MyJVN Product Dictionary Challenge
for Collaboration with Vulnerability Database and Asset Management

Masato Terada (Information-technology Promotion Agency, Japan)
JVN and My Contributions

The Japan Vulnerability Notes (JVN) is a comprehensive and widely-used public vulnerability database consisting of JVN, JVN iPedia, and MyJVN, operated by IPA and JPCERT/CC in Japan.

My Contributions:
- 2002: Launched a research site that served as a predecessor to JVN.
- 2008: Launched MyJVN, a security automation platform.
- Current: Leading the development of extensions for MyJVN.
JVN, NVD and MyJVN

JVN utilizes NVD as an input source.

- From Japan
- From CSIRT community

JVN (JVN#12345678) Vulnerability Handling Coordination DB

JVN iPedia (JVNDB-yyyy-0123456) Vulnerability Archiving DB

MyJVN
Machine readable interface by Web APIs

NVD
MyJVN and CPE

MyJVN provides a machine-readable interface cooperating with its CPE.

mjcheck4
Filtered Security Information Tool

MyJVN VC
Version Checker

MyJVN VC
Version Checker
Motivation

- Enable collaboration with Vulnerability Databases (MyJVN) and asset management systems by utilizing Software Product Identification

- Specifically, address the challenges associated with Software Product Identification in the areas of:
  - Software Lifecycle
  - SBOM (Software Bill of Materials)
  - and Vulnerability Alert
Issues in Software Product Identification for the Software Lifecycle

SCAPv2 April Developer Days 2019 presentation highlights.

Need to identify:

- A specific software release as built
- Any applicable patches for a specific software release

Ideally, most software should be identified based on information generated during build and be released with the software.
Issues in Software Product Identification for the Software Lifecycle

SCAPv2 April Developer Days 2019 presentation highlights.

- ~5% of CPE Names are provided by software providers around the point of software “Release”
- ~95% of CPE Names are created by NVD analysts during the “Manage” phase
  - Produced during the vulnerability analysis process
  - Software is identified after a known vulnerability is found
- Identifying software after a vulnerability is discovered is way too late!
## Issues in Software Product Identification for SBOM

- Various ID specifications

<table>
<thead>
<tr>
<th>ID name</th>
<th>ID specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPE v2.3</td>
<td>cpe:2.3:{type}:{vendor}:{product}:{version}:{update}:{edition}:{lang}:{sw_edition}:{target_sw}:{target_hw}:{other}</td>
</tr>
<tr>
<td>purl</td>
<td>pkg:{type}/{name space}/{package name}@{version}?{qualifiers}#{subpath}</td>
</tr>
<tr>
<td>SWHIDs</td>
<td>swh:{schema version}:{object type}:{object ID}</td>
</tr>
<tr>
<td>SWID</td>
<td>globally unique value that shall be globally unique for every SWID tag created (Globally unique values may use a 16 byte GUID, or other globally unique value as defined by the tag creator)</td>
</tr>
</tbody>
</table>
Issues in Software Product Identification for SBOM

- Various combinations with Software Product Identifications and primary formats

<table>
<thead>
<tr>
<th>Format</th>
<th>ID</th>
<th>CPE</th>
<th>Package-Manager</th>
<th>SWHID(s)</th>
<th>Hash</th>
<th>SWID</th>
<th>UUID</th>
<th>Unique ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPDX</td>
<td>CPE field (CPE2.2, CPE2.3)</td>
<td>Package-Manager field (maven-central, npm, nuget, bower, purl)</td>
<td>swh field</td>
<td>Other field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CycloneDX</td>
<td>purl field</td>
<td>SWID field</td>
<td>Hash field</td>
<td>SWID field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SWID</td>
<td>tagid field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Issues in Software Product Identification for the Vulnerability Alert

MyJVN (Vulnerability DB) → Vulnerability Information (Product Name) → Vulnerability Information DB

Product Dictionary (CPE) → Reference (Product Name) → Product Dictionary DB

IT Platform → Installed Software (Product Name) → Software Inventory DB

Matching result → Vulnerability Alert → Matching result → IT asset management tool
Issues in Software Product Identification for the Vulnerability Alert

Matching between Vulnerability Information (Product Name) and Installed Software (Product Name) using Product Dictionary (CPE) as a reference to notify Vulnerability Alert.

- To challenge this approach, need to resolve the following issues:
  - Variations in notation between CPE and product names
  - Variations in version notation
Requirements for Software Product Identification Scheme

Considering the current state of product identification, it is necessary to satisfy the following three requirements.

1. Assigning procedure for PID (software Product IDentifier) to products that are not assigned PID
2. PID dissemination procedure of product that has been assigned PID
3. Minimize identifier spelling variations
Let’s construct MyJVN Product Dictionary!!

Proposal of Software Product Identification scheme for Collaboration with Vulnerability Database and Asset Management

- Software Product Identification scheme
  - Product Vendors register PID to MyJVN Product Dictionary
  - Provide the same PID to the installed Software (Product Name) and the data on the Vulnerability Database (Product Name)
Let’s construct MyJVN Product Dictionary!!

Software Product Identification scheme
Let’s construct MyJVN Product Dictionary!!

Proposal of Software Product Identification scheme for Collaboration with Vulnerability Database and Asset Management

- MyJVN Product Dictionary entry
  - Associate many PIDs that are CPE v2, Package-Manager, SWHIDs, Hash and SWID
  - Assign identifiers to products that do not have PIDs, and distribute those PIDs, while enabling global collaboration
Let’s construct MyJVN Product Dictionary!!

MyJVN Product Dictionary entry

<table>
<thead>
<tr>
<th>Vendor</th>
<th>MyJVN Vendor ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>VendorName</td>
<td></td>
</tr>
<tr>
<td>CPEVendorName</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prod</th>
<th>MyJVN Product ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prod</th>
<th>MyJVN Product ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prod</th>
<th>MyJVN Product ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prod</th>
<th>MyJVN Product ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prod</th>
<th>MyJVN Product ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod</td>
<td>MyJVN Product ID</td>
</tr>
</tbody>
</table>

Product Name

cpe (CPE2.3 value)
nvdpid (SWID(UUID) value)
vendorpid (Specific Identifier value)
spdxid (SPDX id value)
purl (purl value)
hash (Hash value)
Let’s construct MyJVN Product Dictionary!!

MyJVN Product Dictionary entry

```json
{
    "vendors": [
        {
            "vendor_id": "MyJVN Vendor ID",
            "vname": "Vendor Name",
            "cpe": "CPE Vendor Name",
            "products": [
                {
                    "product_id": "MyJVN Product ID",
                    "pname": "Product Name",
                    "product_ids": [
                        {"cpe": "CPE2.3 value"},
                        {"nvdpid": "SWID(UUID) value"},
                        {"vendorpid": "Vendor Specific Identifier value"},
                        {"spdxid": "SPDX id value"},
                        {"purl": "purl value"},
                        {"sha256": "Hash value"}
                    ]
                },
                {Product2},
                ...
            ]
        }
    ]
}
```

Using **MyJVN Product ID** associates various PIDs
Let’s construct MyJVN Product Dictionary!!

MyJVN Product Dictionary entry

```json
{
    "vendors": [
    {
        "vendor_id": "jvnpid:1.0:systembom",
        "vname": "システムボム (systembom)",
        "products": [
            {
                "product_id": "jvnpid:1.0:bomviewer:3.2.1.0.0",
                "pname": "ボムビューア (bomviewer)",
                "product_ids": [
                    {
                        "cpe": "cpe:2.3:a:systembom:bomviewer:3.2.1"
                    },
                    {
                        "nvpid": "65699569-EA51-4346-8BDC-4076FA5C0E72"
                    },
                    {
                        "vendorpid": "a4b2c1910774e07e3d254efca790e562"
                    },
                    {
                        "spdxid": "SPDXRef-systembom-bomviewer-3_2_1"
                    },
                    {
                        "purl": "pkg:rpm/systembom/bomviewer@3.2.1"
                    },
                    {
                        "sha256": "4BF460C97817ACE6418B49C4586C585955E0D9D4B6DD80CB7484398056631860"
                    }
                ]
            }]}
    },
    \{Product2\}, \{ ... \] }
}
```
Let’s construct MyJVN Product Dictionary!!

Proposal of Software Product Identification scheme for Collaboration with Vulnerability Database and Asset Management

● jvnpid (MyJVN Product ID)
  ● Administrative PID of MyJVN Product Dictionary to associate many PIDs
  ● Consider compatibility with CPE
  ● Support stylizing the version (fixed to 5 digits) format to achieve version comparison easily
Let’s construct MyJVN Product Dictionary!!

jvnpid

● Format

```
jvnpid:1.0:{vendor}:{product}:{version}:{update}:{edition}:{language}:{sw_edition}:{target_sw}:{target_hw}:{other}
```

● Sample

```
jvnpid:1.0:sysbom.sample.jp:bomgen:0.46.3.0.0
```
Let’s construct MyJVN Product Dictionary!!

jvnpid

- Specifying a range of affected versions using jvnpid

```json
{
  "product_status": {
    "known_affected": [
      {
        "product_id": "jvnpid:1.0:systembom:bom",
        "cpe": "cpe:2.3:a:systembom:bom",
        "versions": [
          { "at": "1.1.1.0.0" },
          { "greaterThanOrEqual": "1.33.2.0.0", "lessThan": "2.7.8.0.0" },
          { "greaterThan": "3.33.2.0.0", "lessThanOrEqual": "5.7.8.0.0" },
          { "lessThanOrEqual": "0.9.34.0.0" }
        ]
      }
    ]
  }
}
```
Let’s operate MyJVN Product Dictionary!!

Proposal of Software Product Identification scheme for Collaboration with Vulnerability Database and Asset Management

- Product not assigned PID
  - Acts as a tag generator and registration site
- Product assigned PID
  - Acts as a registration site
Let’s operate MyJVN Product Dictionary!!

Product not assigned PID

MyJVN Product Dictionary creates and registers

PID : ボムジェネレーションズ
jvnpid:1.0:systembom:bomgen

{ "vendors": [ {
  "vendor_id": "jvnpid:1.0:systembom",
  "vname": "システムボム (systembom)",
  "cpe": "",
  "products": [ {
    "product_id": "jvnpid:1.0:systembom:bomgen",
    "pname": "ボムジェネレーションズ (bomgen)",
    "product_ids": [ ]
  } ]
} ]

CPE is converted and generated based on jvnpid

Create, registers and disseminates PID
Let’s operate MyJVN Product Dictionary!!

Proposal of Software Product Identification scheme for Collaboration with Vulnerability Database and Asset Management

- Product not assigned PID
  - Acts as a tag generator and registration site
- Product assigned PID
  - Acts as a registration site
Let’s operate MyJVN Product Dictionary!!

Product assigned PID

Product Vendor creates and registers

**PID**: ボムビューア 3.2.1
65699569-EA51-4346-8BDC-4076FA5C0E72

Registers and disseminates PID given by Product Vendor

Let’s operate MyJVN Product Dictionary!!
Let’s operate MyJVN Product Dictionary!!

MyJVN and MyJVN Product Dictionary

MyJVN provides MyJVN Product Dictionary instead of CPE DB.

Filtered Security Information Tool

Extend of STIX and CSAF for MyJVN
MyJVN offers an extended JSON format of STIX and CSAF to facilitate collaboration.

Let’s operate MyJVN Product Dictionary!!

Extend of STIX/CSAF for MyJVN
Conclusion

We are currently reviewing the specifications of the MyJVN Product Dictionary for MyJVN and plan to develop it in 2023, with trial operation starting in 2024.

Our aim is to provide a solution hint for the global issues of Software Product Identification. We also look forward to your ideas for better collaboration.
Thank you!

https://linkedin.com/in/masatoterada/

IPA

Masato Terada
Researcher
Security Engineering Lab. IT Security Center
Information-technology Promotion Agency, Japan

BUNKYO GREEN COURT CENTER OFFICE
2-28-8 HONKOMAGOME, BUNKYO-KU
TOKYO, 113-6591 JAPAN

URL: https://www.ipa.go.jp