Quick Start Guide for CPE Applicability Statements in the CVE Record Format

Version 1.1
Last Updated: 2025-1-21

1 Overview

The CVE Record Format now supports an optional JSON structure that (mostly) mirrors the NVD “configurations” array. This array allows the definition of CPE applicability statements. Within the CVE Record Format, this array has been named “cpeApplicability” to avoid conflict with the existing and unrelated “configurations” array. This document is meant to help those choosing to use the new “cpeApplicability” element to define CPE applicability statements within CVE Records.

> NOTE: Please note that this document is a work-in-progress, and the authors welcome pertinent feedback and commentary. Issues with formatting and images will be resolved as this document is transitioned to a more final form. The authors apologize for any temporary inconveniences or accessibility issues.

2 What is Common Platform Enumeration (CPE)?

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 naming format used in this document is referred to as the Formatted String Binding in the CPE Naming Specification (http://csrc.nist.gov/publications/nistir/ir7695/NISTIR-7695-CPE-Naming.pdf) and we will use this format throughout this document and within the CVE Record Format. This format uses product attributes in a fixed order, separated by the colon character:

cpe:2.3: part : vendor : product : version : update : edition : language : sw_edition : target_sw : target_hw : other

An example of a CPE Name using this format would be:

cpe:2.3:a:hp:openview_network_manager:7.51::*:*:*:linux:*:**

The above example is a CPE Name for the HP OpenView Network Manager application version 7.51 operating within a Linux environment. Also note that the version of CPE used for this CPE Name is 2.3 and is included as part of the CPE Name string. The CVE Record Format only supports CPE 2.3 Names in the cpeApplicability array.

For additional understanding of CPE, including how to search the CPE Dictionary or contribute to its maintenance, please refer to the official Common Platform Enumeration (CPE) Specifications.

3 Terminology

CPE is perhaps more complicated than it first appears. It is important to understand and use the proper names for different parts and aspects of CPE. The terms are summarized here and further explained later.

CPE (Identifier) Name

A Common Platform Enumeration (CPE) Name is a unique string used to identify a specific enumeration of a platform. This CPE Name represents a specific enumeration of known hardware, applications or operating systems that exist within the Official CPE Dictionary for platforms of interest. CPE Names must adhere to certain acceptance criteria to be admissible in the Official CPE Dictionary.
Acceptance Criteria:

- CPE Names MUST have the Part, Vendor, Product and Version attributes populated at a minimum to be valid.
- CPE Names MUST NOT use the NA logical value (“-”) as the Part, Vendor or Product attributes.
- CPE Names MUST NOT use the ANY logical value (“*”) as the Part, Vendor, Product or Version attributes.
- CPE Names MUST NOT contain any asterisk or question mark characters within attribute values unless those characters are quoted (“*”, “\?”).

CPE Match Criteria

These are supersets of CPE Names. Unlike CPE Names, Match Criteria do not require certain attributes be populated to be valid and come in two representations:

- CPE Match String
- CPE Match String Range

CPE Match Criteria are what organizations include within the configurations of a CPE applicability statement to indicate platform applicability. Think of CPE Match Criteria (Match Strings and Match String Ranges) as a bridge between CVE records and CPE Names.

[CVE Record] **<-- CPE Match Criteria -->** **[CPE Names in CPE Dictionary]**

CVE ID

[CVE-2023-25727]

CPE Match String Range

Matches

CPE Match String

A CPE Match string is a formatted string binding that correlates to one or many CPE Names in the Official CPE Dictionary. CPE Match Strings look very similar to a CPE Name but have less restrictions on which attributes must be populated and are intended to reference one or more CPE Names within the CPE Dictionary.

For example, a CPE Match String could be defined that is intended to match all CPE Names for a given vendor:

cpe:2.3:a:some_vendor::*:*:*:*:*:*:*:**

Another CPE Match String example would be to match all CPE Names for a given vendor and product:

cpe:2.3:a:some_vendor:product_x::*:*:*:*:*:*:**

CPE Match String Range

A CPE Match String Range looks very similar to a CPE Match String but is defined with a CPE Base String and separate boundaries for the affected version attributes(s). The CPE Base String and affected version attributes are combined to define the affected products and version ranges. The CPE Base String must not contain any values in the version or update attributes but may contain values in any other attribute. CPE Match String Ranges can represent <, >, <=, >= within a boundary. Unlike CPE Match Strings, CPE Match String Ranges require both CPE Name Matching logic in addition to customized logic to determine CPE Name matches.

cpe:2.3:a:some_vendor:product_x::*:*:*:*:*:*:*
versionStartIncluding: 1.0
versionEndExcluding: 1.5*

In the above example, the first part of the CPE Match String Range is the CPE Base String. The CPE Base String specifies the vendor and platform without any version information. The version information is provided using the Starting and Ending versions. Note that versionStartIncluding means that the starting version is 1.0 and that 1.0 is also part of the range match. versionStartExcluding can also be used to start a range that would not include 1.0 but anything after (e.g., 1.01 or 1.1). The same can be used for the end of the version range, and in this case 1.5 is the end of the range but is excluded. In practice this would probably mean that 1.5 is the fixed version.

CPE Base String

The Formatted String Binding of a CPE Match String that represents a pre-determined specificity for relevant CPE Names. CPE Base Strings SHALL NOT contain Version or Update attribute values other than "*". CPE Base Strings are used when constructing a CPE Match String Range or when consolidating returned results for CPE Name searches. See the previous CPE Match String Range section for an example of a CPE Base String.

CPE Applicability Statement

The CPE Applicability Language data model provides the functionality required to allow CPE users to construct complex groupings of CPE names to describe IT platforms. These groupings are referred to as applicability statements because they are used to designate which platforms particular guidance, policies, etc. apply to. In the case of CVE Records, the CPE Applicability Statements help to define which products and platforms are affected (or not affected) by the vulnerability described in the CVE Record.

CPE Applicability Statements consist of configurations (i.e., cpeApplicability), which contain nodes, which contain match criteria, which match CPE Names.

4 cpeApplicability Structure

Where is cpeApplicability defined?

The cpeApplicability array is defined within the CVE Record Format JSON cnaContainer at the same level as the affected, description, problemType, etc. An example of a cnaContainer with a cpeApplicability array is provided below.

```
{
  "cnaContainer": {
    "title": "Simple Example CVE Record",
    "datePublic": "2024-12-04T08:00:00+00:00",
    "cpeApplicability": [
      {
        "nodes": [
          {
            "operator": "OR",
            "negate": false,
            "cpeMatch": [
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:1.0.0:*:*:*:*:*:*:*"
              }
            ]
          }
        ]
      }
    ],
    "affected": [
      {
        "vendor": "Acme",
        "product": "Brick",
        "versions": [
          {
            "status": "affected",
            "version": "1.0.0",
            "versionType": "custom"
          }
        ],
        "defaultStatus": "unaffected"
      }
    ],
    "descriptions": [
      {
        "lang": "en",
```

The cpeApplicability array can include one or more configurations. Each configuration includes one or more “nodes”. Each node array item can include an operator property (AND or OR), a negate property (true or false), and a cpeMatch array. The cpeMatch array can contain one or more CPE Match Criteria (using the criteria property) and a vulnerable property (true or false). It can also include version ranges. The version range properties include versionStartExcluding, versionStartIncluding, versionEndExcluding, and versionEndIncluding. Use of these properties will be covered in the examples in the following sections.

Some other things to keep in mind are that an operator and/or negate property can be used at the nodes array level if more than one nodes array item is defined. For example, if multiple affected products running on multiple different platforms are defined within the CVE Record, you would need a way to tie them together with AND and OR. An example of this will be provided in the Advanced example section that follows.

When a CVE Record affects multiple products on multiple different platforms, it may be necessary to create multiple configurations within the cpeApplicability array. Note that when creating multiple configurations, the configurations are evaluated using an implicit OR. Advanced Example #4 in the Advanced Examples section 6 provides an example using multiple configurations.

Name	Type	Description
cpeApplicability	array	Top level array that can contain one or more configurations. The cpeApplicability array is defined within the CVE Record Format JSON cnaContainer at the same level as the affected, description, problemType, etc. cpeApplicability array items use an implicit OR.
operator	property - string - enumeration	Possible values are AND or OR. This property is required within a nodes array item and designates how multiple cpeMatch Names and Strings are evaluated together. Can optionally be used within a configuration array and designates how multiple nodes array items are evaluated together.
negate	Boolean	Optional property that can be used to invert the item specified. Similar to the operator property, negate can be specified at both the configuration and nodes levels. NVD generally includes this property in the nodes array items their records and sets to false.
nodes	array	Defines a CPE configuration node in an applicability statement. One or more nodes array items can exist within a configuration. Each nodes array item is required to have an operator property and one or more cpeMatch array items. The negate property can also be specified but is optional.

cpeMatch	array	Defines a CPE configuration cpeMatch in an applicability statement. One or more cpeMatch array items can exist within a nodes array item. The properties within a cpeMatch array item contain the CPE Names or Match Strings and the version ranges if defined. Each cpeMatch array item is required to have a vulnerable property and a criteria string (CPE Name or Match String), but can have other optional properties defined below.
vulnerable	Boolean	Required boolean property that designates whether the defined criteria represents a platform/CPE is vulnerable.
criteria	string	The criteria holds either a CPE Identifier Name or a CPE Match String using the CPE v2.3 format.
matchCriteriaId	uuid	This optional property can be used to define the NIST NVD Official CPE Dictionary matchCriteriaId if it is known.
versionStartExcluding	string	Used when defining a CPE Match String Range. Use this property to define the starting version but excluding the specified version. If this value is defined as 1.0, all versions after 1.0 would match the criteria, but not 1.0 itself as it is excluded.
versionStartIncluding	string	Used when defining a CPE Match String Range. Use this property to define the starting version and include the specified version. If this value is defined as 1.0, all versions starting with and including 1.0 would match the criteria.

versionEndExcluding	string	Used when defining a CPE Match String Range. Use this property to define the ending version but excluding the specified version. If this value is defined as 2.0, all versions before 2.0 would match the criteria, but not 2.0 itself as it is excluded. Use this property when defining the fixed version as the end of the version range.
versionEndIncluding	string	Used when defining a CPE Match String Range. Use this property to define the ending version and include the specified version. If this value is defined as 2.0, all versions before and including 2.0 would match the criteria.

5 Basic Examples

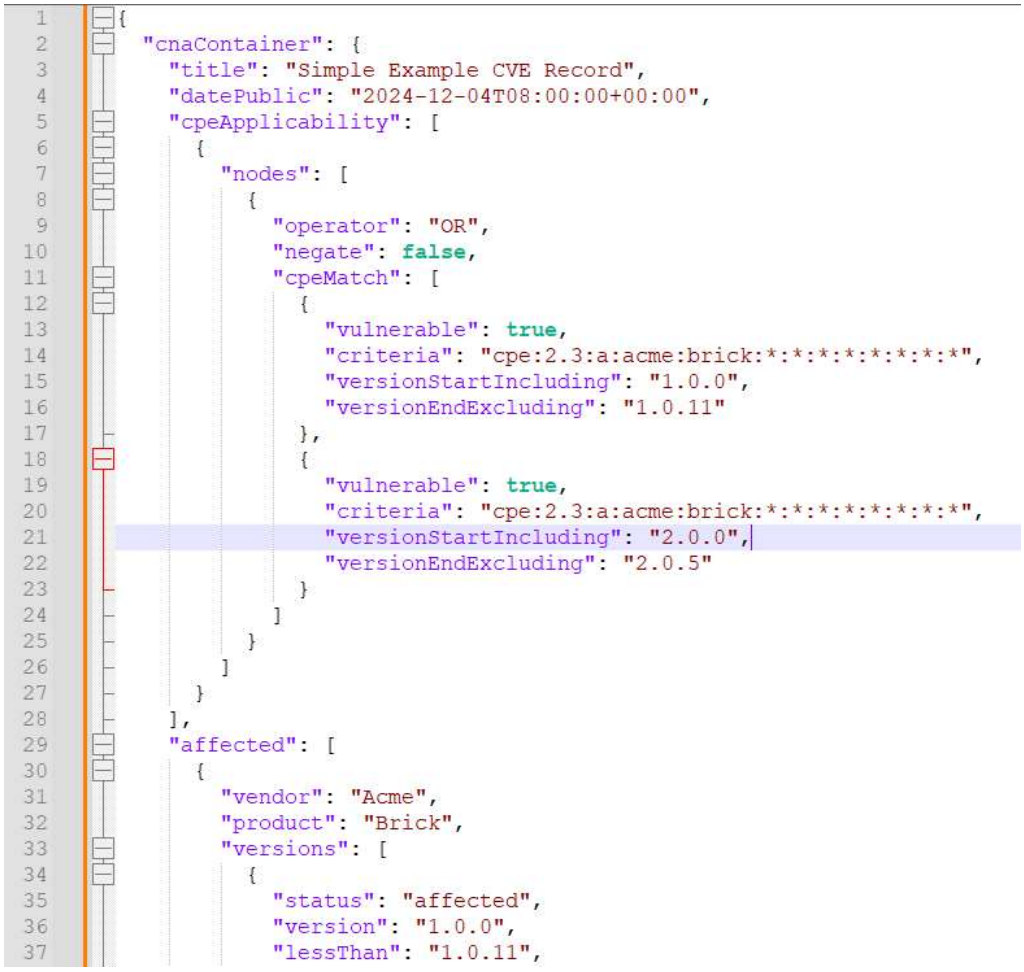
Basic Example #1

```
{
  "cnaContainer": {
    "title": "Simple Example CVE Record",
    "datePublic": "2024-12-04T08:00:00+00:00",
    "cpeApplicability": [
      {
        "nodes": [
          {
            "operator": "OR",
            "negate": false,
            "cpeMatch": [
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:1.0.0:*:*:*:*:*:*:*"
              }
            ]
          }
        ]
      }
    ],
    "affected": [
      {
        "vendor": "Acme",
        "product": "Brick",
        "versions": [
          {
            "status": "affected",
            "version": "1.0.0",
            "versionType": "custom"
          }
        ],
        "defaultStatus": "unaffected"
      }
    ],
    "descriptions": [
      {
        "lang": "en",
```

In this first basic example of defining a cpeApplicability array, you can see CPE data provided using the cpeApplicability array on the top, and the required affected array showing the affected versions immediately after. In all but a few rare cases, the data in cpeApplicability and affected should match. The main difference being that the cpeApplicability data will provide CPE Match Criteria, while the affected array will just provide the products and versions in their previously defined separate fields.

For this example, only a single version (1.0.0) of the defined product is affected. For the cpeApplicability array, only a single nodes array is needed as there is only one affected product and version to define. Under that nodes array is an operator, negate, and a cpeMatch array. The operator is a required property, but in this case it can be ignored since there is just one cpeMatch item. The negate property is NOT required, but can be provided for clarity. The cpeMatch array includes a vulnerable property with a value of true, which designates this cpeMatch as vulnerable. The criteria property is where the CPE Name or Match Criteria is provided. In this example case it is providing a CPE Identifier Name for a single product.

Basic Example #2



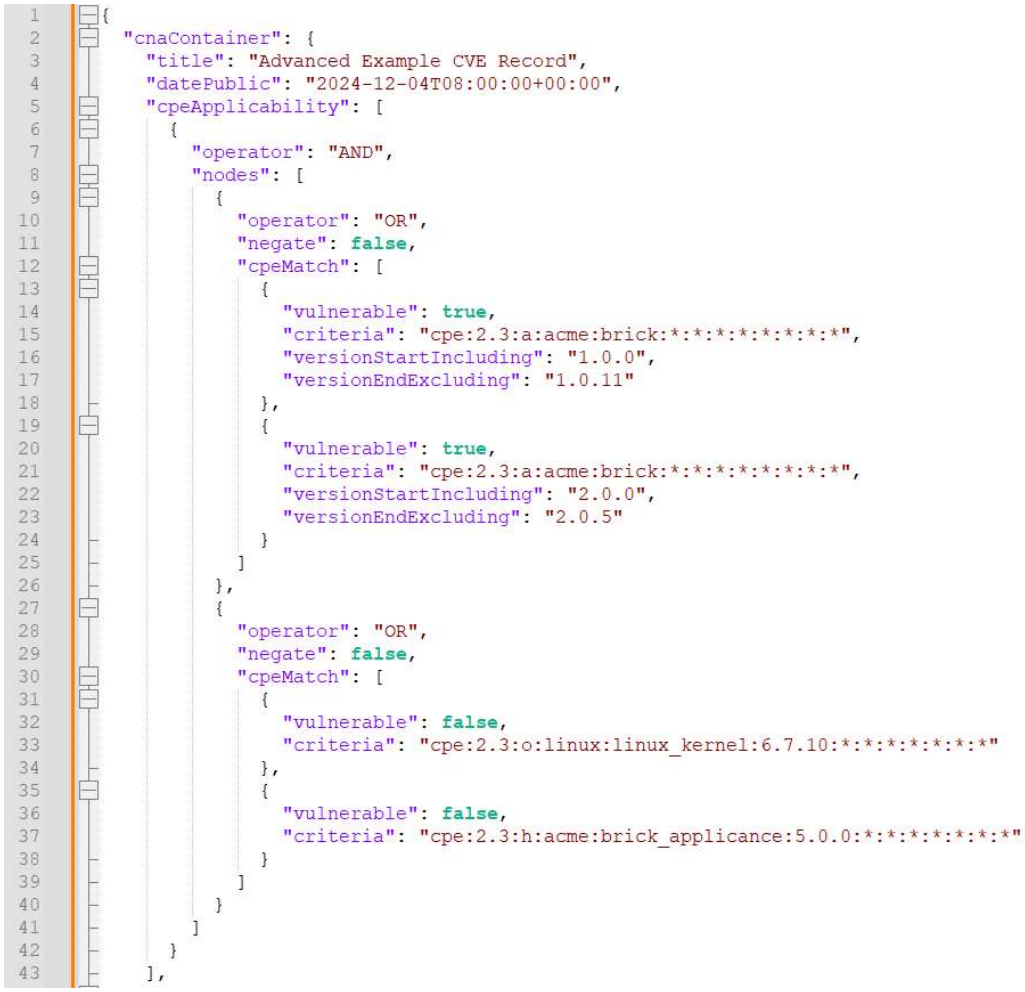
```
{
  "cnaContainer": {
    "title": "Simple Example CVE Record",
    "datePublic": "2024-12-04T08:00:00+00:00",
    "cpeApplicability": [
      {
        "nodes": [
          {
            "operator": "OR",
            "negate": false,
            "cpeMatch": [
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
                "versionStartIncluding": "1.0.0",
                "versionEndExcluding": "1.0.11"
              },
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
                "versionStartIncluding": "2.0.0",
                "versionEndExcluding": "2.0.5"
              }
            ]
          }
        ]
      }
    ],
    "affected": [
      {
        "vendor": "Acme",
        "product": "Brick",
        "versions": [
          {
            "status": "affected",
            "version": "1.0.0",
            "lessThan": "1.0.11",
```

In basic example #2, we have included multiple affected version ranges for the same product.In this case, note that the version is NOT included in the CPE Match Criteria defined in the criteria property. In this example case, the criteria is displaying a CPE Base String and the separate versionStartIncluding and versionEndExcluding represent the version range. Taken all together, this is what is referred to as a CPE Match String Range (see the Terminology section for the specific definitions).

The affected versions for this example case are 1.0.0 up to but not including 1.0.11 (1.0.11 is the fixed version), and 2.0.0 up to but not including 2.0.5 (2.0.5 is the fixed version). If no fixed version existed yet for the 2.0.x branch, you could just use “versionEndIncluding” instead of “versionEndExcluding.” Note that in this example case, the OR operator at the cpeMatch level is important and defines that the match should use an OR on the listed cpeMatch array items. This example would match on the 1.0.x or 2.0.x branch as defined above. Using an AND here would likely be wrong as a version of the platform from both the 1.0.x and 2.0.x branches would need to exist simultaneously for the record to be applicable.

6 Advanced Examples

Advanced Example #3 (Running on/With configurations)



```
{
  "cnaContainer": {
    "title": "Advanced Example CVE Record",
    "datePublic": "2024-12-04T08:00:00+00:00",
    "cpeApplicability": [
      {
        "operator": "AND",
        "nodes": [
          {
            "operator": "OR",
            "negate": false,
            "cpeMatch": [
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
                "versionStartIncluding": "1.0.0",
                "versionEndExcluding": "1.0.11"
              },
              {
                "vulnerable": true,
                "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
                "versionStartIncluding": "2.0.0",
                "versionEndExcluding": "2.0.5"
              }
            ]
          },
          {
            "operator": "OR",
            "negate": false,
            "cpeMatch": [
              {
                "vulnerable": false,
                "criteria": "cpe:2.3:o:linux:linux_kernel:6.7.10:*:*:*:*:*:*:*"
              },
              {
                "vulnerable": false,
                "criteria": "cpe:2.3:h:acme:brick_applicance:5.0.0:*:*:*:*:*:*:*"
              }
            ]
          }
        ]
      }
    ],
```

The main difference between this example case and the previous examples is that this example includes one configuration (i.e., one cpeApplicability array item) with two nodes array items. The nodes items are AND'd using the operator property at the nodes array level. Each nodes array item has two criteria specified. In this example we are looking for the same two vulnerable products from the Basic Example #2, but they also need to be running on either the linux kernel 6.7.10 OS, or the brick_applicance 5.0.0 for the record to be applicable.

Note that the vulnerable property is false for the linux and appliance criteria. The reason for this is because these platforms are NOT vulnerable themselves. It is the products and versions in the first cpeMatch array that are actually vulnerable for the given CVE Record example.

Advanced Example #4

```
"cpeApplicability": [
  {
    "operator": "AND",
    "nodes": [
      {
        "operator": "OR",
        "negate": false,
        "cpeMatch": [
          {
            "vulnerable": true,
            "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
            "versionStartIncluding": "1.0.0",
            "versionEndExcluding": "1.0.11"
          }
        ]
      },
      {
        "operator": "OR",
        "negate": false,
        "cpeMatch": [
          {
            "vulnerable": false,
            "criteria": "cpe:2.3:o:linux:linux_kernel:6.7.10:*:*:*:*:*:*:*"
          }
        ]
      }
    ]
  },
  {
    "operator": "AND",
    "nodes": [
      {
        "operator": "OR",
        "negate": false,
        "cpeMatch": [
          {
            "vulnerable": true,
            "criteria": "cpe:2.3:a:acme:brick:*:*:*:*:*:*:*:*",
            "versionStartIncluding": "2.0.0",
            "versionEndExcluding": "2.0.5"
          }
        ]
      },
      {
        "operator": "OR",
        "negate": false,
        "cpeMatch": [
          {
            "vulnerable": false,
            "criteria": "cpe:2.3:h:acme:brick_applicance:5.0.0:*:*:*:*:*:*:*"
          }
        ]
      }
```

This last example is similar to the previous and shows how the same data can be provided in different ways using multiple configurations (i.e., multiple cpeApplicability array items). This first configuration would identify the CVE record as applicable to the brick product 1.0.0 up to but not including 1.0.11 when it is running on linux kernel 6.7.10. The OR operators for each node, paired with individual cpeMatch arrays are used to describe how the content of each node should be evaluated. The node's OR indicates that any platform represented by a matched CPE Name should be considered applicable.

The second configuration would identify another applicability case of the brick product 2.0.0 up to but not including 2.0.5 when it is running on the brick_applicance 5.0.0. In the first cpeMatch of the second configuration, there might be 5 matching CPE Names within the CPE Dictionary with different versions (2.0.0, 2.0.1, 2.0.2, 2.0.3, and 2.0.4). The OR here also indicates that any platform represented by a matched CPE name should be considered a match in this case.

As mentioned in a previous section, the multiple configurations specified in this example use an implicit OR and there is no option to control the logic. The AND operator above the nodes property in this example applies to the content of the nodes and the cpeMatch within (e.g., “brick 1.0.0 - 1.0.11” AND “linux 6.7.10”).

Last thing to note on these advanced examples, example #3 is more efficient as it covers 4 cases (brick 1.0.x on linux 6.7.10, brick 2.0.x on linux 6.7.10, brick 1.0.x on brick_applicance 5.0.0, and brick 2.0.x on brick_applicance 5.0.0). Advanced example #4 only covers 2 of those cases (brick 1.0.x on linux 6.7.10 and brick 2.0.x on brick_applicance 5.0.0) but is a bit more lengthy to define. In some cases you might need this extra flexibility so it was provided here as an example. However, if all 4 are vulnerable then example #3 would be easier to define and interpret.

7 Frequently Asked Questions (FAQ)

7.1 What’s the difference between the cpeApplicability array and the previously existing cpes array within affected?

The previously existing cpes array did not allow CNAs to define how the CPE date should be interpreted or used. This led to varying ways of defining and using the cpes array. Sometimes, the list of CPEs in the cpes array might be one or more affected CPE Names, other times it might be one or more CPE Match Criteria for fixed versions of products, sometimes it might be a parent product that includes the affected package within the CVE Record.

The new cpeApplicability array allows CNAs to have much more control over how the CPE Match Criteria are defined and how they should be interpreted by a consumer of the data. This flexibility is provided using the CPE Applicability Statements format.

7.2 As a CNA, Vendor or Maintainer, how do I confirm if my products have CPE Name representation in the NIST NVD Official CPE Dictionary?

There are multiple methods (listed below) to query the NIST NVD official CPE Dictionary and determine if a CPE already exists for your product(s).

NVD CPE Search Tools / Feed Files

- NVD CPE Search Webpage

- NVD 2.0 API /cpes/ Endpoint
- NVD 2.0 API /cpematch/ Endpoint
- Legacy XML CPE 2.3 and 2.2 Dictionary Files
- Legacy CPE Match Feed File

Keyword Usage

Keyword search capabilities reference the content of each attribute (part, vendor, product, etc.) of any CPE Name, CPE Name Title(s), and CPE Name Provenance (reference) links. Older CPE Names may also contain legacy provenance type information which is also referenced by keyword searches.

If multiple keywords are provided, the search will function like an 'AND' statement. Returning results where all keywords exist somewhere in the referenced content, though not necessarily together.

Keyword searches operate as though a wildcard is placed after each keyword provided.
For example, providing "circle" will return results such as "circles" but not "encircle".

CPE Match String Usage

NVD CPE search tools allow users to leverage CPE Match Strings to review the contents of the CPE Dictionary.

Use Preceding and Trailing Asterisks

The CPE Search Webpage and 2.0 API /cpes/ endpoint offered by the NVD allow the use of preceding and trailing asterisks “*” within each attribute of a CPE Match String. The asterisk represents a wildcard, allowing much broader searches.

Start with Broad Search Strings

While more likely to occur with platforms that have been available over an extended timeline, platform representation within the CPE Dictionary could reference unexpected attribute values or can be fractured across multiple representations. Organizations should ensure broad coverage of search criteria prior to narrowing in on more specific Match Strings.

Examples:

	`cpe:2.3:*:*vendor1*:*:*:*:*:*:*:*:*`
	`cpe:2.3:*:*vend*:*:*:*:*:*:*:*:*`
	`cpe:2.3:*:*:*productname1*:*:*:*:*:*:*:*`
	`cpe:2.3:*:*:*prod*:*:*:*:*:*:*:*`
	`cpe:2.3:o:vendor1:productname1:-:*:*:*:enterprise:*:*`

7.3 Can I define my own CPE Names?

Yes, new CPE Names can be defined within CVE Records, especially if there is no coverage for the product within the NIST NVD Official CPE Dictionary. However, if the CPE Name for a given product already exists within the CPE Dictionary it is strongly encouraged that the Official CPE Dictionary Name be used.

CNAs may determine (through a search of the Official CPE Dictionary above) that their product(s) does not have representation in the Official CPE Dictionary. When this occurs, a request to have CPE Names added to the Official CPE Dictionary should be made to cpe_dictionary@nist.gov.

7.4 As a CNA, can I copy the CPE data provided by NIST NVD into my CVE Record?

Yes, the NIST NVD configurations defined within an NVD CVE JSON Record could be copied as is into a CVE Record (with the exception of the top level “configurations” -> “cpeApplicability” name change). Note that copying the NIST NVD configurations data would imply concurrence with the NIST NVD assessment of CPE information. Each organization is responsible for the accuracy of the data contributed within their container.

7.5 As a CNA, can I list unaffected versions of products within the cpeApplicability array of a CVE Record?

Yes, CPE Names can be provided along with a vulnerable property equal to false which would designate the platform as NOT vulnerable. However, NOT affected platforms should accompany one or more affected platforms in cpeApplicability since at least one affected platform/product is required within the affected array of a CVE Record.

7.6 Can I use Vulnogram to include the cpeApplicability array data?

Vulnogram does not currently support adding CPE data using the new cpeApplicability array, nor does it allow CPEs to be entered into the older cpes array. The cpeApplicability array can be added to a CVE Record manually within the JSON using any text editor.

7.7 Is the cpeApplicability array and CPE data required?

No, the new cpeApplicability array is optional at this point. The only required product and version data in a CVE Record is that which must be included in the affected array where at least one product and version string value must be included.

7.8 What’s the difference between the data in the affected array and the cpeApplicability array?

The affected array is the standard way of providing product and version data within a CVE Record. It allows a CNA to provide one or more products and versions using free form string values. The affected array is one of the minimum required pieces of data within a CVE Record. The cpeApplicability array allows products and versions to be provided using standardized CPE Applicability Statements. The cpeApplicability array is optional and is not required within a CVE Record.

8 Acknowledgements

Thanks go to the following individuals for their contributions to this document:

- Art Manion, ANALYGENCE Labs
- Megazone, F5
- Lisa Olson, Microsoft
- Christopher Turner, NIST