fixed 90-day period to fix the bugs, and 30-day period after the fix has been released before all details are released.

Vulnerabilities are also categorised by how long time has passed since the software vendor has acquired the knowledge of the vulnerability; *a zero-day vulnerability* means that vendor learns at time of disclosure. The vulnerabilities which have been fixed and the patch is not widely applied are called *one-day vulnerabilities*. To emphasise time passage since vulnerability disclosure (patched or not) term *n-day vulnerability* is used.

Information about most of the vulnerabilities end up to the public domain through security mailing lists (seclists.org. 2024) like Full Disclosure (Seclists.org. 2026), or a national CERT vulnerability mailing list (NCSC-FI. 2025), with posts linking to the researcher or software vendor blog giving more details about the vulnerability and ways to update.

2.3 Vulnerability Information - Conclusions

Vulnerabilities are different from regular software bugs because they have the potential to endanger the security of a system and its data. Cyber security researchers see this as a business opportunity, striving to claim CVEs for finding new vulnerabilities, branding bugs and creating security solutions to protect against malicious activities and reduce risks. However, as vulnerability information is consumed its usefulness varies between defensive and offensive scopes.

For the defender the vulnerability information has typically enough significance to trigger immediate action even without a patch or PoC information. Assessing the significance to own systems and setting up pre-emptive security controls (mitigations) can start before available patch. Vulnerability management process promotes a defensive approach to potential system risks requiring technical measures that mitigate threats even when detailed vulnerability information is unavailable. Such actions demand domain-specific and system-level expertise, for example in isolating management interfaces from adjacent networks.

For the attacker, a vulnerability information without a working PoC code does not yet have a significant value for the attacker. Even if PoC code is publicly available, it does not mean that anyone can exploit the vulnerability or that the vulnerability can be exploited with desired impact. Nevertheless, using PoC as a starting point for exploit development is easier than starting with just vulnerability description in typical CVE entry. (Shimizu, N. and Hashimoto, M 2026) For example, a vulnerability described just as occasional random changes in target application resources does not offer a lot to build an exploit on; without an exact method to trigger the vulnerability and information on how to control the changes on target the vulnerability remains without direct value. Alternatively, if there would be details on the weakness behind the vulnerability and at least one concrete way to control it, the starting points for exploit development is clear, and there is also a data to evaluate other related tactics, techniques, and procedures (TTP) for the attack (MITRE 2025b).

In conclusion, PoCs in a defensive context help build and validate existing mitigations and strengthen overall system security. In contrast, offensive use demands more detailed vulnerability information, as exploits must align with operational objectives and typically require additional research and development.

3. Security Vulnerability Research and Discovery

3.1 Definitions

Vulnerability research (VR) is a holistic research activity done by academic, commercial or governmental actors which aims for acquiring best available knowledge regarding vulnerability discovery tools and -methods. As sources research actors use public and academic sources, free and commercial trainings and possibly subcontracted research. Scientific method is used where applicable and research results are shared with research community for peer review and feedback. Therefore, found methods and built tools are available to parties doing vulnerability discovery and discovered methodologies and insights are available to other vulnerability researchers.

Vulnerability discovery (VD) is the act of searching for vulnerabilities from the applications, platforms and embedded systems of interest by applying the best public and commercial tools and methods available. The motivations behind the activity vary from actor to another; finding bugs from your own products (software vendors), finding bugs from the products you use (end users), making the internet safer place (white hat actors), earning money (bug bounty participants, vulnerability brokers), or getting enablers for your offensive operations (penetration testers, governmental actors, criminals).

Target setting for vulnerability discovery activity depend on customer; While needs of an offensive cyber operation can be very specific (“application of certain version”), the participants of a bug bounty programs have targets specified by the software vendors (“newest version of our product”), some white hack actors and criminal actors target everything their skills enable, selecting perhaps targets by potential impact of the findings (“All internet-facing popular applications”).

The vulnerability discovery can be divided in following phases:

- Open-source information gathering: Establishing the vulnerability history and current status of the target from partners and open sources.
- Analysis of the assumed usage the target. This can be enriched with existing threat emulation information.
- Architecture and software bill of materials (SBOM) analysis of the target implementation. Especially if target is larger than a single executable binary.
- Static analysis of the source code if available.
- Dynamic analysis of the executable code and the system at large using fuzzers, symbolic execution, or tracing, capturing and modifying the communications and debuggers
- Triage findings to three categories: Exploitable, possibly exploitable and non-exploitable using error logs, debuggers, trace tools and disassemblers
- Reverse engineering and analysis of the found anomalies using same tools as in Triage

- Creating the proof-of-concept code for reliably triggering the vulnerability (for defensive use) or for showing that the vulnerability could be exploited (for offensive use).
- Iterating to new discovery cycle in order to reach closer to actual target
- Communicating the results with customers, including impact analysis

The flow of vulnerability discovery on an abstract system is visualised in following Figure 1 (below) depicting the iterative cycle of vulnerability discovery: First iteration is launched to analyse the protected public application programming interface (API). Second iteration exploits the interim results and internal API in order to reach the wanted results.

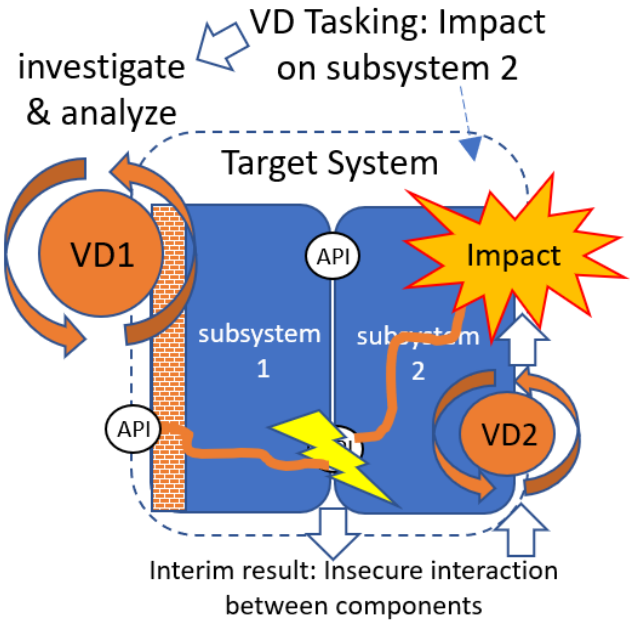

Figure 1: Example of iterative process of vulnerability discovery

Naturally not all vulnerability discovery processes follow all the listed phases; when testing own products during R&D you already have enough information to start static- and dynamic analysis, and no need for sophisticated PoC code to start fixing findings. Furthermore, the more complex and obfuscated the system is the more starting information, iteration and time is needed to discover vulnerabilities, if even possible with current knowledge.

3.2 Discussion on Vulnerability Research and Discovery

In practice every system has exploitable vulnerabilities in its architecture, software, and configuration. Some of these vulnerabilities are remainders of activities in the system R&D, some are result of the customer configuring the system in the usage environment and making decisions regarding update practices. This can be discussed and communicated with a single concept of security debt (Huopio, S. 2020), high amount of it means higher risk of having to pay it later in forms of security incidents.

Changes in the system configuration, operating environment or its functionality are common sources of new vulnerabilities. Because of this, the larger changes of architecture, and especially addition of new type of functionality have to be thoroughly analysed; New additions can for example expose old interfaces to new network segments, or introduce altogether new interfaces to access the system.

In order to have a credible cyber defence posture, a governmental user of critical heterogenous ICT infrastructure has to have an access to efficient vulnerability discovery capability. Usually, the system is so complex and multi-layered that relying just to the ability of the supply chain informing about software issues is not enough; *Objective for credible cyber defence posture has to be set on ability to do basic vulnerability discovery on all the relevant software platforms and embedded system types in own inventory.* (Goel, J.N. and Mehtre, B.M. 2015)

It is a good idea to start vulnerability discovery activities with a technical threat analysis workshop with help from vulnerability researchers and subject matter experts on related systems. Main objective of these workshops is to raise the technical vulnerability and risk knowledge of system/capability owner to a realistic

level subsequently prioritizing defensive activities according to defence-in-depth principles while gathering information for vulnerability discovery campaign. The workshop results can include: revised knowledge of the system use- and misuse cases, realistic threat environment and identification of critical paths and interfaces that could be used to disrupt the system.

The technical act of vulnerability discovery builds on top of results on the technical threat analysis workshop and involves a lot of manual work, interactive iterations of reviewing, choosing and applying tools and analysing the resulting system behaviour. It can be pursued in many abstraction levels; logical flaws can be found from analysing system architecture and (human or machine) interface thresholds or -networks, static analysis on source code and dynamic analysis of system essential binaries. The system can be tested with intentionally limited resources possibly revealing anomalies in less optimal resource conditions.

Results of vulnerability discovery are triaged in order to efficiently move forward with most likely (security) bug candidates. Dynamic analysis, especially fuzzing tools, can generate a lot of false positive results in form of vaguely anomalous behaviour which is typically hard to reproduce. With enough domain knowledge and proper use of automation, debuggers and tracing tools the analysts can extract the actual bugs out of the mass of data.

At this stage of process vulnerability discovery activities have typically produced at least some bugs with or without security impact. When working on defensive role, these findings can be brought to knowledge of the software vendor or in-house developer who can then proceed on fixing them. In this context *the key result of vulnerability discovery is the realistic hands-on knowledge of the weaknesses of the systems and the concrete own ability to assess the security posture.*

3.3 Conclusions

The threshold between vulnerability research and discovery is often blurred. While a target may seem easy to test initially and yield shallow bugs, advanced and experimental techniques might be needed to reach the deeper layers of the software logic and discover the deep bugs. Therefore, advancing on both routes are needed; getting systems under testing fast, and continuously researching more effective ways to discover vulnerabilities. Testing is always preferable to just speculating about potential vulnerabilities.

Vulnerability research is a comprehensive approach that is used to gather knowledge and to develop tools for broad applications, while vulnerability discovery applies those tools and insights to address specific issues. In this context, *vulnerability research is a holistic, exploratory process, whereas vulnerability discovery is a focused deterministic task.*

As time passes vulnerability research will continuously find new interfaces and approaches for vulnerability discovery to enhance defensive security posture: researchers bring new testing technologies and integrate and evaluate latest tools. Applying new methods and processes to unknown (Schumilo, S. et al. 2017) or known part of the system-under-test (SUT) can lead to surprising (Schumilo, S. et al. 2021) or even contested results (Nicolae, M. I., Eisele, M., & Zeller, A. 2023) and in a best-case scenario driving up the software quality by continuous testing and requirement crafting.

Ultimately, the primary distinction between vulnerability research and discovery lies in the intended target and its operational context, whether offensive or defensive. The less clearly defined the target, the more holistic the approach becomes, requiring broader research and information. A more targeted approach doesn't eliminate the need for research but instead directs it toward either expanding the attack surface or refining tool development.

4. Security Vulnerability Usage in Military Operations

4.1 Definitions

Cyber operation is a military operation executed in cyber domain. It can be of defensive or offensive nature, and has many separately defined subtypes (Scott, K.D. 2018). As discussed before vulnerabilities play a central role in all cyber operations as *all effects in cyber domain can be mapped as results of exploitation of a weakness or vulnerability*.

Operational context for using vulnerabilities therefore means that the existence of vulnerability is either a reason for executing protective actions in defensive operation or an enabler of one or more Course of Action (COA) proposals in an offensive military operation. The operational importance of the vulnerability is higher when PoC code or weaponised exploit code is available. A discovered vulnerability without apparent impactful

exploitation can be used as R&D starting point but if its usefulness is uncertain the cyber operation has to prepare for delays in execution and planning cycle.

Impact of exploited vulnerability can be defined as the desired effect (e.g. denial of service, remote execution, privilege escalation) and exploitation parameters (authentication needed/not needed, other environmental or version requirements for exploitation) in a specific part of the target system. In the operational context impacts description can be accompanied with additional parameters regarding the impact success probability, temporal factors (effects single transaction, certain time, or has permanent impact), or collateral impact description (Impact of system stability and non-targeted functionalities). The intended impact is partly shaped by the maturity of the exploit code. Although further R&D can enhance maturity, the inherent impact of a vulnerability is fixed by its underlying effects and cannot be changed. In context of military operational planning the impact of vulnerability should be used when planning for operational effect.

Payload is a piece of executable code or series of separate executable commands crafted to be used together for desired effect. Payload is provided to the vulnerability exploit code, which transfers it to the target system, prepares the environment so that the code can be executed, and triggers the payload code. It can be argued that in some cases part of exploit code can also be considered as payload, as it itself executes commands on target system. In the context of this paper, payload is defined as code specifically providing the planned effect for the operational target.

Vulnerability Equities Process (VEP) is a military process of making decisions on the usage of software vulnerabilities that are discovered or developed within own military organisation, or acquired from partners or subcontractors. By this definition the vulnerabilities that are of public domain are not considered in VEP.

4.2 Discussion

Vulnerability information is in central role in cyber operation planning, related decision making and in the actual execution both in defence an offence: when defending a system, the existence of vulnerable code in the system is critical information and causes immediate defensive actions of mitigation and patching. This information can be also gathered via preparation of battlefield; including vulnerable components in honeypots can lure attackers to reveal their attack capabilities and give the defenders information on how the vulnerabilities are being exploited.

In offensive operations vulnerability information is an enabler. Vulnerability discovery requires intelligence on target architecture, interfaces and specific software versions. Exploiting this information requires obtaining or developing suitable exploit code and maintaining the expertise to test and deploy it effectively. As the public vulnerability information is often lacking in the technical exploitation details, own vulnerability research and discovery capabilities are practically the only way forward. Preparing for almost unlimited variation on vulnerability discovery targets cannot be managed without capable vulnerability research supporting the vulnerability discovery team. The resource discussion is out of scope of this paper. What can be discussed is the approach in principle.

Offensive cyber operations imply a need for capability to develop proof of concept code for triggering and exploiting found vulnerabilities. It is worth noting that the knowledge of n-day vulnerabilities have also their worth; it is very common to discover outdated systems during reconnaissance. As vulnerability databases do not usually host exploit code with vulnerability details, exploit development is often needed also for n-day vulnerabilities.

The typical needs of offensive operations are targeted in a very specific way; vulnerabilities are needed for specific application or service, with a specific software version and specific operating environment. These kinds of tasks are hard and slow while outcome is very difficult or impossible to estimate. This type of activity requires additional information on adversary defence capabilities to allow targeting adversary weak spots while evading adversary. While the possibility of finding an exploitable bug in offensive operations timeframe is low, the impact of a new zero-day vulnerability would be high enough to justify the needed resources.

Another way of approaching offensive vulnerability discovery dilemma is to apply vulnerability discovery as “background” activity to large target space (for example, a Linux distribution). To maximise the yield and the probability of desired results in large scale the discovery tools should be used in necessary scale, parallelism and in an optimized computing infrastructure, as in OSS-FUZZ project (GitHub 2022). After a vulnerability is discovered a reverse search for target can be applied and operational planning can start with an initial COA.

In the larger context of cyber operation planning, especially in multi-domain operation (MDO) context, the potential external attack surface is much larger than just a single weakness, and typically more weaknesses need to be exploited after the initial access in order to reach the operational goal. As established earlier, there is a wide range of weakness types and each of them need different kind of approach for exploitation. One good reference of the exploitation techniques are the initial access and privilege escalation categories of MITRE ATT&CK knowledge base (MITRE 2025b).

While pure cyber operations have their place, especially on intelligence information gathering, many military operations are executed as multi-domain operations. ATT&CK lists some vulnerability exploitation methods that require physical access to parts of target system, e.g. by connecting an attacker-controlled hardware to the target system. These additional mission parameters help vulnerability discovery as it circumvents security boundaries through usage of other domains and provides more COAs.

In military cyber operations, vulnerability discovery is a key enabler of success. The ability to discover new exploitable vulnerabilities in a system of interest can used for both defensive and offensive gains. Therefore, in military cyber operations the vulnerability discovery should be exploited as supportive action whenever it is possible. In addition to own vulnerability discovery actions, access to non-public vulnerabilities and exploit code can be established.

Regardless of the source of new vulnerability information, the usage of specific vulnerability has to be decided in the context of operational needs. It is not outright certain that all vulnerabilities should be disclosed as they could be used offensively. This leads to discussion about Vulnerability Equities Process (VEP) (Vulnerability Equities Policy and Process for the US Government) (The White House 2017) which is out of scope of this paper.

Figure 2 illustrates the vulnerability-related data flows around security vulnerability research and discovery and security vulnerability usage in operations.

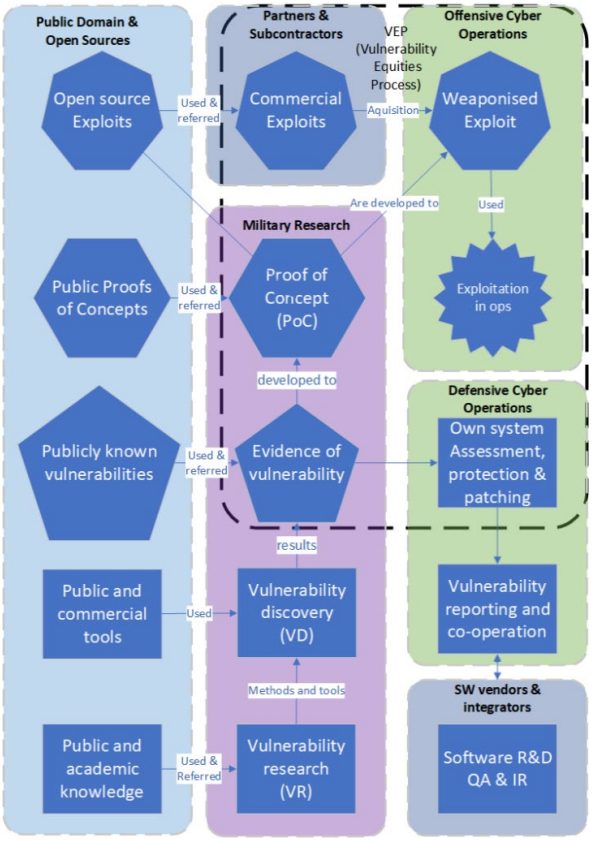

Figure 2: Vulnerability related data flows in military context

4.3 Conclusions

In cyber domain, all effects require a vulnerability on target system that is exploited. Without the technical knowledge of vulnerability and its exploitation and means to execute it, an operation cannot use it as a part to

reach its goal. For ability to plan and execute offensive cyber operations there has to be means to acquire vulnerabilities and related exploit code from open sources, through own vulnerability discovery or from other venues.

Vulnerability discovery is not inherently offensive or defensive. Even exploit code development is not offensive as such. Only when the exploit code is used to deliver a payload in order to make an impact is the act clearly offensive.

In an operational context, PoC code is used to define and validate the potential impacts of a vulnerability in offensive scenarios. The absence of PoC code can have collateral effects also in defensive operations: if the exploitation method is unclear, only mitigation can be applied, not full remediation. Some weaknesses can't be mitigated due to their context—critical system requirements may even prevent adequate mitigation. Therefore, even in defensive contexts, creating PoC is essential to guide both mitigation and remediation efforts. Thus, a PoC-code is required for both defensive and offensive operations to accurately assess the impact of a vulnerability.

Ethics declaration: Ethical clearance was not required for the research.

AI declaration: Outside of triggering now ubiquitous AI-assisted web searches, no other AI tools were used for the research.

References

Black Duck (2019) 'What are the different types of security vulnerabilities?' Software and Application Security Blog, 26 August 2019, Available at: https://www.blackduck.com/blog/types-of-security-vulnerabilities.html (Accessed May 2025)

GitHub. (2022). OSS-Fuzz: Continuous Fuzzing for Open Source Software. Available at: https://github.com/google/oss-fuzz.

Goel, J.N. and Mehtre, B.M. (2015) Vulnerability assessment & penetration testing as a cyber defence technology. Procedia Computer Science, 57, pp. 710-715.

Huopio, S. (2020). 'A quest for indicators of security debt', *The Cyber Defense Review*, 5(1), pp. 169-184.

Kekül, H, Ergen, B, and Arslan, H, (2022) 'Comparison and analysis of software vulnerability databases.' *International Journal of Engineering and Manufacturing.* 12(4). 1.

MITRE (2024a) About CWE. Available at https://cwe.mitre.org/about/ (Accessed October 2024)

MITRE (2024b) 'Vulnerability' definition in CVE Glossary Available at https://www.cve.org/ResourcesSupport/Glossary/#glossaryVulnerability (Accessed October 2024)

MITRE (2024c) CWE Common Weakness Enumeration -database. Available at https://cwe.mitre.org (Accessed October 2024)

MITRE (2024d) CWE Top 25 Most Dangerous Software Weaknesses. Available at: https://cwe.mitre.org/top25/ (Accessed October 2024)

MITRE (2025a) CWE Glossary. Available at https://cwe.mitre.org/documents/glossary/index.html (Accessed 10 May 2025)

MITRE (2025b). MITRE ATT&CK. Mitre.org. Available at: https://attack.mitre.org/

NCSC-FI. (2025). NCSC-FI newsletters NCSC-FI. Available at: https://www.kyberturvallisuuskeskus.fi/en/ncsc-news/subscribe-our-newsletters [Accessed 13 Jan. 2026]

Nicolae, M. I., Eisele, M., & Zeller, A. (2023). 'Revisiting neural program smoothing for fuzzing.' *In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 133-145).

NIST (2019). NVD. Nist.gov. Available at: https://nvd.nist.gov/

SANS (2024) SANS Institute. Available at: https://www.sans.org/emea (Accessed October 2024)

Schumilo, S. et al. (2017). '{kAFL}:{Hardware-Assisted} feedback fuzzing for {OS} kernels.' *In 26th USENIX security symposium (USENIX Security 17)* (pp. 167-182).

Schumilo, S. et al. (2021). 'Nyx: Greybox hypervisor fuzzing using fast snapshots and affine types.' *In 30th USENIX Security Symposium (USENIX Security 21)* (pp. 2597-2614).

Scott, K. D. (2018). 'Joint Publication (JP) 3-12 Cyberspace Operation.' *The Joint Staff: Washington*, DC, USA.

Seclists.org. (2024). SecLists.org Security Mailing List Archive. Available at: https://seclists.org/

Seclists.org. (2026). Full Disclosure Mailing List. Available at: https://seclists.org/fulldisclosure/ [Accessed 13 Jan. 2026]

Shea, S. (2019). 6 Different Types of hackers, from Black Hat to Red Hat. SearchSecurity. Available at: https://www.techtarget.com/searchsecurity/answer/What-is-red-and-white-hat-hacking

Shimizu, N. and Hashimoto, M (2026) Vulnerability management chaining: An integrated framework for efficient cybersecurity risk prioritization. IEEE Access.

'Vulnerable' (2024a) Available at: https://www.merriam-webster.com/dictionary/vulnerable (Accessed 25 September 2024)

'Vulnerable' (2024b) Available at: https://dictionary.cambridge.org/english/vulnerabile (Accessed 25 September 2024)

The White House (2017). Vulnerabilities equities policy and process for the United States government.' *White House Report.*