# Vulnerabilities & PILAR (version 2024.1)

19.2.2024

## 1 Introduction

PILAR uses CVE items to process unexpected weaknesses that are reported by third parties or discovered during a vulnerability scan.

CVE may be associated to CPE, that are identifiers that identify products. if you have qualified your assets with their CPE names, you may search vulnerability databases for CVEs associated to those CPEs and, indirectly associated to your assets. that is, use CPE to identify asset vulnerabilities. It is an option.

CVE are frequently qualified with a few attributes referred to as CVSS. These attributes are used by PILAR to translate the CVE into a threat which likelihood and consequences are estimated according to the CVSS attributes.

CVSS is an industry standard. PILAR extends it with some new attributes to better reflect the circumstances and derive better likelihood and consequence estimates.

CVE are fine to detect attacker opportunities. Incidents that may happen. Your system is expected to have a protection framework, a collection of countermeasures or controls that react to the incident mitigating its consequences. In PILAR we make a distinction between

- CVE: technical vulnerabilities
- protection system weaknesses

When you combine potential incidents with protection weaknesses you may estimate the risk, that is the final or effective consequences on your business.

### 1.1 CPE – Common Platform Enumeration

Common Platform Enumeration (CPE) is a structured naming scheme for information technology systems, software, and packages. Based upon the generic syntax for Uniform Resource Identifiers (URI), CPE includes a formal name format, a method for checking names against a system, and a description format for binding text and tests to a name.

The CPE Product Dictionary provides an agreed upon list of official CPE names. The dictionary is provided in XML format and is available to the general public. The CPE Dictionary is hosted and maintained at NIST, may be used by nongovernmental organizations on a voluntary basis, and is not subject to copyright in the United States.

https://nvd.nist.gov/products/cpe

PILAR reads CPE from XML files, version 2.3.

### 1.2 CVE – Common Vulnerabilities and Exposures

The Common Vulnerabilities and Exposures (CVE) system provides a reference-method for publicly known information-security vulnerabilities and exposures. The National Cybersecurity FFRDC, operated by the Mitre Corporation, maintains the system, with funding from the National Cyber Security Division of the United States Department of Homeland Security.

https://cve.mitre.org/

## 1.3 CVSS – Common Vulnerability Scoring System

The **Common Vulnerability Scoring System** (**CVSS**) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. CVSS attempts to assign severity scores to vulnerabilities, allowing responders to prioritize responses and resources according to threat. Scores are calculated based on a formula that depends on several metrics that approximate ease of exploit and the impact of exploit. Scores range from 0 to 10, with 10 being the most severe. While many utilize only the CVSS Base score for determining severity, temporal and environmental scores also exist, to factor in availability of mitigations and how widespread vulnerable systems are within an organization, respectively.

The current version of CVSS (CVSS v3.1) was released in June 2019.

Example (xml):

```
    <vuln:cvss>
      <cvss:base_metrics>
        <cvss:score>7.5</cvss:score>
        <cvss:access-vector>NETWORK</cvss:access-vector>
        <cvss:access-complexity>LOW</cvss:access-complexity>
        <cvss:authentication>NONE</cvss:authentication>
        <cvss:confidentiality-impact>PARTIAL</cvss:confidentiality-impact>
        <cvss:integrity-impact>PARTIAL</cvss:integrity-impact>
        <cvss:availability-impact>PARTIAL</cvss:availability-impact>
        <cvss:source>http://nvd.nist.gov</cvss:source>
        <cvss:generated-on-datetime>2018-01-30T17:21:59.327-
05:00</cvss:generated-on-datetime>
      </cvss:base_metrics>
    </vuln:cvss>
```

Example (json):

```
  "impact" : {
    "baseMetricV3" : {
      "cvssV3" : {
        "version" : "3.0",
        "vectorString" : "CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H",
        "attackVector" : "NETWORK",
        "attackComplexity" : "LOW",
        "privilegesRequired" : "NONE",
        "userInteraction" : "NONE",
        "scope" : "UNCHANGED",
        "confidentialityImpact" : "HIGH",
        "integrityImpact" : "HIGH",
        "availabilityImpact" : "HIGH",
        "baseScore" : 9.8,
        "baseSeverity" : "CRITICAL"
      },
      "exploitabilityScore" : 3.9,
```

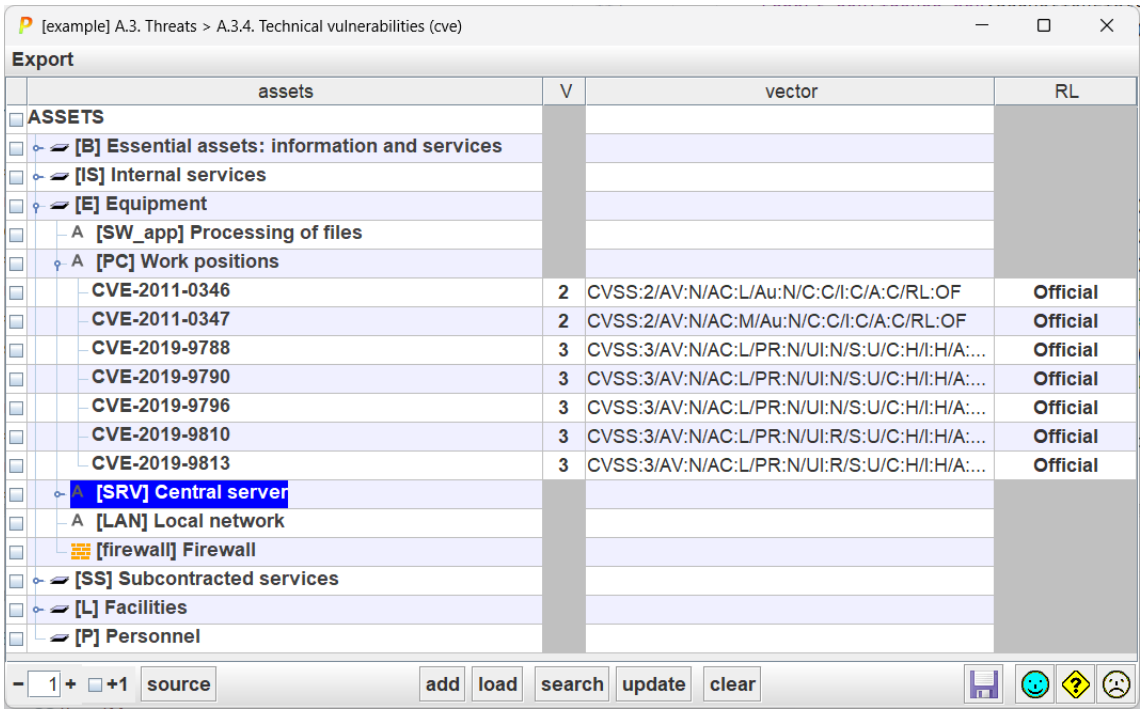
```
      "impactScore" : 5.9
    },
    "baseMetricV2" : {
      "cvssV2" : {
        "version" : "2.0",
        "vectorString" : "AV:N/AC:L/Au:N/C:P/I:P/A:P",
        "accessVector" : "NETWORK",
        "accessComplexity" : "LOW",
        "authentication" : "NONE",
        "confidentialityImpact" : "PARTIAL",
        "integrityImpact" : "PARTIAL",
        "availabilityImpact" : "PARTIAL",
        "baseScore" : 7.5
      },
      "severity" : "HIGH",
      "exploitabilityScore" : 10.0,
      "impactScore" : 6.4,
      "obtainAllPrivilege" : false,
      "obtainUserPrivilege" : false,
      "obtainOtherPrivilege" : false,
      "userInteractionRequired" : false
    }
  },
```


# 2 Load CVEs in PILAR

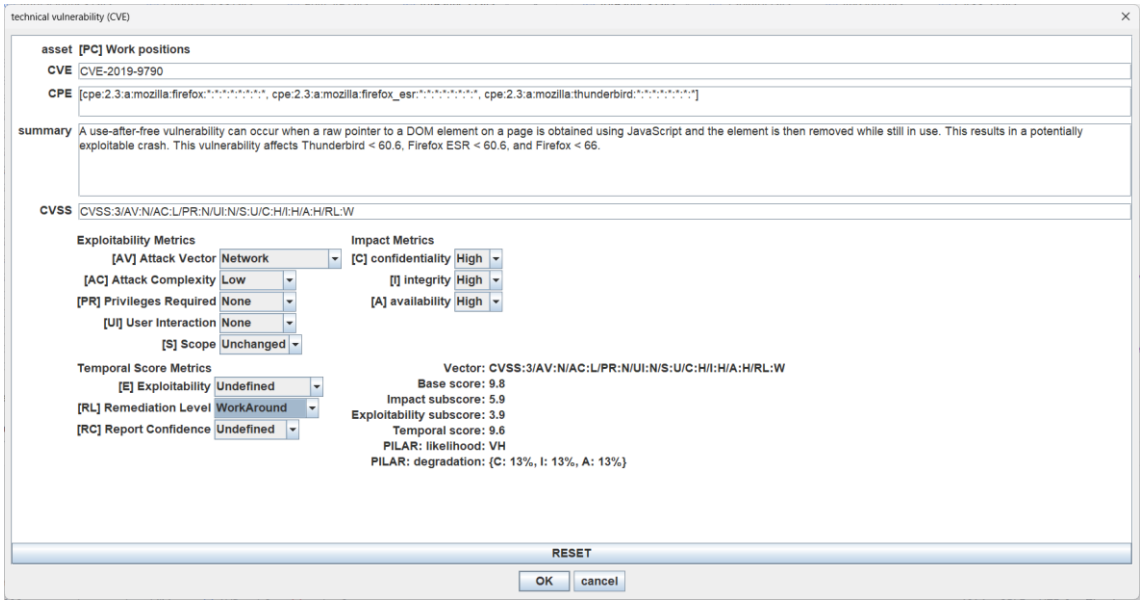
## 2.1 Manual

Risk analysis >> Threats >> Technical vulnerabilities (CVE)

Select on first column, then click ADD



Fill data as appropriate.

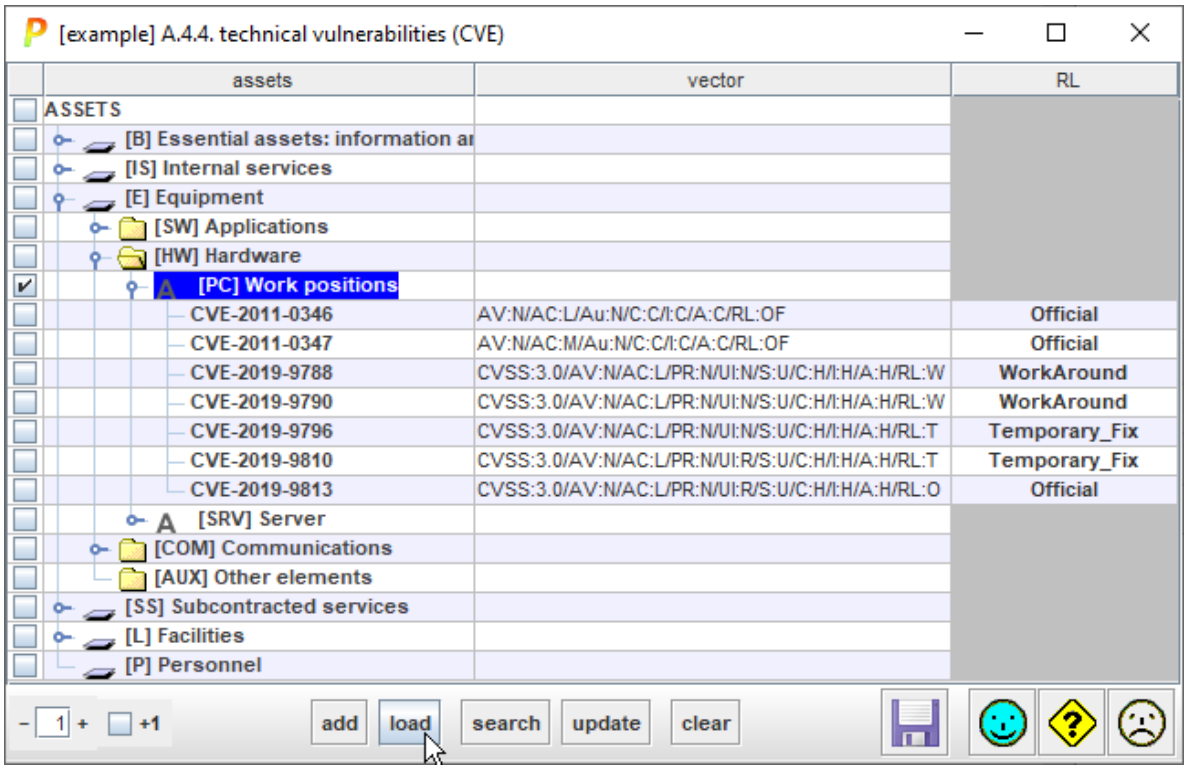



Please, note that version 2 and version 3 are significantly different. You may provide both, then PILAR will prefer version 3 information.

## 2.2 From XML

Risk analysis >> Threats >> Technical vulnerabilities (CVE)

Select on first column, then click ADD




See example in section 1.2 above.

These are the minimal fields read by PILAR for CVSS version 2

```
<?xml version='1.0' encoding='UTF-8'?>
<nvd>
  <entry id="CVE-2018-0001 min">
    <vuln:cvss>
      <cvss:base_metrics>
        <cvss:access-vector>NETWORK</cvss:access-vector>
        <cvss:access-complexity>LOW</cvss:access-complexity>
        <cvss:authentication>NONE</cvss:authentication>
        <cvss:confidentiality-impact>PARTIAL
           </cvss:confidentiality-impact>
        <cvss:integrity-impact>PARTIAL
           </cvss:integrity-impact>
        <cvss:availability-impact>PARTIAL
           </cvss:availability-impact>
      </cvss:base_metrics>
    </vuln:cvss>
    <vuln:summary>A remote, unauthenticated attacker ...</vuln:summary>
  </entry>
</nvd>
```

Please, note that XML feeds are no longer supported by NIST National Vulnerability Database.

## 2.3 From Json

Similar to load from XML.

The relevant fields read from the json files are as follows:

```
{
  "cve" : {
    "CVE_data_meta" : {
      "ID" : "CVE-2018-0001 min"
    },
    "description" : {
      "description_data" : [ {
       "lang" : "en",
       "value" : "A remote, unauthenticated attacker ..."
      } ]
    }
  },
  "impact" : {
    "baseMetricV3" : {
      "cvssV3" : {
       "version" : "3.0",
       "vectorString" : "CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H",
       "attackVector" : "NETWORK",
       "attackComplexity" : "LOW",
       "privilegesRequired" : "NONE",
       "userInteraction" : "NONE",
       "scope" : "UNCHANGED",
       "confidentialityImpact" : "HIGH",
       "integrityImpact" : "HIGH",
```

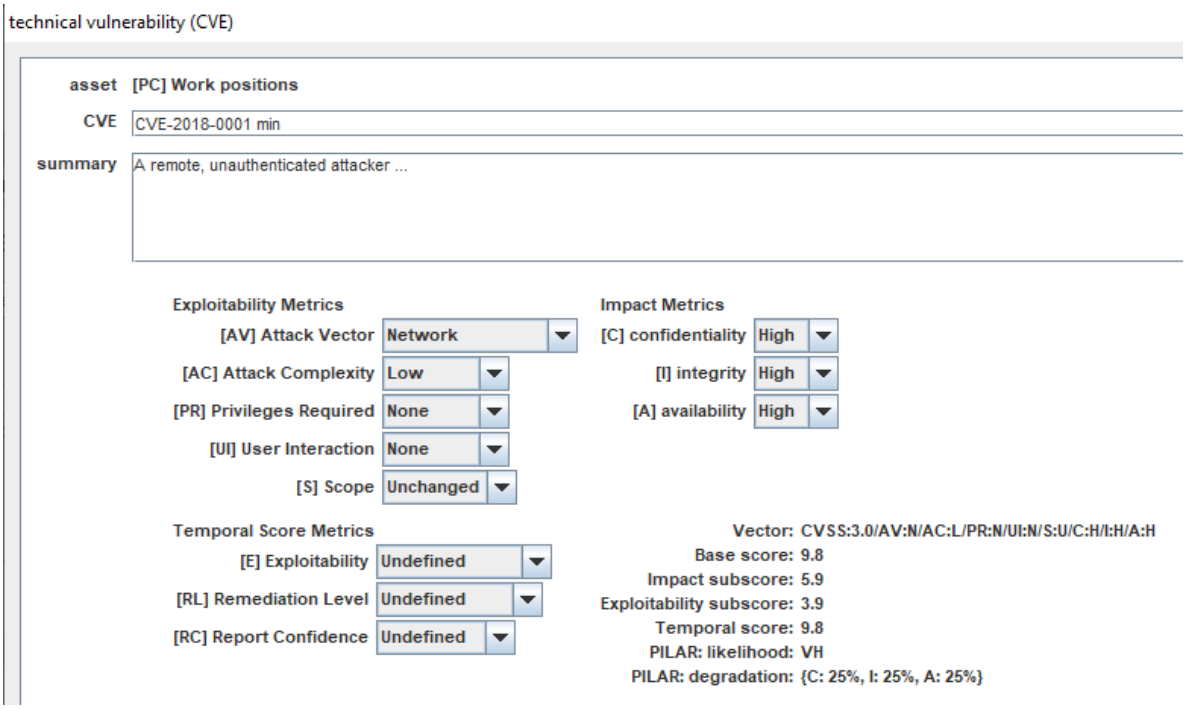
```
          "availabilityImpact" : "HIGH",
          "baseScore" : 9.8,
          "baseSeverity" : "CRITICAL"
        },
        "exploitabilityScore" : 3.9,
        "impactScore" : 5.9
      },
      "baseMetricV2" : {
        "cvssV2" : {
          "version" : "2.0",
          "vectorString" : "AV:N/AC:L/Au:N/C:P/I:P/A:P",
          "accessVector" : "NETWORK",
          "accessComplexity" : "LOW",
          "authentication" : "NONE",
          "confidentialityImpact" : "PARTIAL",
          "integrityImpact" : "PARTIAL",
          "availabilityImpact" : "PARTIAL",
          "baseScore" : 7.5
        },
        "severity" : "HIGH",
        "exploitabilityScore" : 10.0,
        "impactScore" : 6.4,
        "obtainAllPrivilege" : false,
        "obtainUserPrivilege" : false,
        "obtainOtherPrivilege" : false,
        "userInteractionRequired" : false
      }
    }
  }
```


The minimal info to be loaded is as follows; that is, unique identifier and CVSS:3 vector:


```
{
  "cve" : {
    "CVE_data_meta" : {
      "ID" : "CVE-2018-0001 core"
    }
  },
  "impact" : {
    "baseMetricV3" : {
      "cvssV3" : {
        "vectorString" : "CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H"
      }
    }
  }
}
```

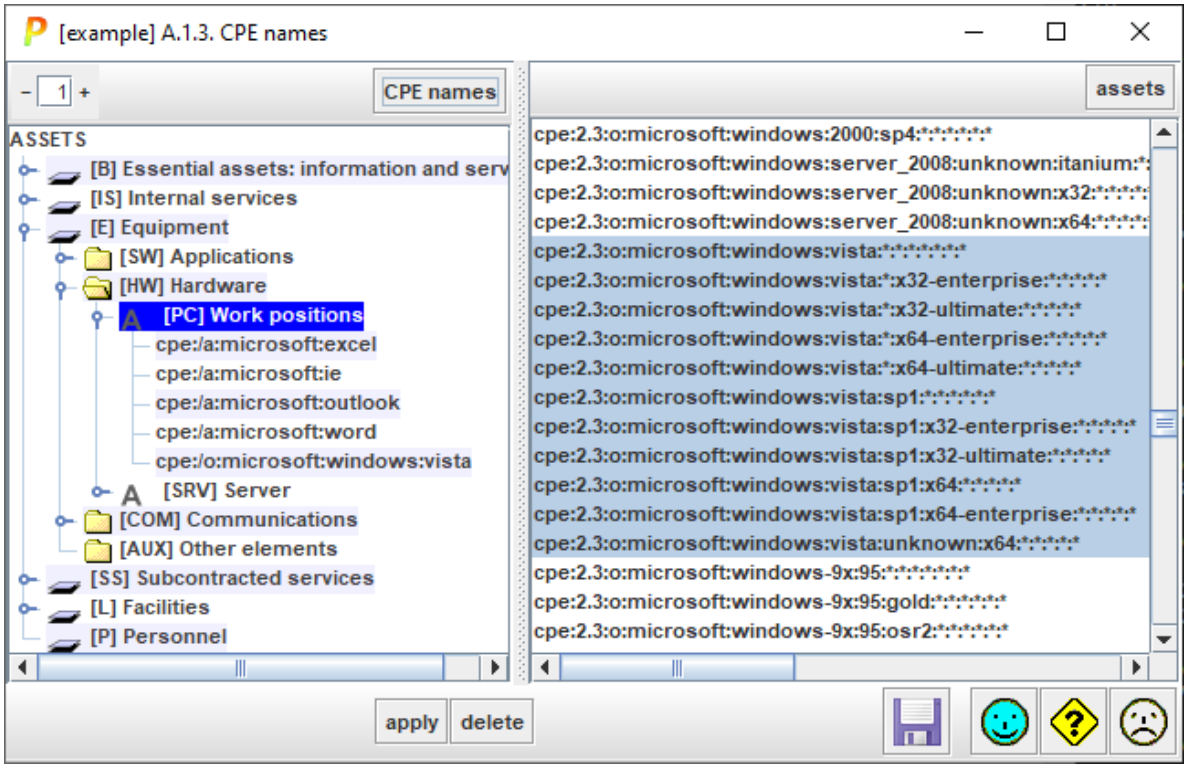

## 2.4 CVSS version 2 and version 3

PILAR reads both CVSS version 2 and version 3, if available. When both are present, CVSS version 3 is preferred.




## 2.5 Search by CPE

When CPE values are associated to assets




then you may search a feed for CVEs matching the assigned CPE names.

## 2.6 UPDATE CVEs

UPDATE updates information on the selected CVEs.

# 3 CVEs in PILAR

CVEs may be associated to assets in PILAR. This association is translated into a security risk. The details are described below.

## 3.1 Summary

CVSS metrics are used to estimate likelihood and a consequences (degradation) for PILAR:

| metric | likelihood | degradation |
|---|---|---|
| attack vector | yes | |
| attack complexity | yes | |
| privileges required | yes | |
| user interaction | yes | |
| scope | | |
| confidentiality impact | | yes |
| integrity impact | | yes |
| availability impact | | yes |
| exploitability | yes | yes |
| remediation level | yes | yes |
| report confidence | yes | yes |

## 3.2 New attributes

CVSS attributes are described in Annex.

**Attacker level (AL)**

This metric makes an estimate of the strength of the attacker.

| SL | Target | Skills | Motivation | Means | Resources |
|---|---|---|---|---|---|
| **SL1** | Casual or coincidental violations | No Attack Skills | Mistakes | Non-intentional | Individual |
| **SL2** | Cybercrime, Hacker | Generic | Low | Simple | Low (Isolated Individual) |
| **SL3** | Hacktivist, Terrorist | Specific | Moderate | Sophisticated (Attack) | Moderate (Hacker Group) |
| **SL4** | Nation State | Specific | High | Sophisticated (Campaign) | Extended (Multi-disciplinary Teams) |

**Attack surface (AS)**

This metric makes an estimate of how many points has an attacker to perform the attack.

| | |
|---|---|
| **one (1)** | There is only one (1) asset subject to the attack. |
| **Narrow (N)** | Very few assets on which the attack is feasible.<br>Less than 10. |
| **Medium (M)** | A few assets on which the attack is feasible.<br>Between 10 and 100. |

| | |
|---|---|
| **Large (L)** | Many assets on which the attack is feasible.<br>More than 100. |

**Exposure (PE)**

This metric captures the exposure of the system; that is, how exposed the attack points are.

| | |
|---|---|
| **Very low (VL)** | The attacker must traverse a professional border protection system.<br>Maturity level L4 or higher. |
| **Low (L)** | The attacker must traverse an ordinary border protection system.<br>Maturity level L3. |
| **Medium (M)** | The attacker must traverse an ad-hoc (weak) border protection system.<br>Maturity level L2 or L1. |
| **High (H)** | There is no border protection system: open access<br>Maturity level: L0. |

**History (H)**

This metric captures the actual observation of incidents.

| | |
|---|---|
| ~~**Never (N)**~~ | ~~There is no registry of the incident happening in the system.~~ |
| **Rare (R)** | Unseen |
| **Occasional (O)** | The incident has been observed in the last 1-year period. |
| **Frequent (F)** | Regular attacks (CERT report) |

**Associated threat**

We associate a threat from PILAR to model the kind of attack. The most usual ones are

| threat | A | I | C |
|---|---|---|---|
| [A.3] Manipulation of activity records (log) | N | N | N |
| [A.4] Manipulation of configuration | L | M | H |
| [A.5] Masquerading of identity | N | H | H |
| [A.6] Abuse of access privileges | N | M | H |
| [A.11] Unauthorized access | N | L | H |
| [A.15] Deliberate alteration of information | N | H | M |
| [A.18] Destruction of information | H | N | N |
| [A.19] Disclosure of information | N | N | H |
| [A.22] Software manipulation | L | L | M |
| [A.24] Denial of service | H | N | N |

## 3.3 Likelihood

PILAR assigns a likelihood to the chances that a CVE is exploited on an asset.

PILAR uses ARO (Annual Rate of Occurrence) as metric. The ARO associated to a CVE is derived from its CVSS metrics.

ARO is estimated as

1 * AV * AC * PR * UI * E * RL * RC

using the following coefficients:

| AV | AC | PR | UI | E | RL | RC |
|---|---|---|---|---|---|---|
| X: 100% | X: 100% | X: 100% | X: 100% | X: -50% | X: 100% | X: -50% |
| P: -50% | H: -50% | N: 10 | N: 2 | U: -50% | O: 0% | U: -50% |
| L: 100% | L: 2 | L: 100% | R: -50% | P: -90% | T: -90% | R: -20% |
| A: 2 | | H: -67% | | F: -10% | W: -50% | C: 100% |
| N: 10 | | | | H: 100% | U: 100% | |

The codes are explained in the annex.

Where

- X stands for 'not defined'.
- red background means that the value increases

Some standard values in the table translated into a multiplying coefficient

| value | coefficient |
|---|---|
| 10 | 10.0 |
| 2 | 2.0 |
| 100% | 1.0 |
| -10% | 0.9 |
| -20% | 0.8 |
| -50% | 0.5 |
| -90% | 0.1 |
| -99% | 0.01 |
| 0 | 0.0 |

Then, we add our additional attributes

| *AL* | *AS* | *PE* | *H* |
|---|---|---|---|
| X: 100% | X: 100% | X: 100% | X: 100% |
| SL1: -99% | 1: 100% | VL: -99% | R: -60% |
| SL2: -90% | N: 2.0 | L: -90% | O: +30% |
| SL3: 100% | M: 3.0 | M: -20% | F: 5.0 |
| SL4: 10 | L: 4.0 | H: 100% | |

## 3.4 Degradation

Degradation is the percentage of value that is lost because of the incident. 0% means no loss (that is, no consequences), 100% means the value is completely lost.

PILAR uses impact metrics to derive degradation,

degradation = impact * E + RL * RC

using an ‘order of magnitude’ scale:

| impact | E | RL | RC |
|---|---|---|---|
| N: 0% | X: -50% | X: 100% | X: -50% |
| L: -90% | U: -50% | O: 0% | U: -50% |
| H: 100% | P: -90% | T: -90% | R: -20% |
| | F: -10% | W: -50% | C: 100% |
| | H: 100% | U: 100% | |

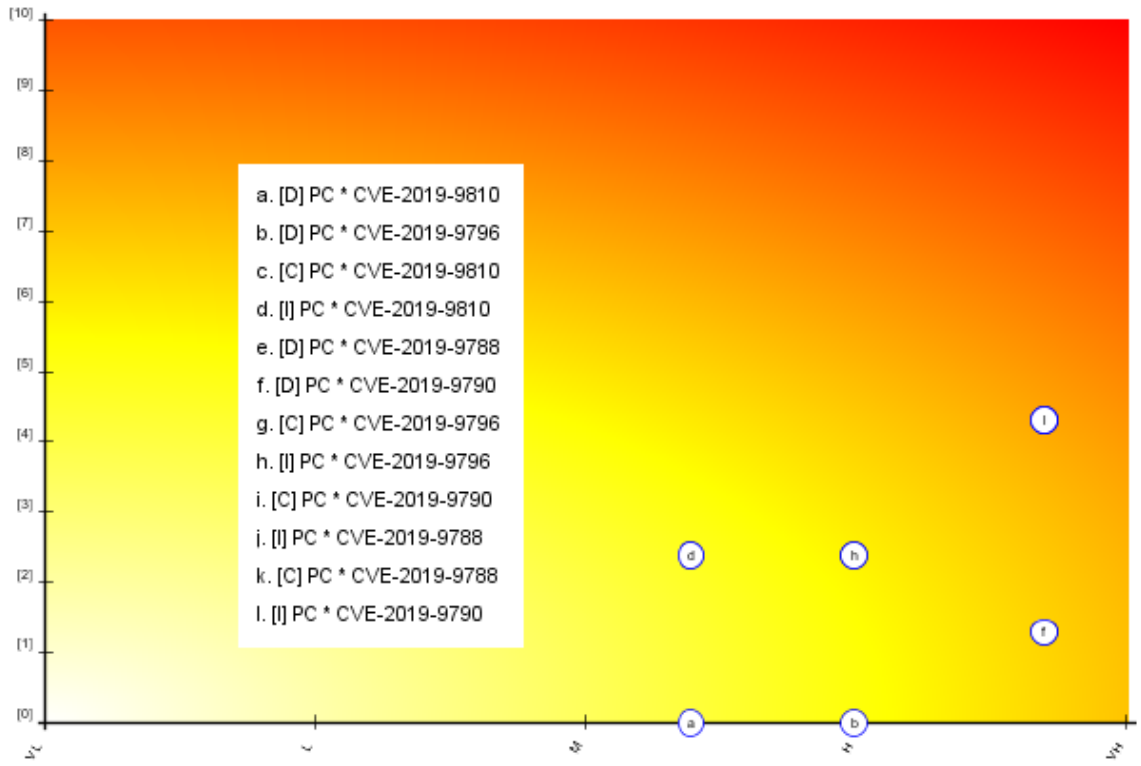
## 3.5 Risk

Risk is calculated as usual in PILAR

impact = accumulated_value * degradation;

risk = impact * likelihood;

Please, note that for qualitative risk, a heat map table is used, assigning a higher weight to impact and a lower one to likelihood. Refer to Magerit for long reasoning.




# 4 CVSS fields

Information taken from https://www.first.org/cvss/ version 3.1.

**Attack vector (AV)**

This metric reflects the context by which vulnerability exploitation is possible.

| | |
|---|---|
| **Network (N)** | The vulnerable component is bound to the network stack and the set of possible attackers extends beyond the other options listed below, up to and including the entire Internet. Such a vulnerability is often termed "remotely exploitable" and can be thought of as an attack being exploitable *at the protocol level* one or more network hops away (e.g., across one or more routers). An example of a network attack is an attacker causing a denial of service (DoS) by sending a specially crafted TCP packet across a wide area network (e.g., CVE-2004-0230). |
| **Adjacent (A)** | The vulnerable component is bound to the network stack, but the attack is limited *at the protocol level* to a logically adjacent topology. This can mean an attack must be launched from the same shared physical (e.g., Bluetooth or IEEE 802.11) or logical (e.g., local IP subnet) network, or from within a secure or otherwise limited administrative domain (e.g., MPLS, secure VPN to an administrative network zone). One example of an Adjacent attack would be an ARP (IPv4) or neighbor discovery (IPv6) flood leading to a denial of service on the local LAN segment (e.g., CVE-2013-6014). |
| **Local (L)** | The vulnerable component is not bound to the network stack and the attacker's path is via read/write/execute capabilities. Either:<br>• the attacker exploits the vulnerability by accessing the target system locally (e.g., keyboard, console), or remotely (e.g., SSH); *or*<br>• the attacker relies on User Interaction by another person to perform actions required to exploit the vulnerability (e.g., using social engineering techniques to trick a legitimate user into opening a malicious document). |
| **Physical (P)** | The attack requires the attacker to physically touch or manipulate the vulnerable component. Physical interaction may be brief (e.g., evil maid attack) or persistent. An example of such an attack is a cold boot attack in which an attacker gains access to disk encryption keys after physically accessing the target system. Other examples include peripheral attacks via FireWire/USB Direct Memory Access (DMA). |

**Attack complexity (AC)**

This metric describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability.

| | |
|---|---|
| **Low (L)** | Specialized access conditions or extenuating circumstances do not exist. An attacker can expect repeatable success when attacking the vulnerable component. |
| **High (H)** | A successful attack depends on conditions beyond the attacker's control. That is, a successful attack cannot be accomplished at will, but requires the attacker to invest in some measurable amount of effort in preparation or execution against the vulnerable |

| | |
|---|---|
| | component before a successful attack can be expected. For example, a successful attack may depend on an attacker overcoming any of the following conditions:<br><br>• The attacker must gather knowledge about the environment in which the vulnerable target/component exists. For example, a requirement to collect details on target configuration settings, sequence numbers, or shared secrets.<br>• The attacker must prepare the target environment to improve exploit reliability. For example, repeated exploitation to win a race condition, or overcoming advanced exploit mitigation techniques.<br>• The attacker must inject themselves into the logical network path between the target and the resource requested by the victim in order to read and/or modify network communications (e.g., a man in the middle attack). |

**Privileges required (PR)**

This metric describes the level of privileges an attacker must possess *before* successfully exploiting the vulnerability.

| | |
|---|---|
| **None (N)** | The attacker is unauthorized prior to attack, and therefore does not require any access to settings or files of the vulnerable system to carry out an attack. |
| **Low (L)** | The attacker requires privileges that provide basic user capabilities that could normally affect only settings and files owned by a user. Alternatively, an attacker with Low privileges has the ability to access only non-sensitive resources. |
| **High (H)** | The attacker requires privileges that provide significant (e.g., administrative) control over the vulnerable component allowing access to component-wide settings and files. |

**User interaction (UI)**

This metric captures the requirement for a user, other than the attacker, to participate in the successful compromise of the vulnerable component. This metric determines whether the vulnerability can be exploited solely at the will of the attacker, or whether a separate user (or user-initiated process) must participate in some manner.

| | |
|---|---|
| **None (N)** | The vulnerable system can be exploited without interaction from any user. |
| **Required (R)** | Successful exploitation of this vulnerability requires a user to take some action before the vulnerability can be exploited. For example, |

| | |
|---|---|
| | a successful exploit may only be possible during the installation of an application by a system administrator. |

**Scope (S)**

the ability for a vulnerability in one software component to impact resources beyond its means, or privileges.

Formally, Scope refers to the collection of privileges defined by a computing authority (e.g. an application, an operating system, or a sandbox environment) when granting access to computing resources (e.g. files, CPU, memory, etc).

| | |
|---|---|
| **Unchanged (U)** | An exploited vulnerability can only affect resources managed by the same security authority. In this case, the vulnerable component and the impacted component are either the same, or both are managed by the same security authority. |
| **Changed (C)** | An exploited vulnerability can affect resources beyond the security scope managed by the security authority of the vulnerable component. In this case, the vulnerable component and the impacted component are different and managed by different security authorities. |

**Confidentiality Impact**

| | |
|---|---|
| **High (H)** | There is a total loss of confidentiality, resulting in all resources within the impacted component being divulged to the attacker. Alternatively, access to only some restricted information is obtained, but the disclosed information presents a direct, serious impact. For example, an attacker steals the administrator's password, or private encryption keys of a web server. |
| **Low (L)** | There is some loss of confidentiality. Access to some restricted information is obtained, but the attacker does not have control over what information is obtained, or the amount or kind of loss is limited. The information disclosure does not cause a direct, serious loss to the impacted component. |
| **None (N)** | There is no loss of confidentiality within the impacted component. |

**Integrity Impact**

| | |
|---|---|
| **High (H)** | There is a total loss of integrity, or a complete loss of protection. For example, the attacker is able to modify any/all files protected by the impacted component. Alternatively, only some files can be modified, but malicious modification would present a direct, serious consequence to the impacted component. |

| | |
|---|---|
| **Low (L)** | Modification of data is possible, but the attacker does not have control over the consequence of a modification, or the amount of modification is limited. The data modification does not have a direct, serious impact on the impacted component. |
| **None (N)** | There is no loss of integrity within the impacted component. |

**Availability Impact**

| | |
|---|---|
| **High (H)** | There is a total loss of availability, resulting in the attacker being able to fully deny access to resources in the impacted component; this loss is either sustained (while the attacker continues to deliver the attack) or persistent (the condition persists even after the attack has completed). Alternatively, the attacker has the ability to deny some availability, but the loss of availability presents a direct, serious consequence to the impacted component (e.g., the attacker cannot disrupt existing connections, but can prevent new connections; the attacker can repeatedly exploit a vulnerability that, in each instance of a successful attack, leaks a only small amount of memory, but after repeated exploitation causes a service to become completely unavailable). |
| Low (L) | Performance is reduced or there are interruptions in resource availability. Even if repeated exploitation of the vulnerability is possible, the attacker does not have the ability to completely deny service to legitimate users. The resources in the impacted component are either partially available all of the time, or fully available only some of the time, but overall there is no direct, serious consequence to the impacted component. |
| None (N) | There is no impact to availability within the impacted component. |

**Exploit code maturity (E)**

| | |
|---|---|
| Not Defined (X) | Assigning this value indicates there is insufficient information to choose one of the other values, and has no impact on the overall Temporal Score, i.e., it has the same effect on scoring as assigning High. |
| High (H) | Functional autonomous code exists, or no exploit is required (manual trigger) and details are widely available. Exploit code works in every situation or is actively being delivered via an autonomous agent (such as a worm or virus). Network-connected systems are likely to encounter scanning or exploitation attempts. Exploit development has reached the level of reliable, widely available, easy-to-use automated tools. |

| | |
|---|---|
| Functional (F) | Functional exploit code is available. The code works in most situations where the vulnerability exists. |
| Proof-of-Concept (P) | Proof-of-concept exploit code is available, or an attack demonstration is not practical for most systems. The code or technique is not functional in all situations and may require substantial modification by a skilled attacker. |
| Unproven (U) | No exploit code is available, or an exploit is theoretical. |

**Remediation level (RL)**

| | |
|---|---|
| Not Defined (X) | Assigning this value indicates there is insufficient information to choose one of the other values, and has no impact on the overall Temporal Score, i.e., it has the same effect on scoring as assigning Unavailable. |
| Unavailable (U) | There is either no solution available or it is impossible to apply. |
| Workaround (W) | There is an unofficial, non-vendor solution available. In some cases, users of the affected technology will create a patch of their own or provide steps to work around or otherwise mitigate the vulnerability. |
| Temporary Fix (T) | There is an official but temporary fix available. This includes instances where the vendor issues a temporary hotfix, tool, or workaround. |
| Official Fix (O) | A complete vendor solution is available. Either the vendor has issued an official patch, or an upgrade is available. |

**Report confidence (RC)**

| | |
|---|---|
| Not Defined (X) | Assigning this value indicates there is insufficient information to choose one of the other values, and has no impact on the overall Temporal Score, i.e., it has the same effect on scoring as assigning Confirmed. |
| Confirmed (C) | Detailed reports exist, or functional reproduction is possible (functional exploits may provide this). Source code is available to independently verify the assertions of the research, or the author or vendor of the affected code has confirmed the presence of the vulnerability. |
| Reasonable (R) | Significant details are published, but researchers either do not have full confidence in the root cause, or do not have access to source code to fully confirm all of the interactions that may lead to the result. Reasonable confidence exists, however, that |

| | |
|---|---|
| | the bug is reproducible and at least one impact is able to be verified (proof-of-concept exploits may provide this). An example is a detailed write-up of research into a vulnerability with an explanation (possibly obfuscated or "left as an exercise to the reader") that gives assurances on how to reproduce the results. |
| Unknown (U) | There are reports of impacts that indicate a vulnerability is present. The reports indicate that the cause of the vulnerability is unknown, or reports may differ on the cause or impacts of the vulnerability. Reporters are uncertain of the true nature of the vulnerability, and there is little confidence in the validity of the reports or whether a static Base Score can be applied given the differences described. An example is a bug report which notes that an intermittent but non-reproducible crash occurs, with evidence of memory corruption suggesting that denial of service, or possible more serious impacts, may result. |

# 5 Glossary

**Common Platform Enumeration (CPE)**

A nomenclature and dictionary of hardware, operating systems, and applications.

A SCAP specification that provides a standard naming convention for operating systems, hardware, and applications for the purpose of providing consistent, easily parsed names that can be shared by multiple parties and solutions to refer to the same specific platform type.

https://nvd.nist.gov/products/cpe

**Common Vulnerabilities and Exposures (CVE)**

A nomenclature and dictionary of security-related software flaws.

An SCAP specification that provides unique, common names for publicly known information system vulnerabilities.

https://cve.mitre.org/

**Common Vulnerability Scoring System (CVSS)**

A system for measuring the relative severity of software flaw vulnerabilities.

An SCAP specification for communicating the characteristics of vulnerabilities and measuring their relative severity.

https://www.first.org/cvss/

**degradation**

Degradation measures the loss of value of an asset when a threat occurs.

**impact**

The magnitude of harm that can be expected to result from the consequences of unauthorized disclosure of information, unauthorized modification of information, unauthorized destruction of information, or loss of information or information system availability. [CNSSI_4009:2010]

**magerit**

Methodology for Information Systems Risk Analysis and Management.

https://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Magerit.html?idioma=en

**likelihood**

In Information Assurance risk analysis, a weighted factor based on a subjective analysis of the probability that a given threat is capable of exploiting a given vulnerability. [CNSSI_4009:2010]

**risk**

A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence.

Framework for Improving Critical Infrastructure Cybersecurity, National Institute of Standards and Technology, February 12, 2014

**Security Content Automation Protocol (SCAP)**

The Security Content Automation Protocol (SCAP) is a method for using specific standards to enable automated vulnerability management, measurement, and policy compliance evaluation of systems deployed in an organization, including e.g., FISMA compliance. The National Vulnerability Database (NVD) is the U.S. government content repository for SCAP. An example of an implementation of SCAP is OpenSCAP.

**vulnerability**

In computer security, a vulnerability is a weakness which can be exploited by a threat actor, such as an attacker, to perform unauthorized actions within a computer system. To exploit a vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. In this frame, vulnerability is also known as the attack surface.

# 6 Help tool

A simple tool is available to experiment with the attribution of a vulnerability