OWASP Dependency Check

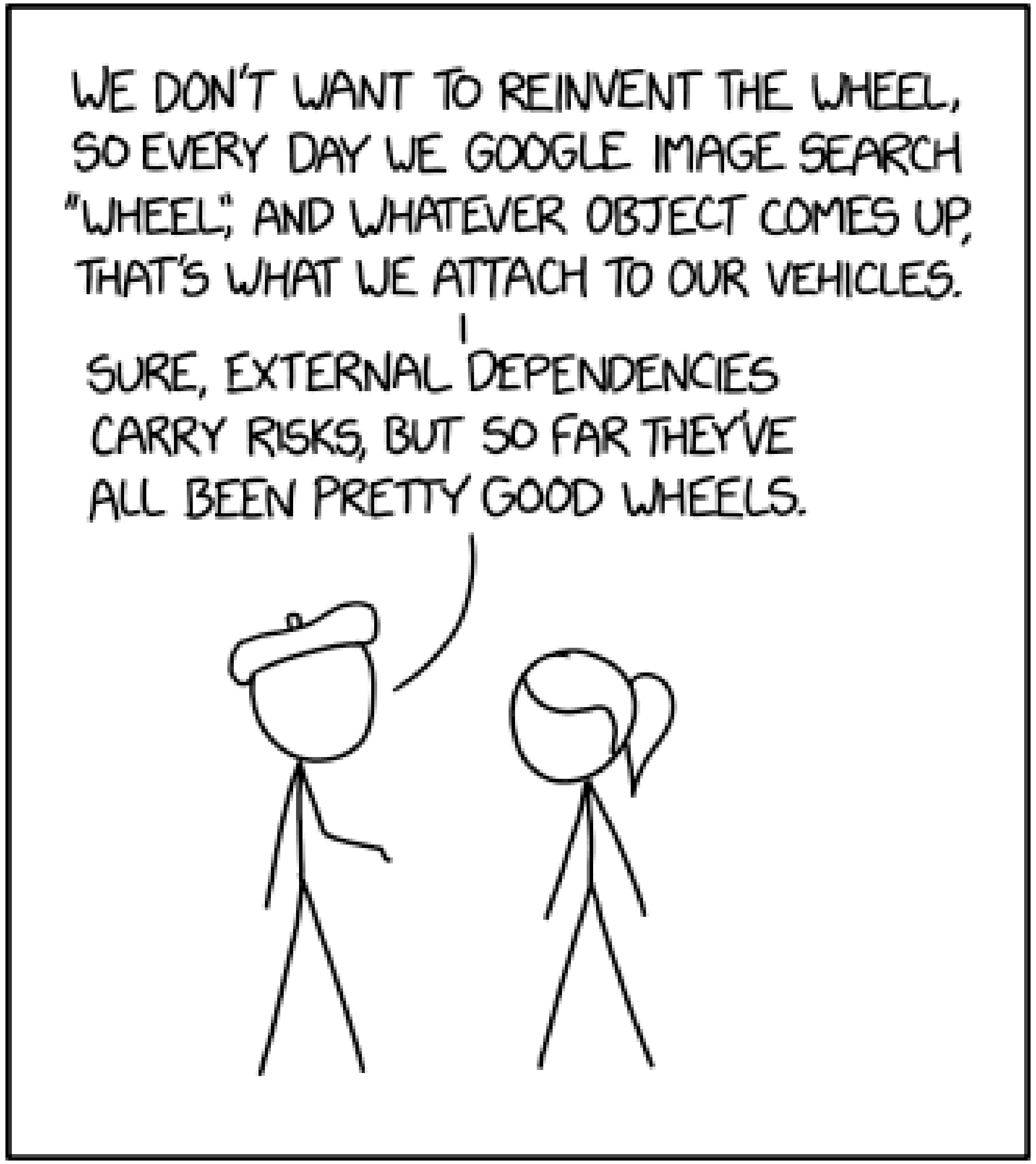
A Primer

The Problem

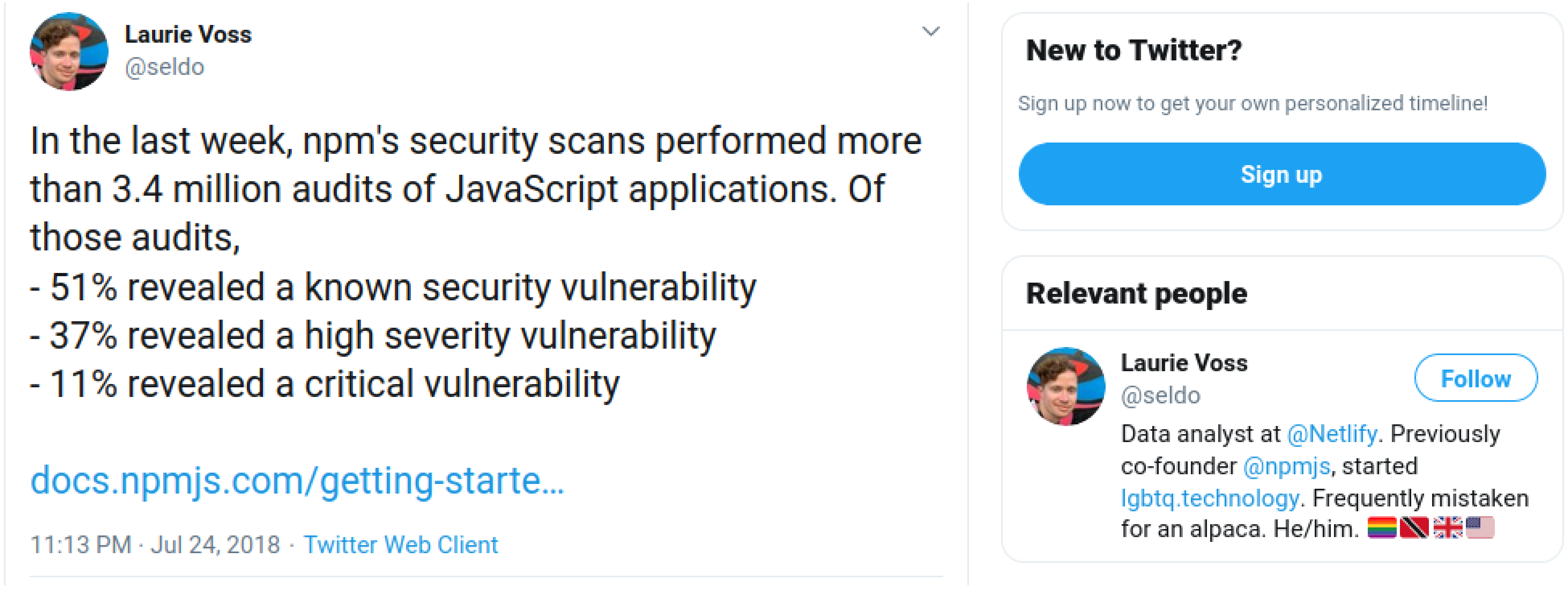
In A Nutshell

1. Modern applications depend on open source,
2. they contain many 3rd party components **and their vulnerabilities**.

"Reinvent The Wheel" by xkcd: https://xkcd.com/2140/


WE DON'T WANT TO REINVENT THE WHEEL, SO EVERY DAY WE GOOGLE IMAGE SEARCH "WHEEL," AND WHATEVER OBJECT COMES UP, THAT'S WHAT WE ATTACH TO OUR VEHICLES.
SURE, EXTERNAL DEPENDENCIES CARRY RISKS, BUT SO FAR THEY'VE ALL BEEN PRETTY GOOD WHEELS.

The Problem


Laurie Voss
@seldo

In the last week, npm's security scans performed more than 3.4 million audits of JavaScript applications. Of those audits,
- 51% revealed a known security vulnerability
- 37% revealed a high severity vulnerability
- 11% revealed a critical vulnerability

docs.npmjs.com/getting-starte…

11:13 PM · Jul 24, 2018 · Twitter Web Client

New to Twitter?

Sign up now to get your own personalized timeline!

Relevant people

Laurie Voss
@seldo

Data analyst at @Netlify. Previously co-founder @npmjs, started lgbtq.technology. Frequently mistaken for an alpaca. He/him. 🏳️‍🌈🇹🇹🇬🇧🇺🇸

Vulnerabilities on NPM in 2018: https://twitter.com/seldo/status/1021865857813630976

The Problem

This problem has been recognized by the OWASP Top 10 Web Application Security Risks.

OWASP Top 10

> *"**#9 Using Components with Known Vulnerabilities.** Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts."* [owasp.org]

For more Information see:
Using_Components_with_Known_Vulnerabilities

Enter *OWASP Dependency Check*

A tool for mitigating OWASP Top 10 #9.

Checks dependencies for **Known Vulnerabilities**.

Developed by OWASP / Jeremy Long.

Full support for Java and .NET applications.

Experimental support for Python, Ruby, PHP and JavaScript/Node.js applications.

References:

- Project site on owasp.org
- Online documentation on github.io

Azure Pipeline Integration

Hosted Agents

Just add this tasks to your pipeline.

```yaml
- task: dependency-check-build-task@5
  displayName: 'Dependency Check: Run'
  inputs:
    projectName: MyProject       # name of the project
    scanPath: path/to/scanPath   # path of artifacts to scan
    failOnCVSS: 0                # threshold when to fail build
    format: 'HTML'               # output format
    enableExperimental: false    # use experimental analyzers
    enableRetired: false         # use retired analyzers
    enableVerbose: false         # run in verbose mode
```

Azure Pipeline Integration

On-Premise Agents (1/2)

Dependency-Check requires JRE/JDK to run.

```
- task: JavaToolInstaller@0
  displayName: 'Dependency Check: Install OpenJDK'
  inputs:
    versionSpec: "13"
    jdkArchitectureOption: x64
    jdkSourceOption: LocalDirectory
    jdkFile: "path/to/openjdk-13.0.2_windows-x64_bin.zip"
    jdkDestinationDirectory: "DependencyCheck/Binaries/Externals"
    cleanDestinationDirectory: true

- task: dependency-check-build-task@5
  ...
```

Azure Pipeline Integration

On-Premise Agents (2/2)

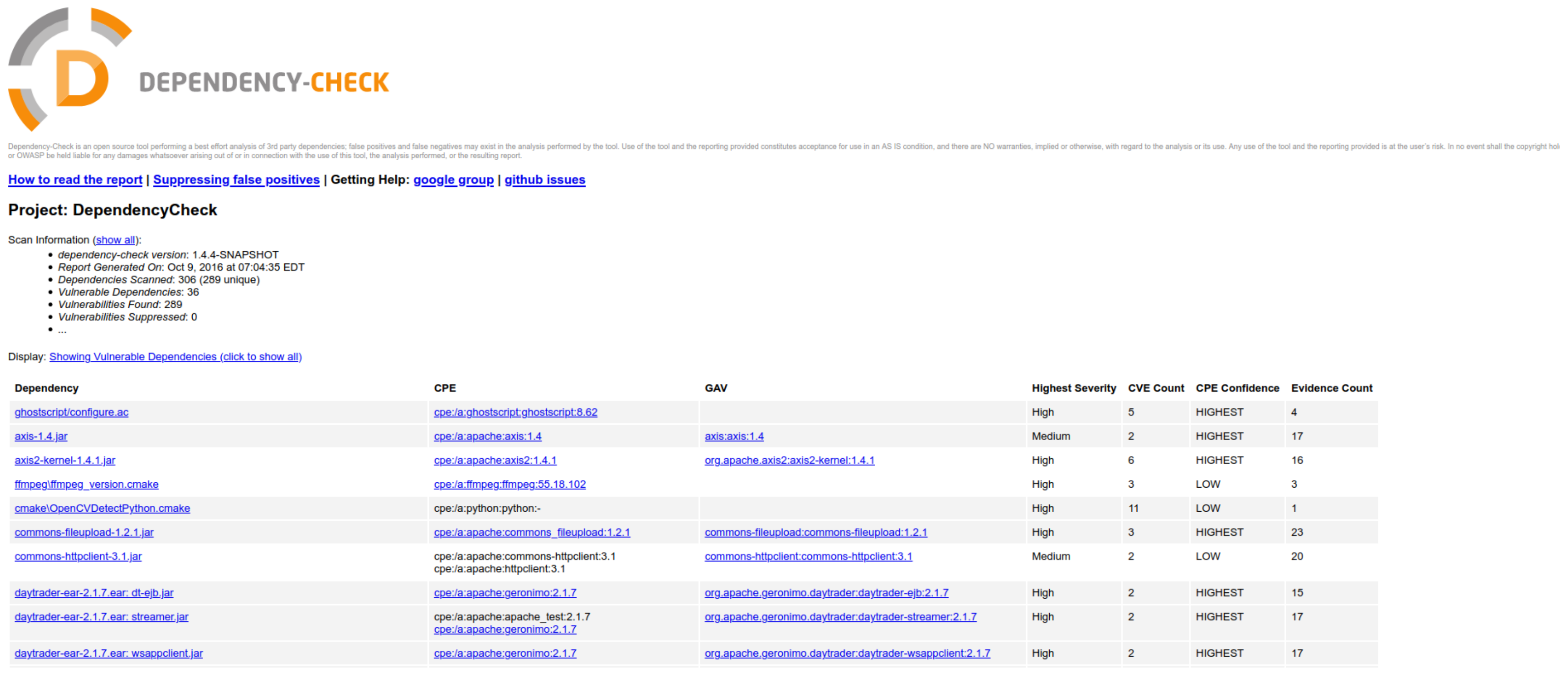
.NET Analyzers require .NET Core.

```
- task: UseDotNet@2
  displayName: 'Dependency Check: Install .NET Core sdk'
  inputs:
    packageType: sdk
    version: 2.x
    installationPath: $(Agent.ToolsDirectory)/dotnet

- task: JavaToolInstaller@0
  ...

- task: dependency-check-build-task@5
  ...
```

Dependency Check Reports



Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright hol
or OWASP be held liable for any damages whatsoever arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

How to read the report | Suppressing false positives | Getting Help: google group | github issues

Project: DependencyCheck

Scan Information (show all):

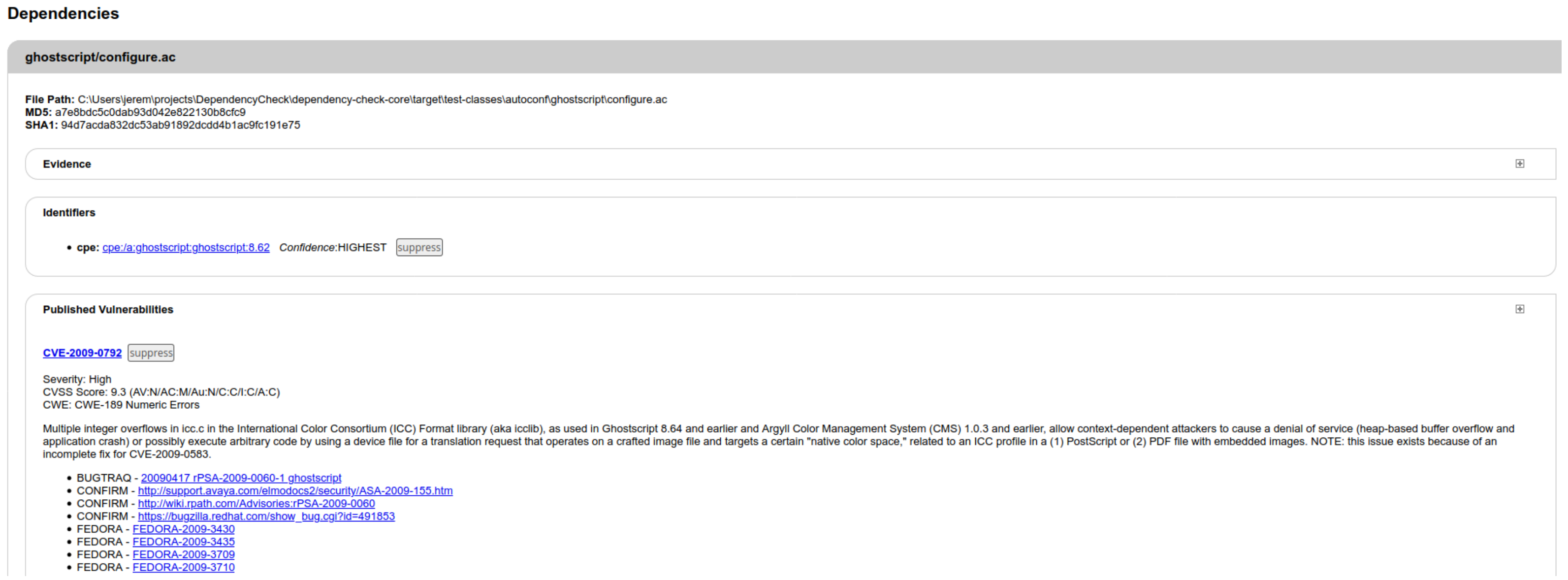
- *dependency-check version*: 1.4.4-SNAPSHOT
- *Report Generated On*: Oct 9, 2016 at 07:04:35 EDT
- *Dependencies Scanned*: 306 (289 unique)
- *Vulnerable Dependencies*: 36
- *Vulnerabilities Found*: 289
- *Vulnerabilities Suppressed*: 0
- ...

Display: Showing Vulnerable Dependencies (click to show all)

Dependency	CPE	GAV	Highest Severity	CVE Count	CPE Confidence	Evidence Count
ghostscript/configure.ac	cpe:/a:ghostscript:ghostscript:8.62		High	5	HIGHEST	4
axis-1.4.jar	cpe:/a:apache:axis:1.4	axis:axis:1.4	Medium	2	HIGHEST	17
axis2-kernel-1.4.1.jar	cpe:/a:apache:axis2:1.4.1	org.apache.axis2:axis2-kernel:1.4.1	High	6	HIGHEST	16
ffmpeg\ffmpeg_version.cmake	cpe:/a:ffmpeg:ffmpeg:55.18.102		High	3	LOW	3
cmake\OpenCVDetectPython.cmake	cpe:/a:python:python:-		High	11	LOW	1
commons-fileupload-1.2.1.jar	cpe:/a:apache:commons_fileupload:1.2.1	commons-fileupload:commons-fileupload:1.2.1	High	3	HIGHEST	23
commons-httpclient-3.1.jar	cpe:/a:apache:commons-httpclient:3.1 cpe:/a:apache:httpclient:3.1	commons-httpclient:commons-httpclient:3.1	Medium	2	LOW	20
daytrader-ear-2.1.7.ear: dt-ejb.jar	cpe:/a:apache:geronimo:2.1.7	org.apache.geronimo.daytrader:daytrader-ejb:2.1.7	High	2	HIGHEST	15
daytrader-ear-2.1.7.ear: streamer.jar	cpe:/a:apache:apache_test:2.1.7 cpe:/a:apache:geronimo:2.1.7	org.apache.geronimo.daytrader:daytrader-streamer:2.1.7	High	2	HIGHEST	17
daytrader-ear-2.1.7.ear: wsappclient.jar	cpe:/a:apache:geronimo:2.1.7	org.apache.geronimo.daytrader:daytrader-wsappclient:2.1.7	High	2	HIGHEST	17

Report header with a list of found vulnerabilities.

Dependency Check Reports



Dependencies

ghostscript/configure.ac

File Path: C:\Users\jerem\projects\DependencyCheck\dependency-check-core\target\test-classes\autoconf\ghostscript\configure.ac
MD5: a7e8bdc5c0dab93d042e822130b8cfc9
SHA1: 94d7acda832dc53ab91892dcdd4b1ac9fc191e75

Evidence

Identifiers

- **cpe:** cpe:/a:ghostscript:ghostscript:8.62 *Confidence:*HIGHEST suppress

Published Vulnerabilities

CVE-2009-0792 suppress

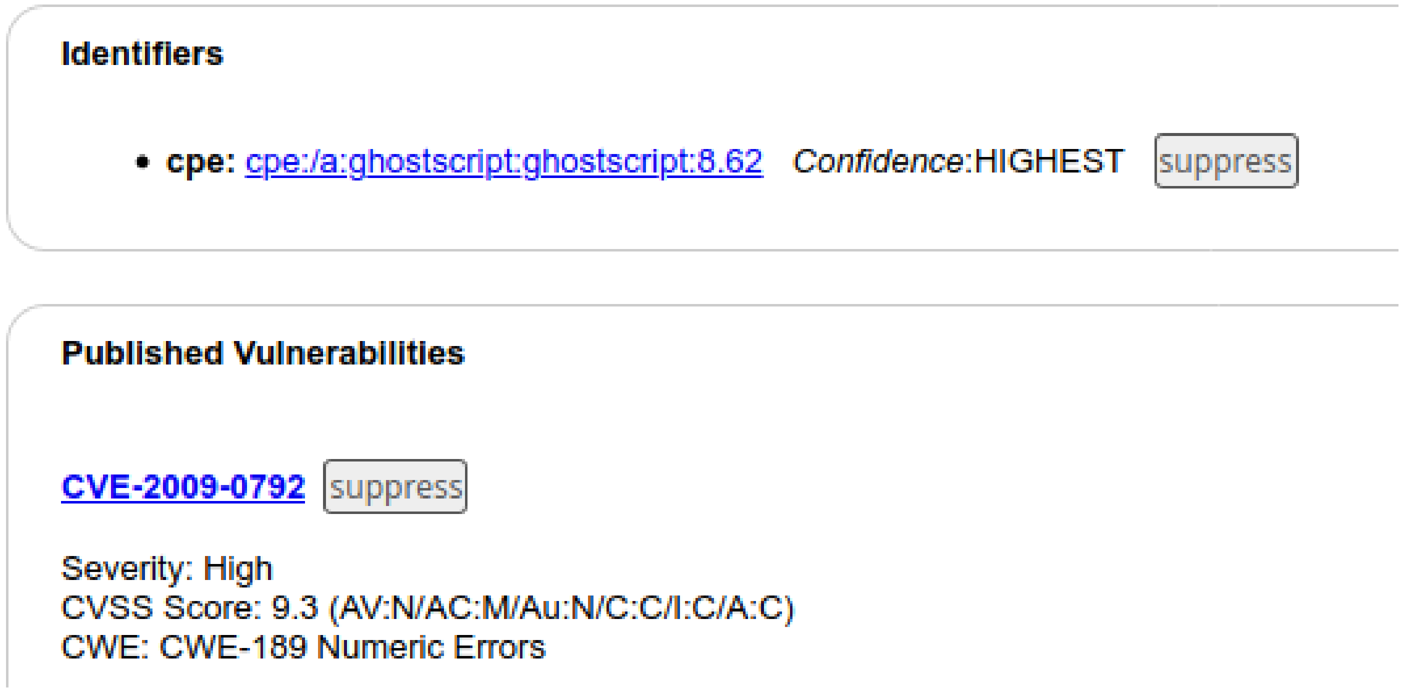
Severity: High
CVSS Score: 9.3 (AV:N/AC:M/Au:N/C:C/I:C/A:C)
CWE: CWE-189 Numeric Errors

Multiple integer overflows in icc.c in the International Color Consortium (ICC) Format library (aka icclib), as used in Ghostscript 8.64 and earlier and Argyll Color Management System (CMS) 1.0.3 and earlier, allow context-dependent attackers to cause a denial of service (heap-based buffer overflow and application crash) or possibly execute arbitrary code by using a device file for a translation request that operates on a crafted image file and targets a certain "native color space," related to an ICC profile in a (1) PostScript or (2) PDF file with embedded images. NOTE: this issue exists because of an incomplete fix for CVE-2009-0583.

- BUGTRAQ - 20090417 rPSA-2009-0060-1 ghostscript
- CONFIRM - http://support.avaya.com/elmodocs2/security/ASA-2009-155.htm
- CONFIRM - http://wiki.rpath.com/Advisories:rPSA-2009-0060
- CONFIRM - https://bugzilla.redhat.com/show_bug.cgi?id=491853
- FEDORA - FEDORA-2009-3430
- FEDORA - FEDORA-2009-3435
- FEDORA - FEDORA-2009-3709
- FEDORA - FEDORA-2009-3710

Where do the links go?

Dependency Check Reports



Identifiers

- **cpe:** cpe:/a:ghostscript:ghostscript:8.62 *Confidence*:HIGHEST

Published Vulnerabilities

CVE-2009-0792

Severity: High
CVSS Score: 9.3 (AV:N/AC:M/Au:N/C:C/I:C/A:C)
CWE: CWE-189 Numeric Errors

What is CVE, CWE, CVSS, CPE?

National Vulnerability Database

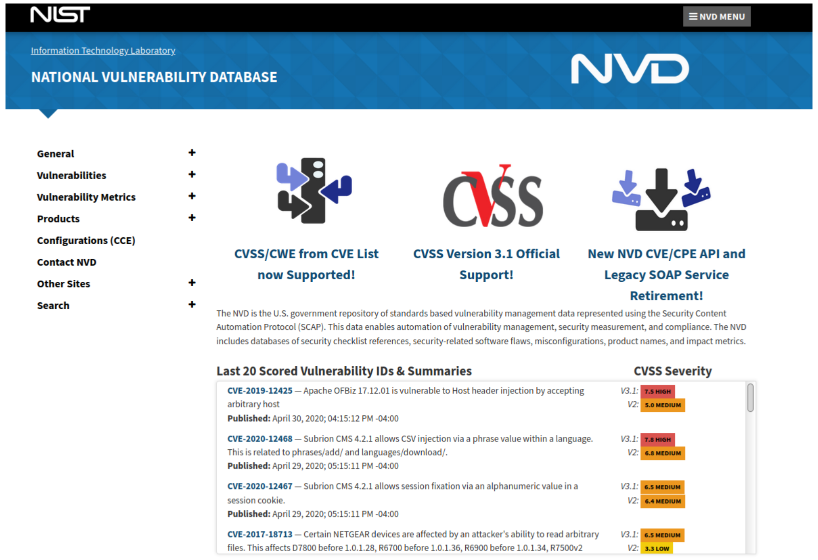
The main source for understanding reports of Dependency Check.

The US Government repository for *Security Content Automation Protocol* (SCAP) content.

SCAP Components

- Common Vulnerabilities and Exposures (CVE)
- Common Vulnerability Scoring System (CVSS)
- Common Platform Enumeration (CPE)
- and more.

NOTE: *(1.) SCAP contains other components which are not important for now. (2.) CVE content includes Common Weakness Enumeration (CWE) content which is not part of SCAP, AFAIK.*


NIST
NVD MENU
Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
NVD
General
Vulnerabilities
Vulnerability Metrics
Products
Configurations (CCE)
Contact NVD
Other Sites
Search
CVSS/CWE from CVE List now Supported!
CVSS Version 3.1 Official Support!
New NVD CVE/CPE API and Legacy SOAP Service Retirement!

The NVD is the U.S. government repository of standards based vulnerability management data represented using the Security Content Automation Protocol (SCAP). This data enables automation of vulnerability management, security measurement, and compliance. The NVD includes databases of security checklist references, security-related software flaws, misconfigurations, product names, and impact metrics.

Last 20 Scored Vulnerability IDs & Summaries	CVSS Severity
CVE-2019-12425 — Apache OFBiz 17.12.01 is vulnerable to Host header injection by accepting arbitrary host **Published:** April 30, 2020; 04:15:12 PM -04:00	*V3.1:* 7.5 HIGH *V2:* 5.0 MEDIUM
CVE-2020-12468 — Subrion CMS 4.2.1 allows CSV injection via a phrase value within a language. This is related to phrases/add/ and languages/download/. **Published:** April 29, 2020; 05:15:11 PM -04:00	*V3.1:* 7.8 HIGH *V2:* 6.8 MEDIUM
CVE-2020-12467 — Subrion CMS 4.2.1 allows session fixation via an alphanumeric value in a session cookie. **Published:** April 29, 2020; 05:15:11 PM -04:00	*V3.1:* 6.5 MEDIUM *V2:* 6.4 MEDIUM
CVE-2017-18713 — Certain NETGEAR devices are affected by an attacker's ability to read arbitrary files. This affects D7800 before 1.0.1.28, R6700 before 1.0.1.36, R6900 before 1.0.1.34, R7500v2	*V3.1:* 6.5 MEDIUM *V2:* 3.3 LOW

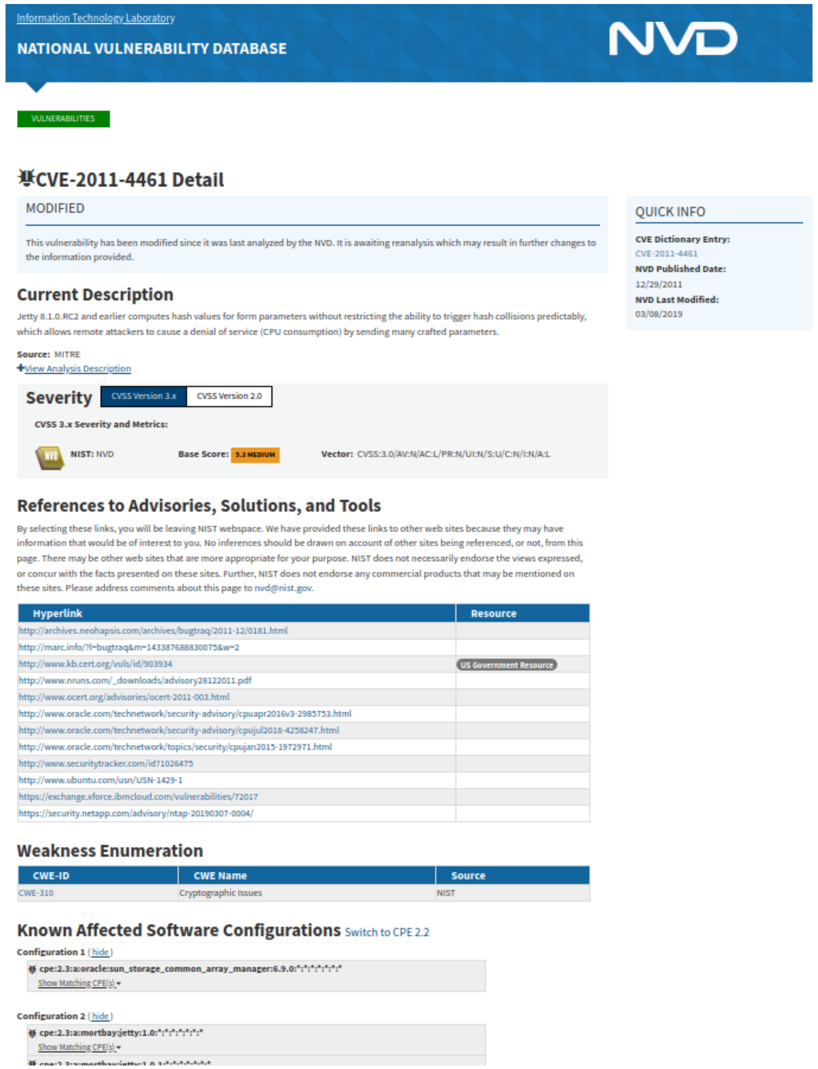
National Vulnerability Database

NVD Entry Sections

- **Title**
  → *Common Vulnerabilities and Exposures (CVE)*
- Quick Info
- Current Description
- **Serverity**
  → *Common Vulnerability Scoring System (CVSS)*
- References to Advisories, Solutions, and Tools
- **Weakness Enumeration**
  → *Common Weakness Enumeration (CWE)*
- **Known Affected Software Configurations**
  → *Common Platform Enumeration (CPE)*
- Change History

Example

https://nvd.nist.gov/vuln/detail/CVE-2011-4461


Information Technology Laboratory
NATIONAL VULNERABILITY DATABASE
NVD

VULNERABILITIES

CVE-2011-4461 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

QUICK INFO
CVE Dictionary Entry:
CVE-2011-4461
NVD Published Date:
12/29/2011
NVD Last Modified:
03/08/2019

Current Description

Jetty 8.1.0.RC2 and earlier computes hash values for form parameters without restricting the ability to trigger hash collisions predictably, which allows remote attackers to cause a denial of service (CPU consumption) by sending many crafted parameters.

Source: MITRE
+View Analysis Description

Severity CVSS Version 3.x | CVSS Version 2.0

CVSS 3.x Severity and Metrics:

NIST: NVD **Base Score:** 5.3 MEDIUM **Vector:** CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L

References to Advisories, Solutions, and Tools

By selecting these links, you will be leaving NIST webspace. We have provided these links to other web sites because they may have information that would be of interest to you. No inferences should be drawn on account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for your purpose. NIST does not necessarily endorse the views expressed, or concur with the facts presented on these sites. Further, NIST does not endorse any commercial products that may be mentioned on these sites. Please address comments about this page to nvd@nist.gov.

Hyperlink	Resource
http://archives.neohapsis.com/archives/bugtraq/2011-12/0181.html	
http://marc.info/?l=bugtraq&m=143387688830075&w=2	
http://www.kb.cert.org/vuls/id/903934	US Government Resource
http://www.nruns.com/_downloads/advisory28122011.pdf	
http://www.ocert.org/advisories/ocert-2011-003.html	
http://www.oracle.com/technetwork/security-advisory/cpuapr2016v3-2985753.html	
http://www.oracle.com/technetwork/security-advisory/cpujul2018-4258247.html	
http://www.oracle.com/technetwork/topics/security/cpujan2015-1972971.html	
http://www.securitytracker.com/id?1026475	
http://www.ubuntu.com/usn/USN-1429-1	
https://exchange.xforce.ibmcloud.com/vulnerabilities/72017	
https://security.netapp.com/advisory/ntap-20190307-0004/	

Weakness Enumeration

CWE-ID	CWE Name	Source
CWE-310	Cryptographic Issues	NIST

Known Affected Software Configurations Switch to CPE 2.2

Configuration 1 (hide)
cpe:2.3:a:oracle:sun_storage_common_array_manager:6.9.0:*:*:*:*:*:*:*
Show Matching CPE(s)

Configuration 2 (hide)
cpe:2.3:a:mortbay:jetty:1.0:*:*:*:*:*:*:*
Show Matching CPE(s)

Common Vulnerabilities and Exposures

Project Site: https://cve.mitre.org/

A system to identify publicly known vulnerabilities and exposures.

- **CVE Number**
  identifies publicly known vulnerabilities and exposures
- **CVE Numbering Authority (CNA)**
  assigns CVE Numbers
  e.g. The MITRE Corporation, Microsoft, Red Hat and others
- **CVE Number Syntax**
  `CVE prefix + Year + Arbitrary Digits`
- **CVE Number Example**
  CVE-2020-11022 (a jQuery XSS vulnerability)

Common Vulnerabilities and Exposures

CVE Numbers are only assigned to flaws which satisfy the following criteria.

A flaw must be:

1. **Independently Fixable**
   The flaw can be fixed independently of any other bugs.
2. **Acknowledged by the affected vendor** or **Documented**
   The flaw is either confirmed by the vendor or has a recorded prove.
3. **Affecting one codebase**
   The flaw may impact many products, e.g white-labeling, but resides in a single codebase.

For further information see:
https://www.redhat.com/en/topics/security/what-is-cve

Common Weakness Enumeration

Project Site: https://cwe.mitre.org/

A category system for software weaknesses and vulnerabilities.

The CWE system is a community project which aims to understand, identify, fix and prevent common security flaws in software and to create automated tools helping with these objectives.

- **CWE Number**
  identifies a category of known weaknesses or a concrete known weakness in software
- **CWE Number Syntax**
  `CWE prefix + Arbitrary Digits`
- **CWE Number Examples**
  - *CWE Category:* CWE-1211 Authentication Errors
  - *CWE Weakness:* CWE-295 Imporper Certificate Validation

The MITRE Corporation

The CVE and CWE systems are maintained and sponsored by The MITRE Corporation:

- Project Site: https://www.mitre.org/
- Non-Profit Organization
- Primary CNA
- Funded by various US Government institutions:
  - Dpt. of Homeland Security
  - Dpt. of Defense
  - Federal Aviation Administration
  - Internal Revenue Service
  - Department of Veterans Affairs.
  - National Institute of Standards and Technology
  - Administrative Office of the United States Courts
  - Centers for Medicare and Medicaid Services

FYI: "MITRE" has no meaning, although it originated around the Massachusetts Institute of Technology (MIT)

Common Vulnerability Scoring System

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

A system for calculating the serverity of vulnerabilities.

- **CVSS Vector**
  describes *exploitability* and *impact* of a vulnerability
  - **Example**
    `CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:L/A:N`
- **CVSS Score**
  describes the approximate serverity of a vulnerability
  floating point value between 0 (good) and 10 (bad)
  - **Example**
    `Base Score 6.9` (medium serverity)

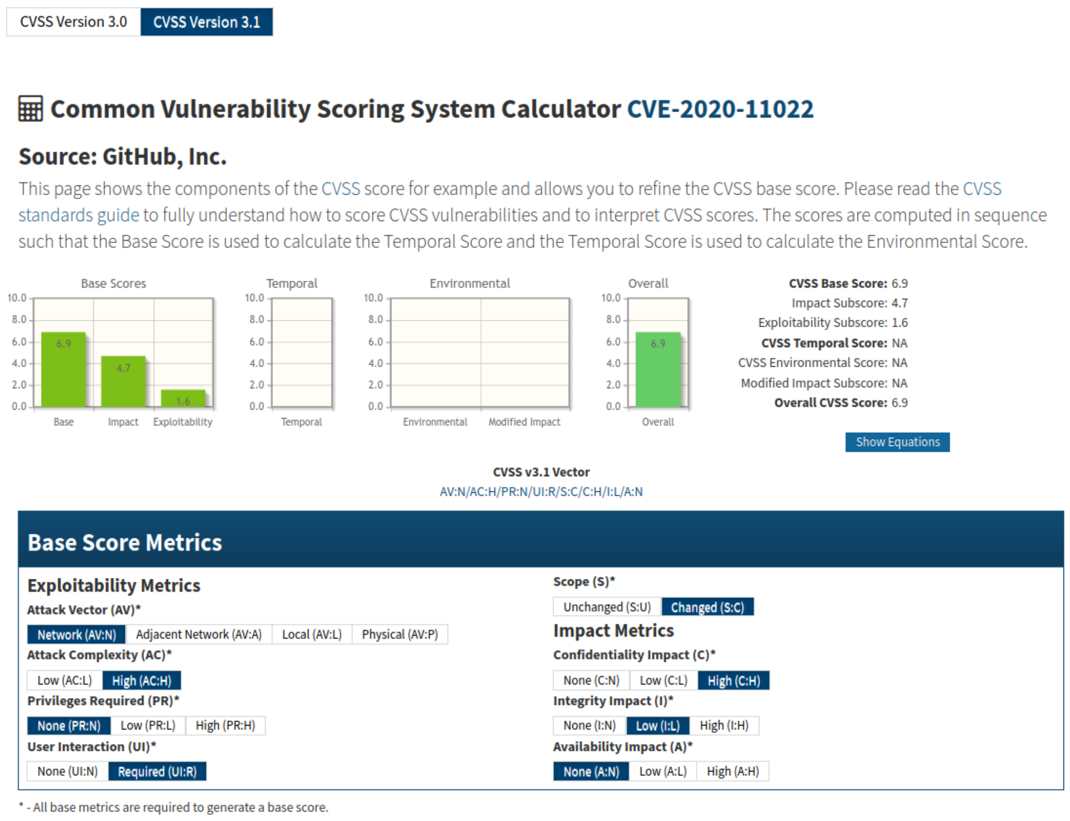
Common Vulnerability Scoring System

CVSS Vector: `CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:L/A:N`

CVSS Version: 3.1

Metric	Value	Category
Attack Vector (AV)	Network (N), Adjacent Network (A), Local (L), Physical (P)	Exploitability
Access Complexity (AC)	Low (L), High (H)	Exploitability
Privileges Required (PR)	None (N), Low (L), High (H)	Exploitability
User Interaction (UI)	None (N), Required (R)	Exploitability
(Authorization) Scope (S)	Unchanged (U), Changed (C)	Exploitability
Confidentiality Impact (C)	None (N), Low (L), High (H)	Impact
Integrity Impact (I)	None (N), Low (L), High (H)	Impact
Availability Impact (A)	None (N), Low (L), High (H)	Impact

Common Vulnerability Scoring System



CVSS Version 3.0 [✓ CVSS Version 3.1]

Common Vulnerability Scoring System Calculator CVE-2020-11022

Source: GitHub, Inc.

This page shows the components of the CVSS score for example and allows you to refine the CVSS base score. Please read the CVSS standards guide to fully understand how to score CVSS vulnerabilities and to interpret CVSS scores. The scores are computed in sequence such that the Base Score is used to calculate the Temporal Score and the Temporal Score is used to calculate the Environmental Score.

CVSS Base Score:	6.9
Impact Subscore:	4.7
Exploitability Subscore:	1.6
CVSS Temporal Score:	NA
CVSS Environmental Score:	NA
Modified Impact Subscore:	NA
Overall CVSS Score:	6.9

CVSS v3.1 Vector
AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:L/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*
[✓ Network (AV:N)] Adjacent Network (AV:A) Local (AV:L) Physical (AV:P)

Attack Complexity (AC)*
Low (AC:L) [✓ High (AC:H)]

Privileges Required (PR)*
[✓ None (PR:N)] Low (PR:L) High (PR:H)

User Interaction (UI)*
None (UI:N) [✓ Required (UI:R)]

Scope (S)*
Unchanged (S:U) [✓ Changed (S:C)]

Impact Metrics

Confidentiality Impact (C)*
None (C:N) Low (C:L) [✓ High (C:H)]

Integrity Impact (I)*
None (I:N) [✓ Low (I:L)] High (I:H)

Availability Impact (A)*
[✓ None (A:N)] Low (A:L) High (A:H)

* - All base metrics are required to generate a base score.

This example of a CVSS calculation for CVE-2020-11022 can be found on NVD.

Common Platform Enumeration

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

A naming system to uniquely identify information technology systems (hardware and software).

Originally developed be MITRE (https://cpe.mitre.org/), now part of the *Security Content Automation Protocol* (SCAP) maintained by the National Institute of Standards and Technology (NIST).

- **CPE Well-Formed Name (WFN)**
  uniquely identifies an information technology system (hardware or software)
- **CPE URI**
  represents a CPE WFN

NOTE: *CPE WFNs are not really important for working with Dependency Check.*

Common Platform Enumeration

- **CPE URI Syntax**
  - *CPE 2.2:*
    `cpe:/{part}:{vendor}:{product}:{version}:{update}:{edition}:{language}`
  - *CPE 2.3:*
    `cpe:2.3:{part}:{vendor}:{product}:{version}:{update}:{edition}:{language}:{sw_edition}:{target_sw}:{target_hw}:{other}`
- **CPE URI Examples**
  - *CPE 2.2:*
    `cpe:/a:jquery:jquery:1.0.1`
  - *CPE 2.3:*
    `cpe:2.3:a:jquery:jquery:1.0.1:*:*:*:*:*:*:*`

THE CPE Naming Specification Version 2.3 can be found here.

Common Platform Enumeration

Important CPE Sections

CPE 2.2:

```
cpe:/{part}:{vendor}:{product}:{version}
cpe:/a:jquery:jquery:1.0.1
```

CPE 2.3:

```
cpe:2.3:{part}:{vendor}:{product}:{version}
cpe:2.3:a:jquery:jquery:1.0.1:*:*:*:*:*:*:*
```

URI Section	Value	Description
part	a	applications (a), operating systems (o), hardware (h)
vendor	jquery	name of the vendor
product	jquery	name of the product
version	1.0.1	version of the product

How It Works

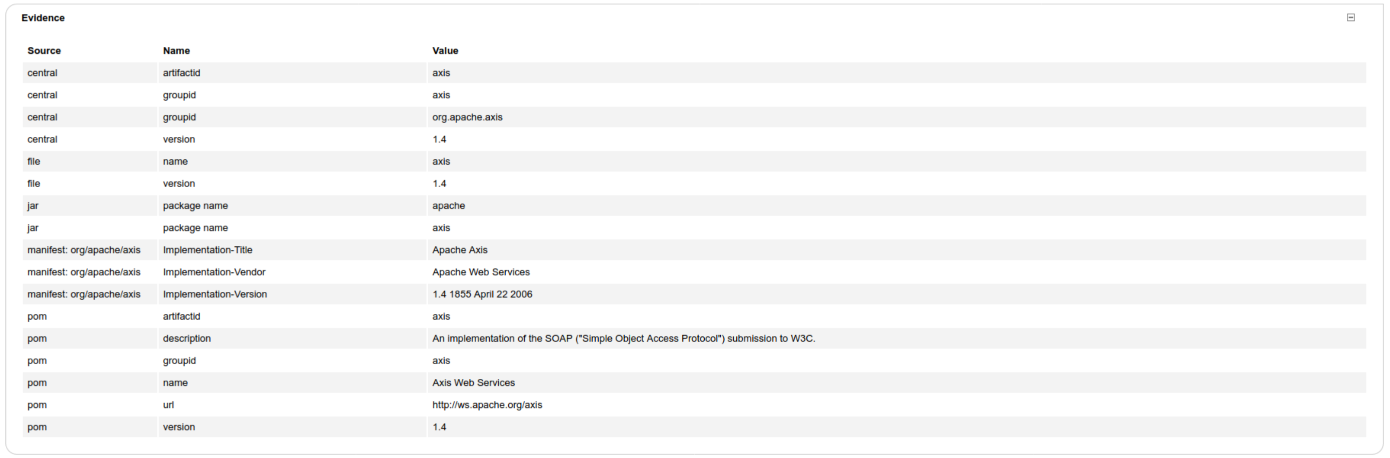
1. **Apply Analyzers**
   Dependency Check applies multiple *Analyzers* to all dependencies, i.e. artifacts under the scan path.
2. **Extract Evidence**
   Analyzers extract *Evidence*, e.g. file name, manifest, POM, package names, etc.
3. **Determine CPE**
   Evidence is grouped into *Vendor*, *Product* and *Version* and determine CPE.
4. **Match CPE**
   Match CPE against the CVE database.

Important: The determined CPE has a confidence level equal to the lowest confidence level of used evidence to create it.

More information can be found here.

DEPENDENCY-CHECK

How It Works



Evidence

Source	Name	Value
central	artifactid	axis
central	groupid	axis
central	groupid	org.apache.axis
central	version	1.4
file	name	axis
file	version	1.4
jar	package name	apache
jar	package name	axis
manifest: org/apache/axis	Implementation-Title	Apache Axis
manifest: org/apache/axis	Implementation-Vendor	Apache Web Services
manifest: org/apache/axis	Implementation-Version	1.4 1855 April 22 2006
pom	artifactid	axis
pom	description	An implementation of the SOAP ("Simple Object Access Protocol") submission to W3C.
pom	groupid	axis
pom	name	Axis Web Services
pom	url	http://ws.apache.org/axis
pom	version	1.4

Due to how the matching process works Dependency Chack may produce:

- **False Positives**
  A matching CVE has nothing to do with your project.
- **False Negatives**
  A matching CVE is not found.

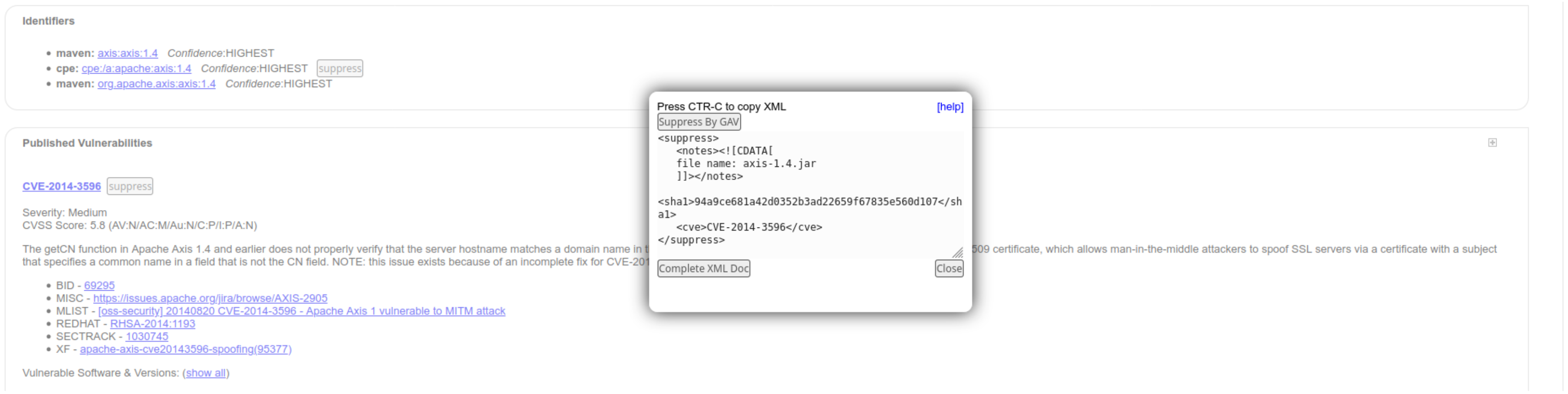
Dealing with False Positives (1/2)

```
- task: dependency-check-build-task@5
  displayName: 'Dependency Check: Run'
  inputs:
    ...
    suppressionPath: 'path/to/DependencyCheck/Supressions.xml' # add a supression file
    ...
```

```
<?xml version="1.0" encoding="UTF-8"?>
<suppressions xmlns="https://jeremylong.github.io/DependencyCheck/dependency-suppression.1.3.xsd">
    <suppress until="2020-01-01Z">
      <notes><![CDATA[
      file name: some.jar
      ]]></notes>
      <sha1>66734244CE86857018B023A8C56AE0635C56B6A1</sha1>
      <cpe>cpe:/a:apache:struts:2.0.0</cpe>
    </suppress>
</suppressions>
```

Add a suppression file to conditionally ignore False Positives.

Dealing with False Positives (2/2)


Identifiers

- **maven:** axis:axis:1.4 *Confidence*:HIGHEST
- **cpe:** cpe:/a:apache:axis:1.4 *Confidence*:HIGHEST suppress
- **maven:** org.apache.axis:axis:1.4 *Confidence*:HIGHEST

Published Vulnerabilities

CVE-2014-3596 suppress

Severity: Medium
CVSS Score: 5.8 (AV:N/AC:M/Au:N/C:P/I:P/A:N)

The getCN function in Apache Axis 1.4 and earlier does not properly verify that the server hostname matches a domain name in t 509 certificate, which allows man-in-the-middle attackers to spoof SSL servers via a certificate with a subject that specifies a common name in a field that is not the CN field. NOTE: this issue exists because of an incomplete fix for CVE-201

- BID - 69295
- MISC - https://issues.apache.org/jira/browse/AXIS-2905
- MLIST - [oss-security] 20140820 CVE-2014-3596 - Apache Axis 1 vulnerable to MITM attack
- REDHAT - RHSA-2014:1193
- SECTRACK - 1030745
- XF - apache-axis-cve20143596-spoofing(95377)

Vulnerable Software & Versions: (show all)

Press CTR-C to copy XML

```
<suppress>
   <notes><![CDATA[
   file name: axis-1.4.jar
   ]]></notes>

<sha1>94a9ce681a42d0352b3ad22659f67835e560d107</sh
a1>
   <cve>CVE-2014-3596</cve>
</suppress>
```

Suppressions can be generated from HTML reports.

Suppressions can be configured for SHA1 Hashes, CVE Numbers, CPE URIs and more.

Suppressions can also be configured with an expiration date.

Dealing with False Negatives

```yaml
- task: dependency-check-build-task@5
  displayName: 'Dependency Check: Run'
  inputs:
    ...
    additionalArguments: '--hints "$(Build.SourcesDirectory)/path/to/DependencyCheck/Hints.xml"' # add a hints file
    ...
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<hints xmlns="https://jeremylong.github.io/DependencyCheck/dependency-hint.1.1.xsd">
    <hint>
        <given>
            <fileName contains="my-thelib-.*\.jar" regex="true" caseSensitive="true"/>
        </given>
        <add>
            <evidence type="product" source="hint analyzer" name="product" value="thelib" confidence="HIGH"/>
        </add>
  </hint>
</hints>
```

Add a hints file to conditionally enrich extracted evidence to reduce False Negatives.

Questions?

Thanks!