CRAVEX: Calculating Risk in VulnerableCode

The design is ready for implementation. Significant revisions were made to this document on 2024-09-05.

CRAVEX project: See https://github.com/orgs/aboutcode-org/projects/8/views/1

Background

Objective: Evaluate severity, exploitability, and context factors to calculate a software package vulnerability risk score in VulnerableCode. Use that risk score to trigger the setting of vulnerability policy values on Product Inventory Items in DejaCode.

Master GitHub issue: https://github.com/nexB/vulnerablecode/issues/1543

Related GitHub issues have been assigned the “risk” tag (a changing list).

- https://github.com/nexB/vulnerablecode/labels/risk
- https://github.com/nexB/dejacode/labels/risk

Vulnerability Risk Factors

Origin

One of the primary benefits of identifying a software object using a PURL, is that in addition to the identification of the package Name and Version, it identifies its origin via Type and Namespace. As an example, note that the severity and exploitability of a package as built, deployed and used in a Linux distro may be different from the same package in another platform or environment.

Severity

Severity may be expressed as a number ranging from 0 to 10. Severity indicates the potential **impact** of a vulnerability as it applies to a software package, and specific severity scores may be assigned a descriptive label as in the following example:

- 9.0 - 10.0 Critical
- 7.0 - 8.9 High
- 4.0 - 6.9 Medium
- 0.1 - 3.9 Low

Ideally, a Severity score should be determined in the context of its origin, and how it is built or deployed. Note that CVSS scores from the NVD are not context sensitive.

Weight

Weight may be expressed as a number ranging from 0 to 10. Weight refers to the **reliability and authority** of the data source assigning a severity. Since this is an inherently subjective score, it may be advisable to implement this using some kind of config file in VulnerableCode, which would be a list of all the data sources imported by VulnerableCode, with a Weight assigned to each one.

Specific weighting scores may be assigned a descriptive label as in the following example:

- 9.0 - 10.0 Strong
- 7.0 - 8.9 Good
- 4.0 - 6.9 Fair
- 0.1 - 3.9 Unreliable

Note that weight is more specific for a specific PURL and not just a package/version in general, because the PURL gives you more context information that is critical to assessing the severity.

Data source weights are defined in https://github.com/aboutcode-org/vulnerablecode/blob/main/vulnerabilities/weight_config.py

Weighted Severity

Weighted Severity may be expressed as a number ranging from 0 to 10. For a specific software package vulnerability, the assignments of severity and weighting values will vary depending on the data source reporting the vulnerability. The availability of the complete range of these severity values may be useful to support analytical research, but a range is not practical to support automated assignment of risk to a software package vulnerability.

Weighted Severity is the maximum value obtained when each Severity is multiplied by its associated Weight/10.

```
Example of Weighted Severity: max(7*(10/10), 8*(3/10), 6*(8/10)) = 7

Example of Weighted Severity: max(3*(10/10), 10*(3/10), 10*(8/10)) = 8
```

Example of Weighted Severity: max(7*(10/10), 7*(3/10), 7*(8/10)) = 7

Exploitability

Exploitability may be expressed as one of these numbers, 0.5, 1.0, 2.0, where:

- 0.5 = no exploit known (Label: “No exploits known”)
- 1.0 = PoC/Exploit script published (Label: “Potential exploits”)
- 2.0 = Automatable Exploit with PoC script published OR known exploits (KEV) in the wild OR known ransomware OR high EPSS. (Label: “Known exploits”)

Exploitability refers to the **potential or probability** of a software package vulnerability being exploited by malicious actors to compromise systems, applications, or networks. It is determined automatically by discovery of exploits.

Risk Calculation

Risk may be expressed as a number ranging from 0 to 10. Risk is calculated from weighted severity and exploitability values. It is the maximum value of (the weighted severity multiplied by its exploitability) or 10.

Risk = max(weighted severity * exploitability, 10)

- Example of Risk: max(7.00 * 0.5, 10) = 3.5
- Example of Risk: max(7.00 * 1, 10) = 7
- Example of Risk: max(7.00 * 2, 10) = 10

Suggested assignment of descriptive Risk labels should take that into account, as in the following example:

- 8.0 - 10.0 Critical, immediate response required
- 6.0 - 7.9 High, response required as soon as possible
- 3.0 - 5.9 Medium, investigation required
- 0.1 - 2.9 Low, response deferred

VulnerableCode Model

When a user gets software package metadata from VulnerableCode, any “Affected By” Vulnerability metadata for the specific PURL should include these 3 values: Weighted Severity, Exploitability, Vulnerability Risk. Note again that these values may vary for the same package Name/Version where the Type/Namespace values are different.

Software applications (such as DejaCode) depending on vulnerability scores for prioritization purposes, based on organization-defined policies, should make use of the Vulnerability Risk value, to automatically set priorities for analysis and the setting of vulnerability status. All 3 values should be available to support the analysis, in addition to the ability to get and explore all pertinent metadata available from VulnerableCode. That analysis will support the information needed to generate a meaningful VEX (Vulnerability Exploitability Exchange) document.

Additional Notes and Discussion

-