CWE/CAPEC User Experience Working Group

June 1, 2022

Agenda

- **Housekeeping**
- **Primary topics**
  - Definitions!
    - *Harmonizing common terms across CWE/CAPEC (CVE?)*
    - *Review proposed definitions and consider initial feedback*
  - Continue/Finalize persona definitions for publication
    - *CWE v4.8 June 28*
  - Reminders
- **Adjourn**

UEWG: Reminders

- **Mission:** Identifying areas where CWE/CAPEC content, rules, guidelines, and best practices must improve to better support stakeholder community, and work collaboratively to fix them
- **Periodic reporting of activities to CWE/CAPEC Board**
  - (next quarterly Board meeting June 3)
- **Please solicit participations from your contacts**
  - Contact: cwe@mitre.org & capec@mitre.org

Housekeeping

- **Welcome new CWE/CAPEC UEWG Co-Chair**
  - Shadya Maldonado Rosado, Sandia National Labs

- **Community member co-chair**
  - Take a more active role in such things as:
    - Plan meeting agendas, identify opportunities for discussion/action, drive working group activities, etc.
    - Help brief the CWE/CAPEC Board on UEWG activities

Topic

Definitions!

Harmonizing common terms across CWE/CAPEC (CVE?)

Alec Summers

CWE / CAPEC Terminology

- **Authoritative sources for cybersecurity terminology define terms differently – including CWE/CAPEC!**
  - Vulnerability is defined *three* different ways between CVE, CWE, and CAPEC ☹
- **Vulnerability, Weakness, and Attack Patterns have multiple definitions across CISA, NIST, ISO Standards, etc.**

Vulnerability

Vulnerability	A flaw in a software, firmware, hardware, or service component resulting from a weakness that can be exploited, causing a negative impact to the confidentiality, integrity, or availability of an impacted component or components	CVE	website
Vulnerability	A characteristic or specific weakness that renders an organization or asset (such as information or an information system) open to exploitation by a given threat or susceptible to a given hazard	CISA	– DHS Risk Lexicon – CNSSI 4009 – NIST SP 800-53 Rev 4
Vulnerability	(CISA Extended definition) Characteristic of location or security posture or of design, security procedures, internal controls, or the implementation of any of these that permit a threat or hazard to occur. Vulnerability (expressing degree of vulnerability): qualitative or quantitative expression of the level of susceptibility to harm when a threat or hazard is realized	CISA	– DHS Risk Lexicon – CNSSI 4009 – NIST SP 800-53 Rev 4
Vulnerability	A vulnerability is a software weakness that can be exploited by an attacker. Bugs and flaws collectively form the basis of most software vulnerabilities	CISA	
Vulnerability	an occurrence of a weakness (or multiple weaknesses) within a product, in which the weakness can be used by a party to cause the product to modify or access unintended data, interrupt proper execution, or perform incorrect actions that were not specifically granted to the party who uses the weakness	CWE	website
Vulnerability	A vulnerability is a weakness that can be directly used by an adversary (via an exploit) to get a cyber-enabled capability to function in an unintended manner. Typically, this is the violation of a reasonable security policy for the cyber-enabled capability resulting in a negative technical impact. Although all vulnerabilities involve a weakness, not all weaknesses are vulnerabilities. Common Vulnerabilities and Exposures (CVE®) is a dictionary of common names for publicly known software-related vulnerabilities.	CAPEC	website
Vulnerability	Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source	NIST	Glossary
Vulnerability	A weakness in the design, implementation, operation or internal control of a process that could expose the system to adverse threats from threat events	ISACA	Glossary
Vulnerability	A vulnerability is a weakness in an asset or group of assets. An asset's weakness could allow it to be exploited and harmed by one or more threats	ISO	27001
Vulnerability	A vulnerability is a hole or a weakness in the application, which can be a design flaw or an implementation bug, that allows an attacker to cause harm to the stakeholders of an application	OWASP	website

Weakness

Weakness	A type of mistake made during the implementation, design, or other phases of a product lifecycle that, under the right conditions, could contribute to the introduction of vulnerabilities in a range of products made by different vendors.	n/a	edited from def on CWE wesbite
Weakness	Poor coding practices, as exemplified by CWEs	NIST	NISTIR 8011 Vol. 4
Weakness	A weakness is an underlying condition or construct existing in a software system that has the potential for negatively impacting the security of the system	CISA	
Weakness	A shortcoming or imperfection in software code, design, architecture, or deployment that, under proper conditions, could become a vulnerability or contribute to the introduction of vulnerabilities.	CISA	– FY 2013 CIO FISMA Reporting Metrics – ITU-T X.1520 CWE
Weakness	A type of preliminary or final finding, which is an ineffective, or lack of, implementation of one or more processes that meet the intent and value of a practice based on verified objective evidence, and applicable across the project(s) and organizational support functions or organizational unit as a whole. This is realized either by a) the process itself does not address a CMMI practice requirement, or b) the project(s) or organizational support functions are not following their process that IS compliant with the intent and value of the applicable CMMI practice	ISACA	Glossary
Weakness	A type of mistake that, in proper conditions, could contribute to the introduction of vulnerabilities within that product. This term applies to mistakes regardless of whether they occur in implementation, design, or other phases of a product lifecycle	CWE	website

Attack Pattern

Attack Pattern	The common approach and attributes related to the exploitation of a known weakness type, usually in cyber-enabled capabilities	n/a	edited from def on CAPEC website
Attack Pattern	Similar cyber events or behaviors that may indicate an attack has occurred or is occurring, resulting in a security violation or a potential security violation	US-CERT	– Oak Ridge National Laboratory Visualization Techniques for Computer Network Defense - CAPEC Website
Attack Pattern	(US-CERT Extended Definition) For software, descriptions of common methods for exploiting software systems	US-CERT	– Oak Ridge National Laboratory Visualization Techniques for Computer Network Defense - CAPEC Website
Attack Pattern	an abstraction mechanism for helping describe how an attack against vulnerable systems or networks is executed	ISACA	
Attack Pattern	An attack pattern is a general framework for carrying out a particular type of attack such as a particular method for exploiting a buffer overflow or an interposition attack that leverages architectural weaknesses. In this paper, an attack pattern describes the approach used by attackers to generate an exploit against software.	CISA	Glossary
Attack Pattern	Attack patterns are descriptions of common methods for exploiting software. They derive from the concept of design patterns [Gamma 95] applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples	CISA	
Attack Pattern	an attack pattern is a description of the common attributes and approaches employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. Attack patterns define the challenges that an adversary may face and how they go about solving it. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples. Common Attack Pattern Enumeration and Classification (CAPEC™) provides a formal list of known attack patterns	CAPEC	website

Proposed Definitions

Term	Definition	Authority	Authorities Doc
Vulnerability	A flaw in a software, firmware, hardware, or service component resulting from a weakness that can be exploited, causing a negative impact to the confidentiality, integrity, or availability of an impacted component or components	CVE	website
Weakness	A type of mistake made during the implementation, design, or other phases of a product lifecycle that, under the right conditions, could contribute to the introduction of vulnerabilities in a range of products made by different vendors.	n/a	edited from def on CWE website
Attack Pattern	The common approach and attributes related to the exploitation of a known weakness type, usually in cyber-enabled capabilities	n/a	edited from def on CAPEC website

Board Member Feedback

Board Member feedback on Glossary spreadsheet:

[redacted]

Red Hat adopted the following definition of a weakness a year or so ago. "A weakness is specifically the absence of a safeguard in an asset or process that provides a higher potential or frequency of a threat occurring, but does not meet the exploitability criteria for a vulnerability." We've also defined vulnerability much more broadly to include weaknesses as a subset "A weakness or absence of a safeguard in an asset that provides a higher potential or frequency of a threat occurring." We were running into differing opinions when we looked at each as separate and unique. The other factor we've called out internally is hardening. The key difference between a weakness and hardening for us is that a weakness is a direct factor in the potential and frequency vs hardening which are safeguards which prevent. **addendum - Process is defined here as an executing process on the stack… More and more I'm seeing the term "offering" used, which may work better than "asset". That said, I don't think there's a silver bullet for phrasing this.

[redacted]

Happy to hear there is an initiative to help align these definitions. I know it's a very common confusion point for many.

A couple of thoughts/comments from me:

- In the weakness definition the word "mistake" throws me off a bit because that implies there was awareness of the issue and an intent to not make it. But many weaknesses appear just because individuals are completely unaware. I'm trying to think of another word, but what is on the CWE site I do like at first glance: "...flaws, faults, bugs, or other errors in software or hardware implementation, code, design, or architecture..." There's probably a compromise in there that's shorter...
- I also think showing how Vulnerabilities, Weaknesses, and Attack Patterns relate to one another with a single picture would be really powerful and helpful for the community. I see the vulnerabilities as the focal point, with a set of weaknesses contributing to the vulnerability, and attack patterns forming how those weaknesses are exploited. So maybe a "funnel" with a vulnerability at the end, weaknesses spread across the input, and a cross section of attacks stringing those weaknesses together. We could surely debate the specific representation of this, but I do think a picture would be very helpful.

[redacted]

(in response to [redacted] message above)
I like the idea of defining a weakness wrt to a protection for an asset. The protection could have weaknesses because of mistakes, forgetfulness, or any other reason (e.g. environment). An asset-based definition fits really well for hardware and I think for a lot of software, but I'm wondering if that generalizes completely to all software?

[redacted] **Micro Focus:**
Please consider the following points:

1. I agree with Jason O. that the terms are a stepping stone to understanding how these concepts play out in the real world. However, a slightly different perspective is the following (without defining all of the base terms):
   a. A ***bug*** is an instance of a *flaw/fault/error/defect* in the design, development/implementation, or operation of a system.
   b. A ***weaknesses*** is a *bug* that **could** (i.e., may, or may not) lead to a vulnerability. *Weakness types* define logical groupings of bugs which share similar properties (e.g., Buffer Overflow).
   c. A ***vulnerability*** is a property of system requirements, design, implementation, or operation that **can**be accidentally or intentionally exploited (resulting in a security failure). A *vulnerability* is made possible due to the presence of one or more underlying *weaknesses*.
   d. An ***exploit*** successfully results in a security failure through one or more *vulnerabilities* which **does**exploit underlying *weaknesses*.
   e. An ***attack*** is an attempt to exploit one or more *vulnerabilities* that **could** lead to an exploit. *Attack patterns* define logical groupings of attacks which share similar properties and approaches related to underlying *weakness types*.

Note: the distinction between can and could is a comparison of probability. Can is likely to occur. Could is less likely to occur.

1. Regarding the Red Hat definition, if we want to be consistent with other standards and best practices, we should probably use the term "control" rather than "safeguard" (e.g., NIST SP 800-53 Rev. 5).

To test the observations, we should be able to apply the terms to descript actual occurrences in the context we are trying to represent. For example, consider the following:

"In December of 2021, a new ***vulnerability*** has been identified within Log4J under the common name Log4Shell (CVE-2021-44228). This *vulnerability* affects version 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1). Specifically, CVE-2021-44228 is caused by an underlying JNDI Injection and LDAP Entry Poisoning ***weaknesses*** which exists in the affected versions. To date, multiple ***exploits*** have been recorded across the industry where **attacks** targeting CVE-2021-44228 have been observed (e.g., VMWare, …).

**** quick correction ****
Just a quick correction… see below. Upon reflection, the example should not attribute an LDAP Entry Poisoning vuln (only JNDI Injection). An attack chain, which could include an LDAP Entry Poising vector (or more simply a maliciously controlled LDAP Server), could be part of a remote code execution exploit

[redacted]

(in response to [redacted] message above)
I favor this approach. The order, specifically, has a spectrum feel to it and reduces the chance of differing opinions on the vulnerability term (imho).

"Note: the distinction between can and could is a comparison of probability. Can is likely to occur. Could is less likely to occur.

1. Regarding the Red Hat definition, if we want to be consistent with other standards and best practices, we should probably use the term "control" rather than "safeguard" (e.g., NIST SP 800-53 Rev. 5)."

Good catch ... and I agree!

Topic

Continuing the Discussion: User Personas and CWE/CAPEC User Experience

Alec Summers

CWE/CAPEC Deputy Project Lead

Personas: Next Steps

- Formally define each user persona
- Share with CWE/CAPEC Board
- Once finalized, make public on our sites
- Use that as a catalyst for modernizing presentation according to persona needs

- **"Development lifecycle"**
  - Those who build systems: [DEFINE]
  - Those who use systems: [DEFINE]
  - Those who protect infrastructure around those systems: [DEFINE]
- **Educators:** Teachers, professors, or certification programs that educate developers and system designers how to develop more secure code, design more secure products, and/or how to find vulnerabilities.
- **Technical Writers:** Those who communicate advanced technical concepts as clearly, accurately, and comprehensively as possible to their intended audience (e.g., code analysis tool users or system designers)
- **Tool Developers:** Developers of code scanning products, services, and other types of automated techniques for finding weaknesses and attacking systems, and reporting/educating on findings to users
- **Security Researchers/Analysts:** Those who look for ways to attack a product by finding weaknesses using manual and/or automated techniques, then reporting the findings to the vendor and/or the general public (to include threat modeling, C-SCRM)
- **Incident Response Teams:** Those responsible for preparation and reaction to any security event

“Two-Types” Model of CWE Presentation

- **Definitions are useful but may not introduce more unnecessary complexity**
- **“Two-Types” Proposal by Premyslaw Roguski (Red Hat)**
  1. Theoretical: users who are more focused on the theoretical aspects of the weaknesses
     - Educators (teachers, professors, solution architects who design the systems' requirements)
     - Technical Writers (people responsible for security content, security blogs and articles)
     - Project and Program Managers who need some level of understanding about security and weaknesses
  2. Technical: users who are managing the security issues and need more details about the nature of the weakness and how to prevent this from happening
     - Tool Developers, Security Researchers, Pen-testers, Incident Response Analysts

Proposed Data Elements for Each Group

- **Theoretical:**
  - Description, Extended Description, Alternate Terms, Common Consequences
- **Technical:**
  - Description, Extended Description, Alternate Terms, Modes of Introduction, Demonstrative Examples, Observed Examples, Potential Mitigations, Relationships
- **Risk Element**
- **Complete filter presentation to remain as well**
- **Thoughts?**

Next Meeting – June 29 @ 12pm

PLEASE CONTACT WITH ANY QUESTIONS OR THOUGHTS

CWE@MITRE.ORG **CAPEC@MITRE.ORG**

BACKUPS

Use Case Scenarios → Content Presentation

- **CAPEC Existing functionality to filter content detail**
  - Basic, High-level, Mapping-friendly, Complete

CWE Content Filter			
BASIC	HIGH-LEVEL	MAPPING	COMPLETE
Description	Description	Description	Description
Applicable_Platforms	View_Audience	View_Audience	View_Audience
Common_Consequences	Alternate_Terms	Alternate_Terms	Alternate_Terms
Likelihood_of_Exploit	Time_of_Introduction	Terminology_Notes	Terminology_Notes
Demonstrative_Examples	Common_Consequences	Relationships	Applicable_Platforms
Potential_Mitigations	Relationships	Relationship_Notes	Modes_of_Introduction
Relationships	Content_History	Theoretical_Notes	Common_Consequences
Content_History		Related_Attack_Patterns	Likelihood_of_Exploit
		Content_History	Enabling_Factors_for_Exploitation
			Detection_Methods

and more…

Use Case Scenarios → Content Presentation

- **CAPEC Existing functionality to filter content detail**
  - Basic, Complete

CAPEC CONTENT	
BASIC	COMPLETE
Description	Description
Relationships	Relationships
Memberships	Memberships
Execution_Flow	Execution_Flow
Prerequisites	Prerequisites
Mitigations	Mitigations
Related_Weaknesses	Likelihood_Of_Attack
	Alternate_Terms
	Typical_Severity
	Skills_Required
	Resources_Required
	Indicators
	Consequences
	Example_Instances
	Related_Weaknesses
	Taxonomy_Mappings
	References
	Notes
	Content_History