Tracking Vulnerabilities with Buildroot and Yocto

Arnout Vandecappelle

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

https://docs.google.com/presentation/d/1p5Y5X3u_6f48AndHk2M4C7pDhfpo7vU67dqdBOxwoc3E
Who is Arnout

Embedded software architect
Focus on Linux OS integration
Mind consultant since 2008
Worked for 40+ customers in multimedia, security, home automation, satellite, telecom, chips, ...
Buildroot maintainer (team of 5)
Tracking Vulnerabilities with Buildroot and Yocto

1. Why track vulnerabilities?
2. CVE and CPE databases
3. Tracking vulnerabilities in Buildroot
4. Tracking vulnerabilities in Yocto
5. Tracking vulnerabilities with (SPDX) SBoM
6. Evaluation
Why track vulnerabilities in embedded systems?

- IoT → every device is exposed
- Single device failure can bring down entire factory
- 40 new CVEs per day
- Software reuse → single attack applies to numerous devices
- Exposed vulnerability hurts sales
- Regulatory liability is coming
- Also in already released code, to supply timely updates
CVE and CPE databases

- People need to be informed about existing vulnerabilities
- CVE = Common Vulnerabilities and Exposures = system for identifying vulnerabilities
  Every vulnerability gets assigned a unique number
- NVD = National Vulnerability Database = US government (NIST) database with CVE information
  Many other databases exist, e.g. distro-specific databases
- CVE entry is very unstructured; no real way to identify which software (version) is affected
  ⇒ Additional database of software packages and versions, linked with CVE database
  = CPE = Common Platform Enumeration

\[
cpe:2.3:a:arm:mbed\_tls:2.28.0:*:*:*:*:*:*:*
\]

Database of CPE entries maintained by NIST
Every software version should have a separate CPE entry
Every CVE has a list of CPEs; version can be a range
Problems with CVE and CPE
But it’s the best we have!

- CPE doesn’t identify a version very well
  - Some software packages don’t do releases, or re-tag
  - Doesn’t take into account patched versions
  - CPE entry needs to be created manually for every release
  - No link to the actual software

- CVE’s CPE information often incorrect
  - Fixed version not (correctly) included in range
  - Missing CPE information
  - Make corrections! [https://nvd.nist.gov/info/contact-form](https://nvd.nist.gov/info/contact-form)
CVE sometimes has incorrect CPE information

CVE-2021-45450 Detail

[Description]
In Mbed TLS before 2.28.0 and 3.x before 3.1.0, `psa_cipher_generate_iv` and `psa_cipher_encrypt` allow policy bypass or oracle-based decryption when the output buffer is at memory locations accessible to an untrusted application.

[Known Affected Software Configurations]

<table>
<thead>
<tr>
<th>Configuration 1</th>
<th>From (including)</th>
<th>Up to (excluding)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpe:2.3:a:arm:mbed_tls:<em>:</em>:<strong><strong>:</strong></strong></td>
<td>2.22.0</td>
<td>3.1.0</td>
</tr>
</tbody>
</table>

2.28.0 falls in this range

<table>
<thead>
<tr>
<th>Configuration 2</th>
<th>Show Matching CPE(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpe:2.3:o:fedora:project:fedora:36:<em>:</em>:<strong><strong>:</strong></strong></td>
<td></td>
</tr>
<tr>
<td>cpe:2.3:o:fedora:project:fedora:37:<em>:</em>:<strong><strong>:</strong></strong></td>
<td></td>
</tr>
</tbody>
</table>

- Actually fixed in both Fedora 36 and 37

https://nvd.nist.gov/vuln/detail/CVE-2021-45450
Tracking vulnerabilities with Buildroot

make pkg-stats

- Download NVD CVE and CPE database as JSON files
  - Database is cached for 24h
- Cross-reference *selected* packages based on CPE info
- Check version ranges in CPE info
- Apply exclusions
- (also other, unrelated package info)
- Write result to JSON and HTML
<table>
<thead>
<tr>
<th>Package</th>
<th>Current version</th>
<th>CVEs</th>
<th>CVEs Ignored</th>
<th>CPE ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>package/ninja/ninja.mk</td>
<td>1.11.1.g95dee.kitwar...</td>
<td>CVE-2021-4336</td>
<td></td>
<td>no verified CPE identifier (Search)</td>
</tr>
<tr>
<td>package/giflib/giflib.mk</td>
<td>5.2.1</td>
<td>CVE-2022-28506</td>
<td></td>
<td>cpe:2.3:a:giflib_project:giflib:5.2.1:<em>:</em>:<em>:</em>:<em>:</em>:*</td>
</tr>
<tr>
<td>package/busybox/busybox.mk</td>
<td>1.36.0</td>
<td>N/A</td>
<td>CVE-2022-28391</td>
<td>cpe:2.3:a:busybox:busybox:1.36.0:<em>:</em>:<em>:</em>:<em>:</em>:*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CPE version unknown in CPE database (Search)</td>
</tr>
</tbody>
</table>

Tracking vulnerabilities with Buildroot and Yocto

Arnout Vandecappelle - ELC 2023
Buildroot vulnerabilities features

- Per package list of CVE entries with link to NIST database
- CVE match based on version range
- Per package CPE information (vendor, product, version) with automatic fallback
- Manually maintained CVE exclusion list
  - Doesn’t exist in Buildroot (e.g. due to distro patch)
  - Patched in Buildroot
  - Vulnerable code not built in Buildroot

CVE-2017-5834
CVE-2017-5835
CVE-2017-5836

CPE-2.3:a:libimobiledevice:
libplist:2.2.0:*:*:*:*:*:*:

no verified CPE identifier [Search]

CVE-2022-28391
Buildroot vulnerabilities limitations

- Vulnerability info is not generated automatically
- Severity analysis (CVSS) not included
- Need full Buildroot source to generate vulnerability list
  - Including config and custom package definition
- No separation of build-only packages
- Exclusions are in Buildroot source
  - Need to modify source for CVEs discovered later
  - Conditional exclusions often not implemented
  - No way to record configuration-specific exclusions
- No easy way to keep track of previous conclusions
Practical approach for vulnerability tracking with Buildroot

1. Generate vulnerability info in CI
2. Before release: evaluate vulnerabilities
   - Copy list to separate document
   - Evaluate if applicable + severity
   - Too high severity: patch + back to step 1
3. After release: regularly re-generate vulnerability info to discover new vulnerabilities
   - Based on released source code
   - New vulnerabilities that are N/A are not excluded
   - Manually maintain vulnerability tracking document
Tracking vulnerabilities with Yocto

INHERIT += "cve-check"
include cve-extra-exclusions.inc

- Download NVD CVE database as sqlite database
- For each recipe, look up *everything* matching CVE_PRODUCT
- Mark as Patched if version doesn’t match or patch file exists
- Mark as Ignored if excluded explicitly
- Write result to JSON and text per package + per image
Example Yocto vulnerabilities output

```json
{
  "package": [
    {
      "name": "libpam",
      "version": "1.5.2",
      ...
      "products": [
        {
          "product": "linux-pam",
          "cvesInRecord": "Yes"
        }
      ],
      "issue": [
        {
          "id": "CVE-2009-0579",
          "summary": "Linux-PAM before 1.0.4 does not enforce the minimum password age",
          "scorev2": "4.6",
          "scorev3": "0.0",
          "vector": "LOCAL",
          "status": "Patched",
          "link": "https://nvd.nist.gov/vuln/detail/CVE-2009-0579"
        },
        {
          "id": "CVE-2022-28321",
          "summary": "The Linux-PAM package before 1.5.2-6.1 for openSUSE Tumbleweed NOTE: the relevance of this issue is largely limited to openSUSE Tumbleweed and openSUSE Factory; it does not affect Linux-PAM upstream."
        }
      ]
    }
  ]
}
```
Yocto vulnerabilities features

Only packages in that specific image (no -native)

Match only on product (unless vendor is given)

Extra info for evaluation

Link to NIST database

Status based on version range
Yocto vulnerabilities limitations

- Vulnerability info is only generated as part of build
- Need all layers to generate vulnerability list
- Exclusions are in Yocto (or custom) source
- Need some additional tooling to process JSON files
  - Because patched/ignored are included, contains 1000s of vulns
Tracking vulnerabilities with SPDX SBoM

SBoM (Software Bill of Materials) contains all packages + their versions
⇒ Perfect to as a source for vulnerability information


Using spdx-to-osv
or osv-scanner
OSV (Open Source Vulnerabilities)  
Alternative to CVE database

- Simplify creation of vulnerability entries
- Accurately track upstreams and versions
  - Link to upstream repository
  - Commit hashes in addition to version numbers
- Package identification through ecosystems
  - PyPI, npm, crates.io, ...
  - Alpine, AlmaLinux, Debian, ...
  - OSS-Fuzz
- Unambiguously determine if your software is vulnerable
- Tooling
  - Using SPDX and CycloneDX SBoM
  - Using dependencies in source (Cargo, Go, Python, ...)
  - REST API to query database
Existing OSV tools don’t work

- Buildroot doesn’t generate SPDX SBoM
- Yocto’s SPDX is not compatible with OSV
  - SPDX doesn’t fully specify how to uniquely identify a package
  - Yocto uses name, version, and CPE externalRef

```json
"name": "acl",
"versionInfo": "2.3.1",
"downloadLocation": "https://download.savannah.gnu.org/releases/acl/acl-2.3.1.tar.gz",
"externalRefs": [{
  "referenceCategory": "SECURITY",
  "referenceLocator": "cpe:2.3:a:*:acl:2.3.1:*:*:*:*:*:*",
  "referenceType": "http://spdx.org/rdf/references/cpe23Type"
}]

"homepage": "http://savannah.nongnu.org/projects/acl/"
```

- osv-scanner expects package identified with [purl](https://purl.org/net)
- spdx-to-osv isn’t able to parse cross-document relationships
Other (theoretical) problems with OSV

- Ecosystem must actively register vulnerabilities
  - 31K CVEs tracked on security-tracker.debian.org
  - 9K OSVs tracked in Debian ecosystem
  - Distros only register vulnerabilities that apply to them
  - Many CVEs never registered anywhere in OSV

- Same vulnerability registered in different ecosystems
  - CVE-2019-6706 in Alpine ecosystem
  - RLSA-2019:3706 in Rocky Linux ecosystem
  - DLA-3469-1 in Debian ecosystem includes several CVEs

- Ecosystem has their own package identification scheme
  - E.g. libcurl vs curl
Conclusions

- Buildroot and yocto have tooling for CVE tracking using CPE ID
- Focused on tracking in Buildroot/yocto itself not on tracking by the user
- OSV and SPDX show promise for improved tracking but tooling is not quite there yet