MANICODE

SECURE CODING EDUCATION

Third-Party Library Security Management

A little background dirt…

jim@manicode.com

@manicode

- Former OWASP Global Board Member
- Project manager of the OWASP Cheat Sheet Series and several other OWASP projects
- 25+ years of software development experience
- Author of "Iron-Clad Java, Building Secure Web Applications” from McGraw-Hill/Oracle-Press
- Kauai, Hawaii Resident and Cobb, California Resident

ORACLE
Iron-Clad Java: Building Secure Web Applications
Best Practices for Secure Java Web Application Development
Jim Manico
August Detlefsen
Oracle Press

WARNING: Please do not attempt to hack any computer system without legal permission to do so. Unauthorized computer hacking is illegal and can be punishable by a range of penalties including loss of job, monetary fines and possible imprisonment.

ALSO: The *Free and Open Source Software* presented in these materials are examples of good secure development techniques. You may have unknown legal, licensing or technical issues when making use of *Free and Open Source Software*. You should consult your company's policy on the use of *Free and Open Source Software* before making use of any software referenced in this material.

Third-Party Library Learning Objectives

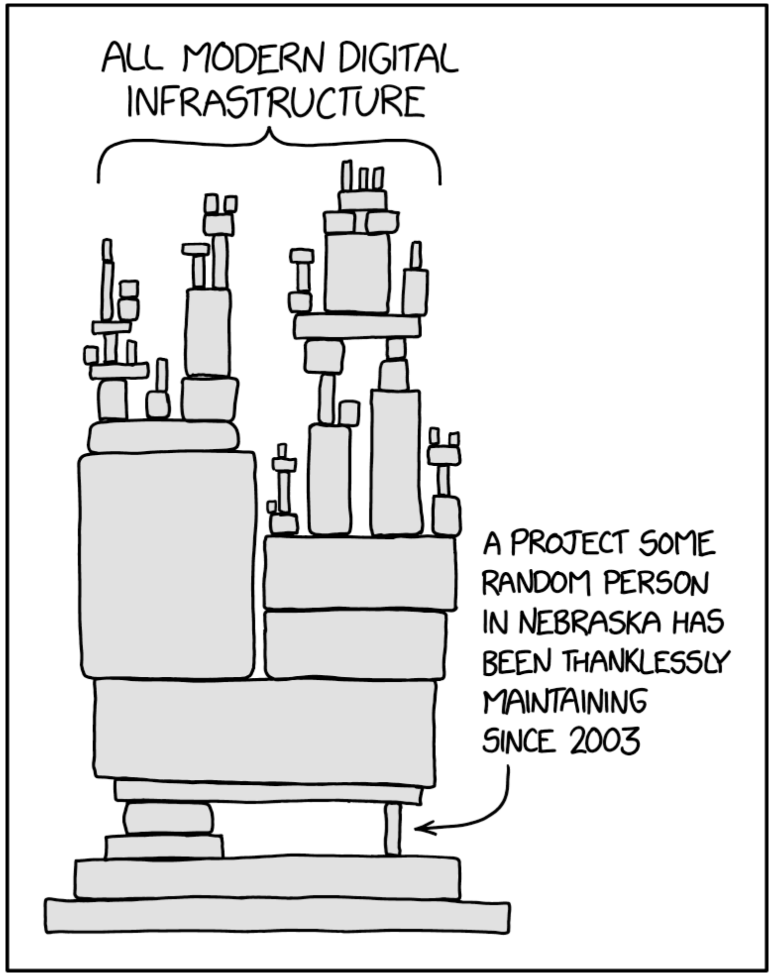
Learn the security significance of the use of third-party libraries

Learn how to vet third-party libraries before using them

Learn how to conduct security verification of your third-party libraries

Learn process tradeoffs regarding the management of third-party libraries

Third-Party Library Security
Introduction


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

https://xkcd.com/2347/

While this slide deck is commercial in nature XKCD's comic is creative commons

A06:2022-Vulnerable and Outdated Components was previously titled Using Components with Known Vulnerabilities and is #2 in the Top 10 community survey, but also had enough data to make the Top 10 via data analysis. This category moves up from #9 in 2017 and is a known issue that we struggle to test and assess risk.

https://owasp.org/www-project-top-ten/

KEY CONCEPTS A06:2021 – VULNERABLE AND OUTDATED COMPONENTS

Software often has

- Past vulnerable versions using outdated components

Systems and frameworks have a date by when

- End of life, end of sale, unsupported

Attackers attempt to exploit newly published vulnerabilities

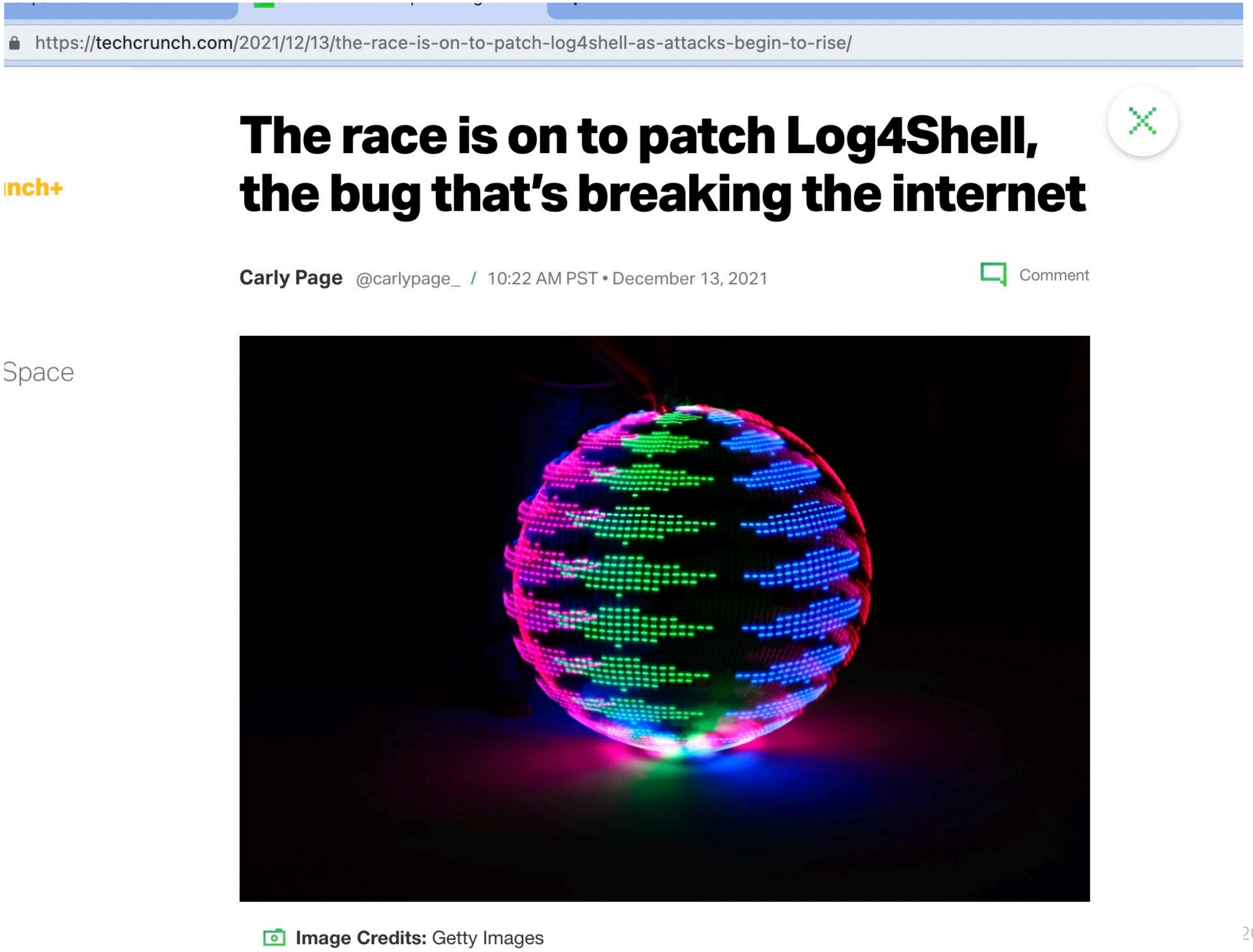
- Requiring you apply fixes or perform updates

DEFINITION A06:2021 – VULNERABLE AND OUTDATED COMPONENTS

- Having in use software components that are vulnerable, unsupported, or out of date
- Environments that are not patched in a timely manner
- Not knowing the versions of software, you use

CHALLENGES A06:2021 – VULNERABLE AND OUTDATED COMPONENTS

- Patching is a monthly, quarterly, or undefined task, leaving organizations exposed to known vulnerabilities
- Ensuring 3rd party libraries are up to date is often neglected as a process due to time and similar constraints
- Updating one library could take a few minutes, a few hours or a few weeks to fix the software


https://techcrunch.com/2021/12/13/the-race-is-on-to-patch-log4shell-as-attacks-begin-to-rise/

nch+

The race is on to patch Log4Shell, the bug that's breaking the internet

Carly Page @carlypage_ / 10:22 AM PST • December 13, 2021

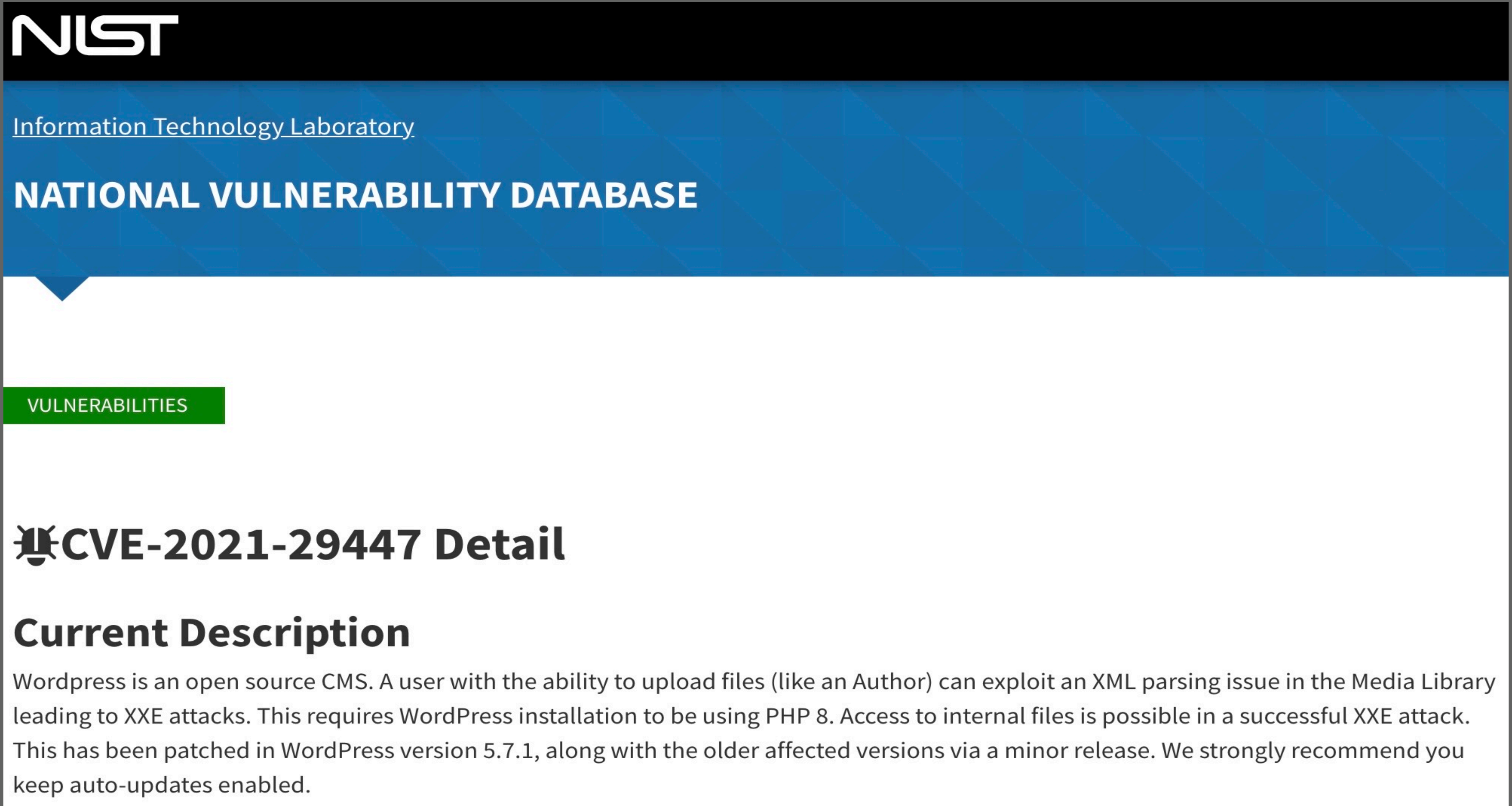
Comment

Space

Image Credits: Getty Images

https://techcrunch.com/2021/12/13/the-race-is-on-to-patch-log4shell-as-attacks-begin-to-rise//

Third Party Library Security in the NVD Database



NIST

Information Technology Laboratory

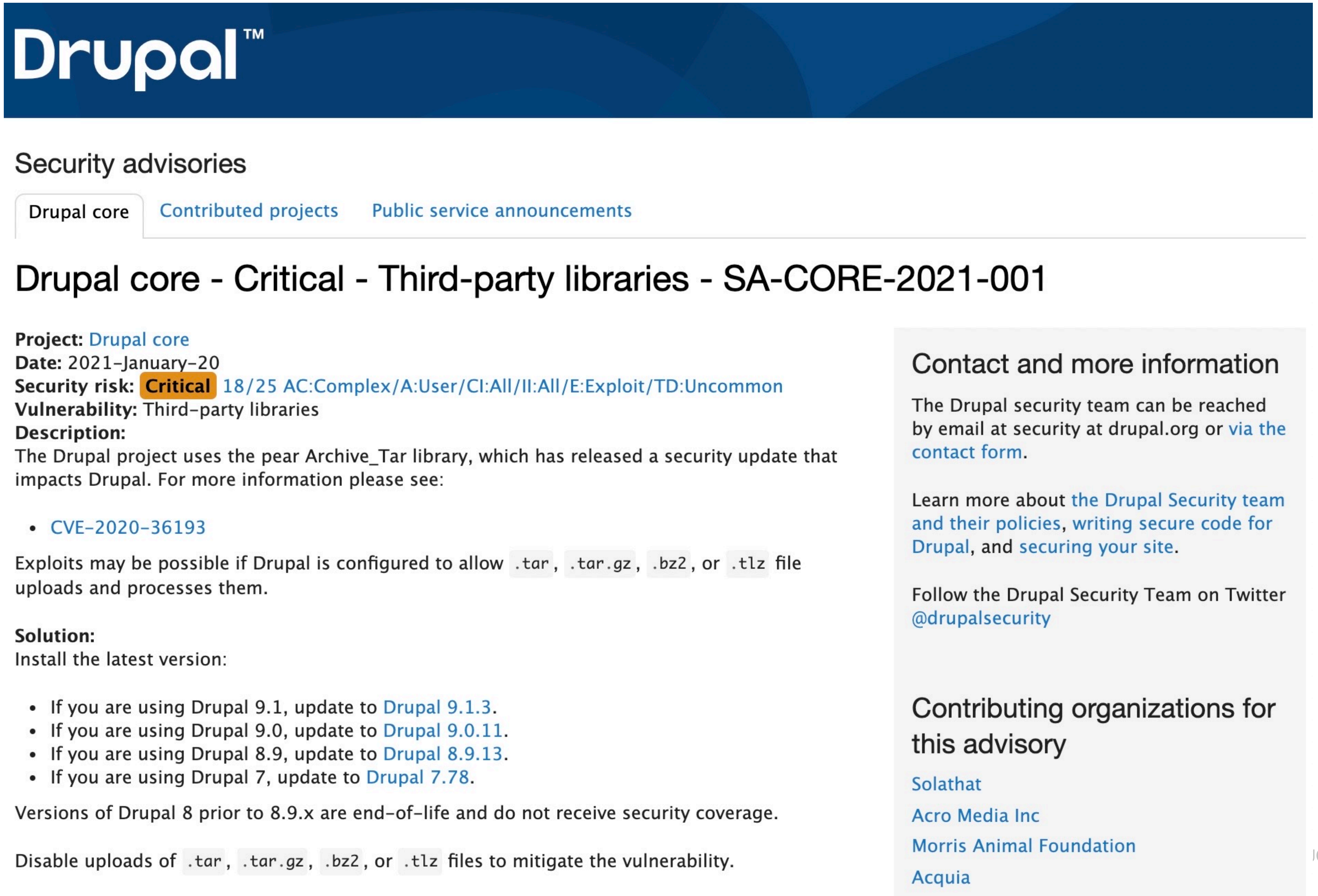
NATIONAL VULNERABILITY DATABASE

VULNERABILITIES

CVE-2021-29447 Detail

Current Description

Wordpress is an open source CMS. A user with the ability to upload files (like an Author) can exploit an XML parsing issue in the Media Library leading to XXE attacks. This requires WordPress installation to be using PHP 8. Access to internal files is possible in a successful XXE attack. This has been patched in WordPress version 5.7.1, along with the older affected versions via a minor release. We strongly recommend you keep auto-updates enabled.



Drupal™

Security advisories

Drupal core | Contributed projects | Public service announcements

Drupal core - Critical - Third-party libraries - SA-CORE-2021-001

Project: Drupal core
Date: 2021-January-20
Security risk: **Critical** 18/25 AC:Complex/A:User/CI:All/II:All/E:Exploit/TD:Uncommon
Vulnerability: Third-party libraries
Description:
The Drupal project uses the pear Archive_Tar library, which has released a security update that impacts Drupal. For more information please see:

- CVE-2020-36193

Exploits may be possible if Drupal is configured to allow `.tar`, `.tar.gz`, `.bz2`, or `.tlz` file uploads and processes them.

Solution:
Install the latest version:

- If you are using Drupal 9.1, update to Drupal 9.1.3.
- If you are using Drupal 9.0, update to Drupal 9.0.11.
- If you are using Drupal 8.9, update to Drupal 8.9.13.
- If you are using Drupal 7, update to Drupal 7.78.

Versions of Drupal 8 prior to 8.9.x are end-of-life and do not receive security coverage.

Disable uploads of `.tar`, `.tar.gz`, `.bz2`, or `.tlz` files to mitigate the vulnerability.

Contact and more information

The Drupal security team can be reached by email at security at drupal.org or via the contact form.

Learn more about the Drupal Security team and their policies, writing secure code for Drupal, and securing your site.

Follow the Drupal Security Team on Twitter @drupalsecurity

Contributing organizations for this advisory

Solathat
Acro Media Inc
Morris Animal Foundation
Acquia

Why care about third-party library security?

solarwinds

- 18,000 organizations compromised
- Pentagon, State Department, Department of Homeland Security, Microsoft

EQUIFAX

- 145 million client records stolen
- $5 billion reduction in market cap
- $200 million increase in annual spent
- $700 million dollar fine

Anthem

- 79 million health records breached
- 90 systems compromised
- $260 million increase in annual spend

Kaseva

- 1 million+ computers impacted
- $70 million ransom

JBS

- Largest meat producer shut down
- $11 million paid
- Did not follow audit advice

FORTNITE

- 200 M client
- records stolen

TESCO

easyJet

Nintendo

Application components are the #1 cause of breaches

- Application security traditionally **neglected**
- External software components form a **large attack surface** with 80-90% of code coverage and are an **attractive target** for cyber criminals

Security vulnerabilities from external software components cause more than 24% of all cybersecurity breaches

— OWASP

90%

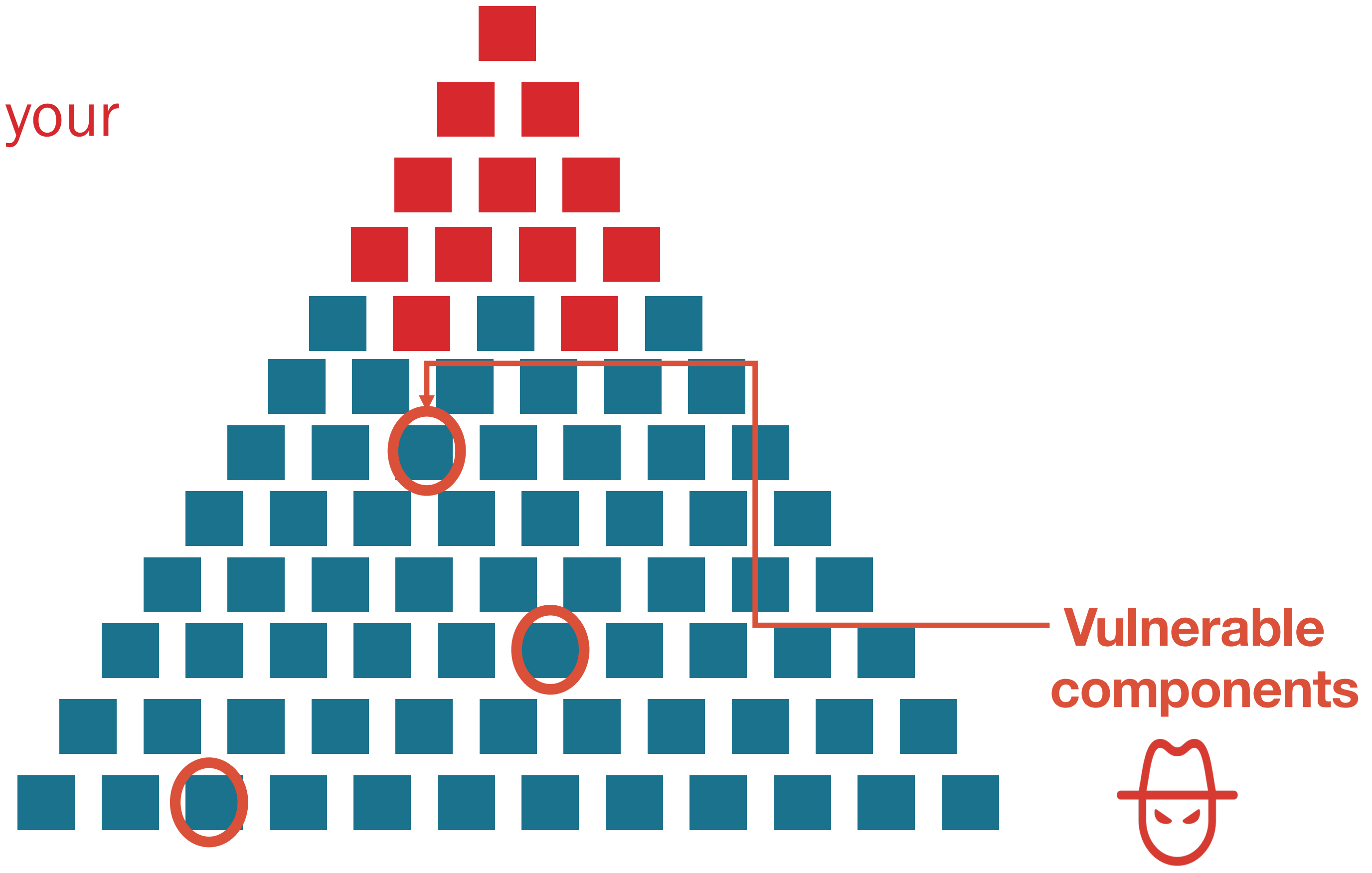
of all enterprises use open source

— Gartner

External components form a large attack surface

10-20% of the application is your business specific code

80 to 90% of the application is developed by others, mostly open source libraries, used to reduce cost and time to market



Components are attractive targets for criminals

They can shop for tools
in the "exploit economy"

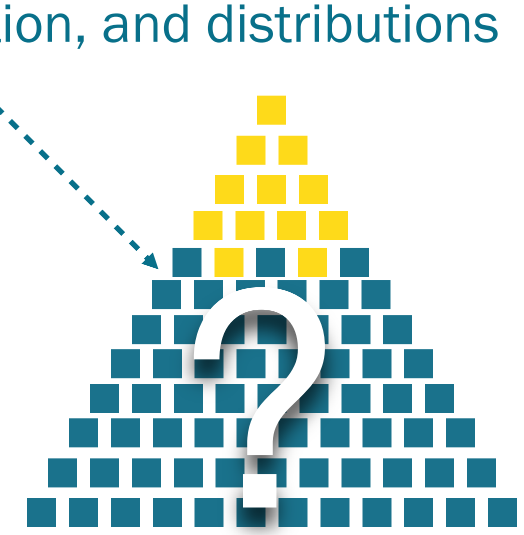
With low effort
you can find security
Bugs in libraries

Simple repeatable process

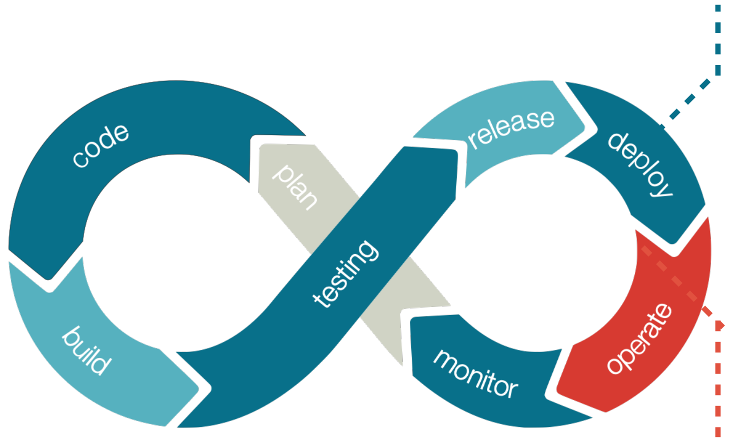
Criminals have
thousands of targets

Problem: Enterprise defense often not optimal

Lacks understanding of the complex dependencies variation, and distributions



Typically controls are applied around time of deployment



Vulnerabilities tend to materialize later and miss the controls

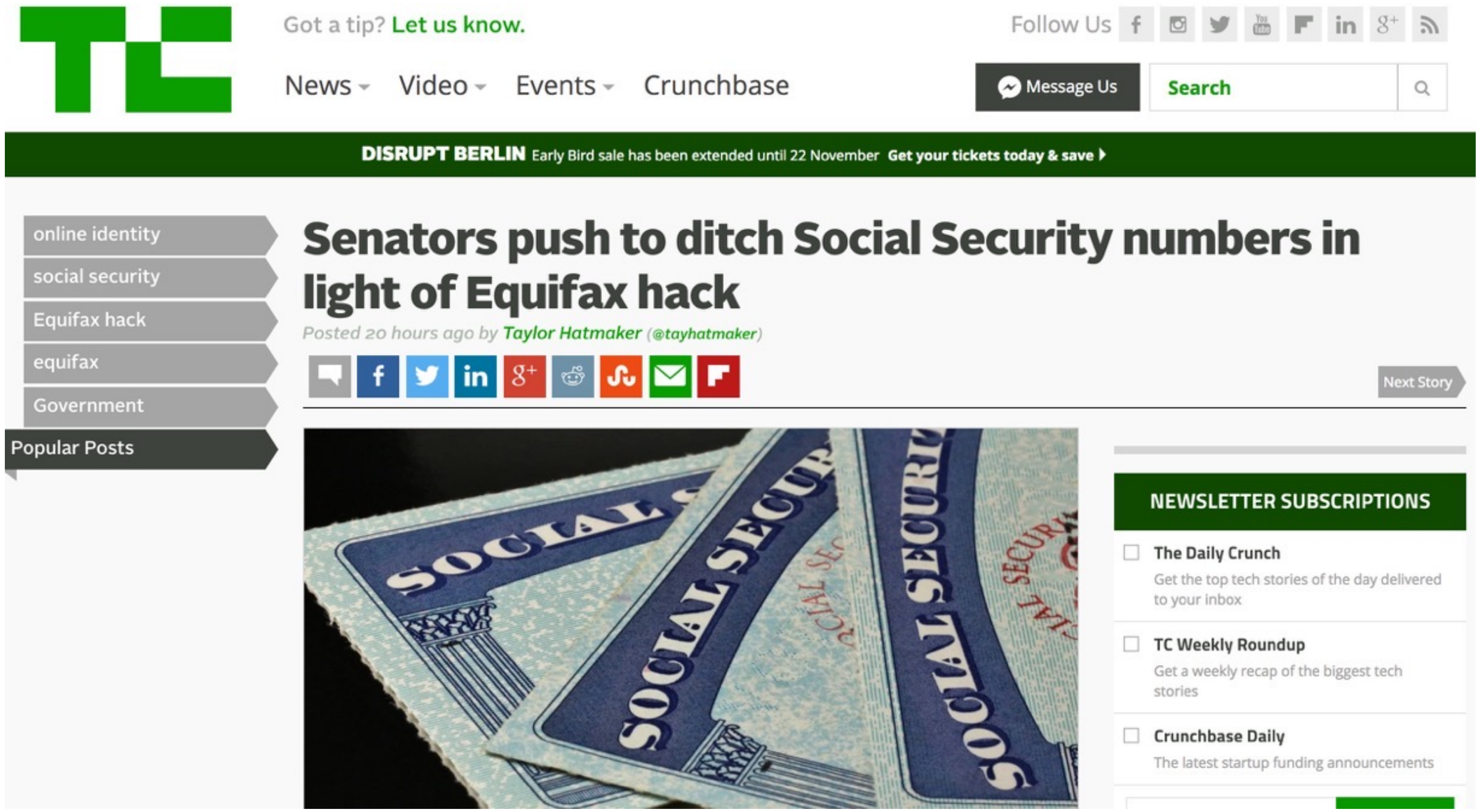
Jim’s Law of Library Scanning

“You need to keep scanning applications for third party library security every single day even when you are not working on that project.

You also want to consider not allowing a merge and/or release unless the projects libraries have no security bugs and are up to date.”

Why should we care about third-party library security?

CVE-2017-5638
Remote Code Execution (RCE) Vulnerability in Apache Struts 2


Got a tip? Let us know.

News Video Events Crunchbase

Follow Us

Message Us Search

DISRUPT BERLIN Early Bird sale has been extended until 22 November Get your tickets today & save

online identity
social security
Equifax hack
equifax
Government
Popular Posts

Senators push to ditch Social Security numbers in light of Equifax hack

Posted 20 hours ago by Taylor Hatmaker (@tayhatmaker)

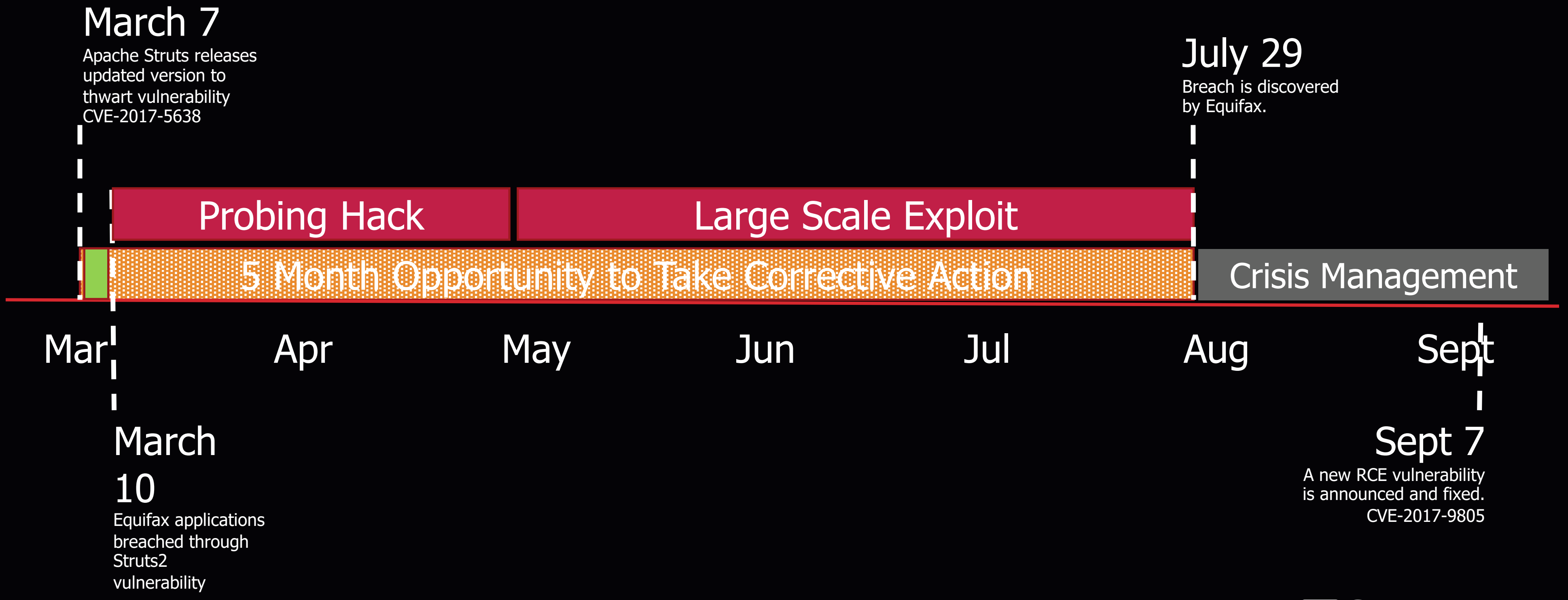
Next Story

NEWSLETTER SUBSCRIPTIONS

The Daily Crunch
Get the top tech stories of the day delivered to your inbox

TC Weekly Roundup
Get a weekly recap of the biggest tech stories

Crunchbase Daily
The latest startup funding announcements

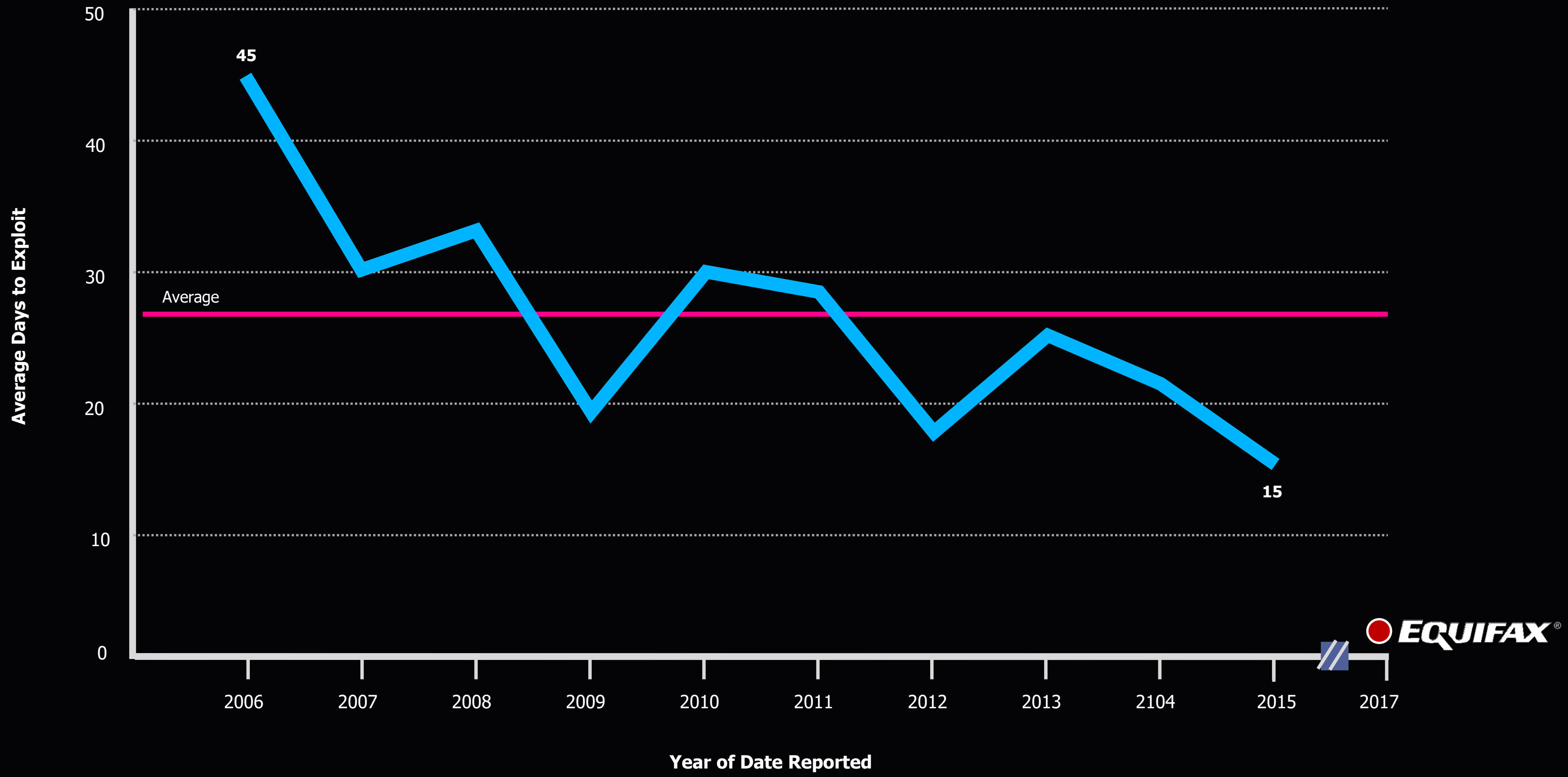

March 7
Apache Struts releases updated version to thwart vulnerability CVE-2017-5638
July 29
Breach is discovered by Equifax.
Probing Hack
Large Scale Exploit
5 Month Opportunity to Take Corrective Action
Crisis Management
Mar
Apr
May
Jun
Jul
Aug
Sept
March 10
Equifax applications breached through Struts2 vulnerability
Sept 7
A new RCE vulnerability is announced and fixed. CVE-2017-9805

EQUIFAX®

Sonatype

3 DAYS **BEFORE** EXPLOIT

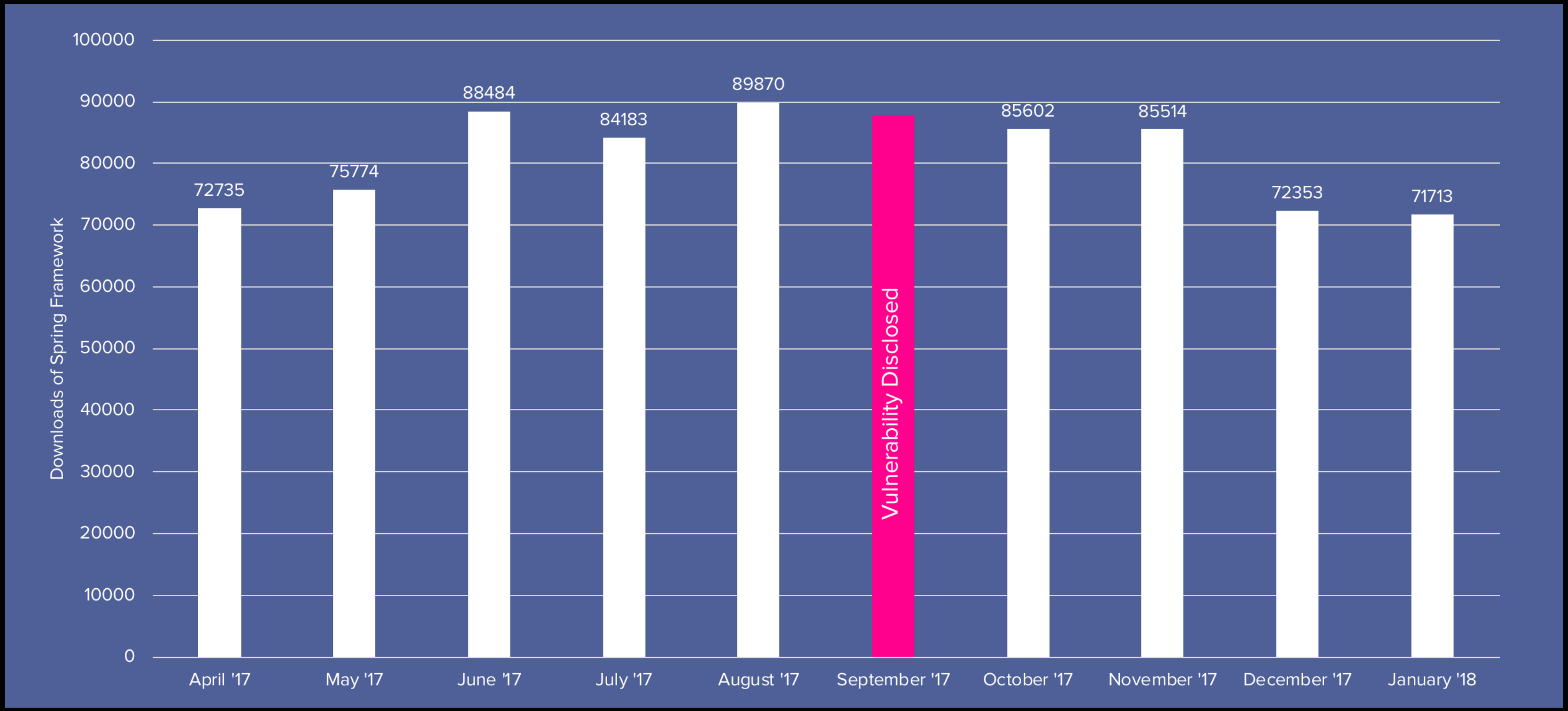
Source: Adapted from IBM X-Force / Analysis by Gartner Research (September 2016)



Sonatype

VULNERABLE SPRING FRAMEWORK DOWNLOADS

CVE-2017-8046



Source: Maven Central Repository, March 2018

Sonatype

OWASP Top 10 2013

A9 – Using components with known vulnerabilities

- Prevalence: Widespread
- Detectability: Difficult

Updating a Library is Difficult for 3 reasons

- Lack of awareness
- Updating may break your software
- No will to confront this problem head on

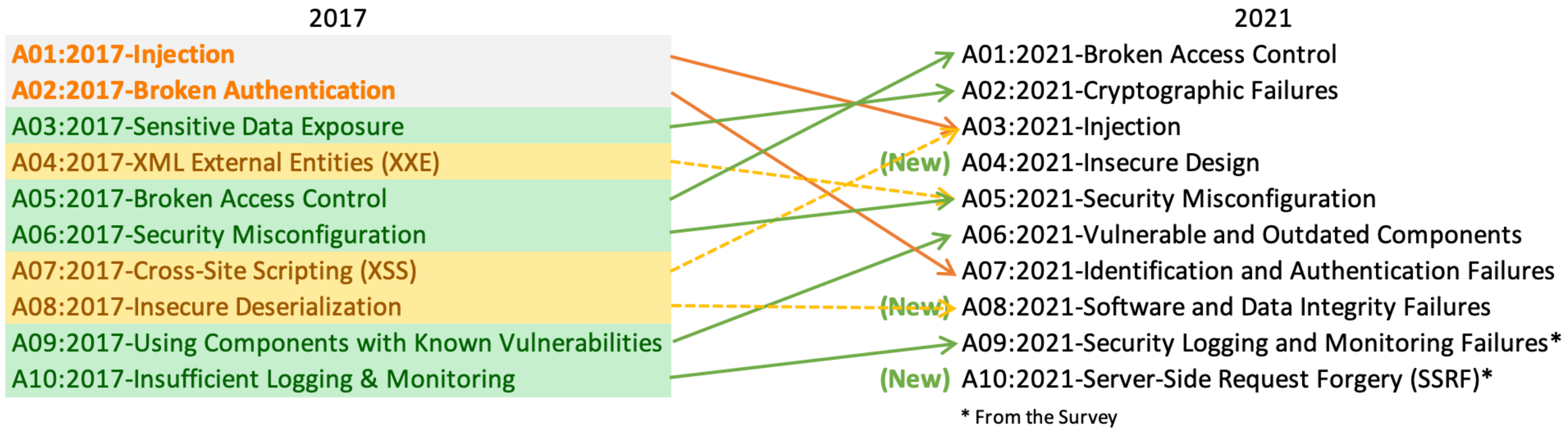
A06:2021-Vulnerable and Outdated Components was previously titled Using Components with Known Vulnerabilities and is #2 in the Top 10 community survey, but also had enough data to make the Top 10 via data analysis. This category moves up from #9 in 2017 and is a known issue that we struggle to test and assess risk.

https://owasp.org/www-project-top-ten/

OWASP Top Ten 2022

- A1: Broken Access Control
- A2: Cryptographic Failures
- A3: Injection
- A4: Insecure Design
- A5: Security Configuration
- A6: Vulnerable and Outdated Components
- A7: Authentication Failures
- A8: Software and Data Integrity Failure
- A9: Security Logging and Monitoring Failures
- A10: Server Side Request Forgery

OWASP Top Ten 2022



2017

A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
A06:2017-Security Misconfiguration
A07:2017-Cross-Site Scripting (XSS)
A08:2017-Insecure Deserialization
A09:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring

2021

A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
(New) A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
(New) A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

Patching Programs

Generally, do not cover application dependencies

- Lack of awareness of third-party or FOSS application dependencies
- Patching teams cannot push patches

Patching application dependencies requires

- Possible code changes
- Full regression testing
- May take a LONG time to update even one library!

Vetting Third-Party Libraries for Security

Vetting Need and Security of Third-Party Libraries

Do you really need it?

How may dependencies are needed with that one library?

Review Security history beforehand

Various aspects to review

- Past security public reports
- Pentest or other assessment reports
- CVE and other databases with reports on library
- Security email list activity for this library
- Does it scan clean?
- Is this library even under active management? Are there recent updates?

Third-Party Library Security Verification

Jim’s Law of Updatability

“The longer you wait to update any library (the more you skip non-security updates); the more of a time-consuming technical challenge it will be to actually update it. Plus, you do not pick up unpublished security updates.”

Jim’s Law of Library Scanning

“You need to keep scanning applications for third party library security every single day even when you are not working on that project.

You also want to consider not allowing a merge unless the projects libraries have no security bugs and are up to date.”

Enter OWASP dependency-check v8.1.3 released March 2023

- Project updated 2023
- As of 2017 this is full on Flagship OWASP Project
- Performs Software Composition Analysis
  - Reports known vulnerabilities
- Easy solution to the OWASP 2021 Top 10 **A6: Vulnerable and Outdated Components**
- Works as:
  - Maven Plugin
  - Gradle Plugin
  - Jenkins Plugin
  - SBT Plugin
  - Ant Task
  - Command Line

Language/Technology Support

Fully Supported

- .NET
- Java
- NPM/JS
- Python
- Ruby

Experimental Analyzers

- CocoaPods
- Swift Package Manager
- PHP (composer)

Vulnerability Data Source

National Vulnerability Database (NVD)

https://nvd.nist.gov

Contains a listing of Common Vulnerability and Exposures (CVE)

Each CVE entry contains

- A description of the vulnerability or exposure
- A Common Vulnerability Scoring System (CVSS) score
- A list of the affected platforms identified by their Common Platform Enumeration (CPE)

Library Identification Problems

Development & Security use different identifiers

Development (GAV coordinates):

org.springframework:spring-core:3.2.0.RELEASE

Security uses Common Platform Enumeration (CPE):

- cpe:/a:springsource:spring_framework:3.2.0
- cpe:/a:pivotal:spring_framework:3.2.0
- cpe:/a:pivotal_software:spring_framework:3.2.0

No publicly available database exists to map between the two

Evidence Based Identification Issues

False Positives

- Evidence extracted may cause incorrect identification

False Negatives

- If key elements are not included in the dependency (e.g. jar, dll) the library will not be identified and may result in un-reported risk

Onboarding an Application

Basic steps

- Configure plugin
- Proxy configuration
- Run initial scan
- Get your fancy SBOM pdf
- Identify Insecure and Out-of-Date Libraries and Components
- Plan the upgrade for identified vulnerable components
- DO THE ACTUAL UPDATE AND HOPE YOU DID NOT BREAK ANYTHING

Use Cases for dependency-check

Prove the existence of the problem

Baseline test when conducting POCs with commercial solutions

OWASP dependency-check is used as the primary tool to identify known vulnerable components

Snyk, MergeBase, JFrog and others

Enterprise Deployments

Use a centralized database to maintain the local copy of the NVD

- Single instance of dependency-check used to update
- Scanning instances do not need to update

Use an internal Nexus instead of Maven Central

Run dependency tool within CI

Continuous monitoring/reporting using **OWASP dependency-check sonar plugin**, OWASP dependency-track, or ThreadFix

Third-Party Management Tools

Java Third-Party Management Tools

OWASP dependency-check

- http://jeremylong.github.io/DependencyCheck/

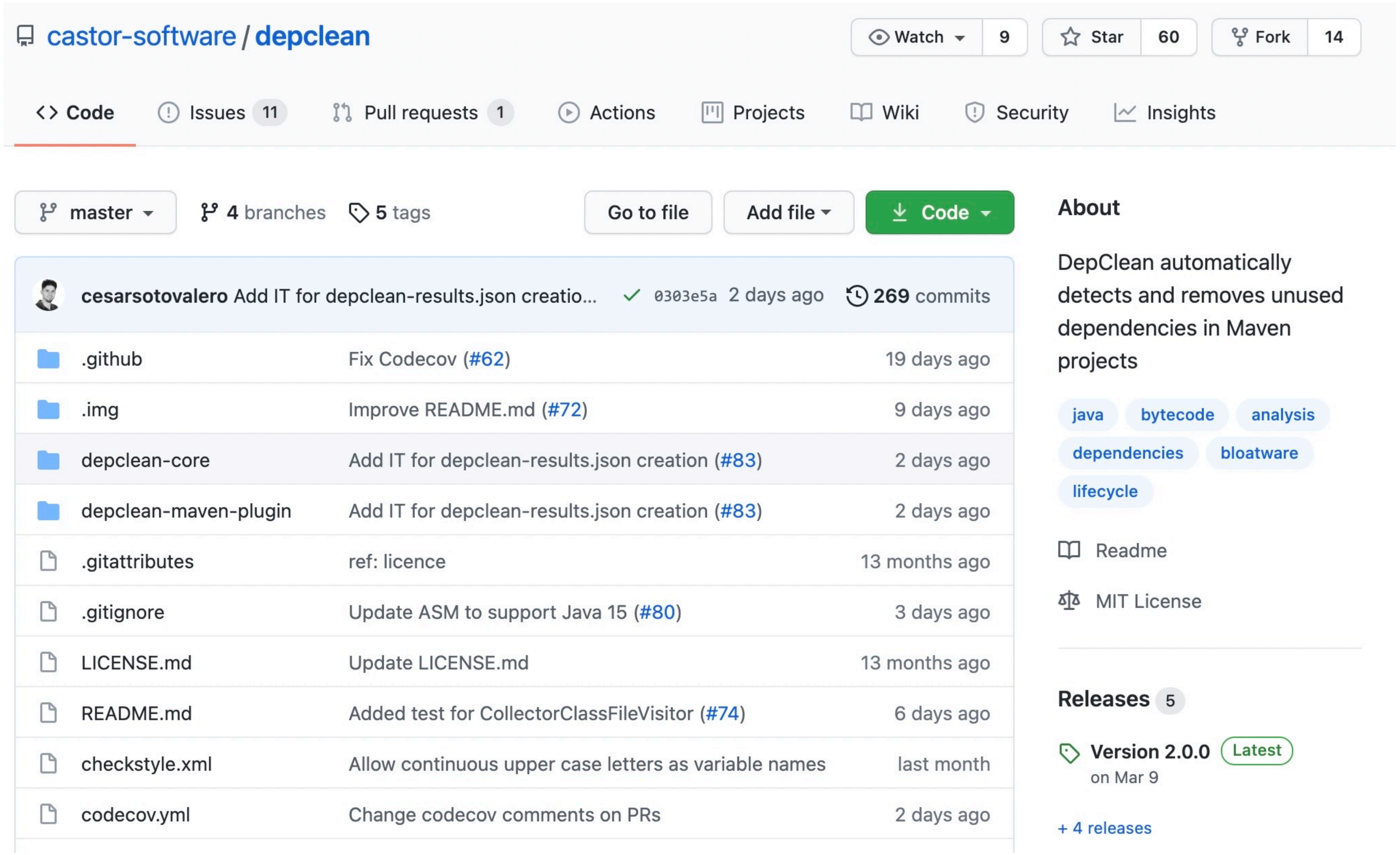
OWASP dependency-track

- https://github.com/stevespringett/dependency-track

OWASP dependency-check-sonar-plugin

- https://github.com/stevespringett/dependency-check-sonar-plugin

Depclean from Castor Software – Use Less Libs


castor-software / depclean

Watch 9 | Star 60 | Fork 14

Code | Issues 11 | Pull requests 1 | Actions | Projects | Wiki | Security | Insights

master | 4 branches | 5 tags | Go to file | Add file | Code

cesarsotovalero Add IT for depclean-results.json creatio... ✓ 0303e5a 2 days ago 269 commits

.github	Fix Codecov (#62)	19 days ago
.img	Improve README.md (#72)	9 days ago
depclean-core	Add IT for depclean-results.json creation (#83)	2 days ago
depclean-maven-plugin	Add IT for depclean-results.json creation (#83)	2 days ago
.gitattributes	ref: licence	13 months ago
.gitignore	Update ASM to support Java 15 (#80)	3 days ago
LICENSE.md	Update LICENSE.md	13 months ago
README.md	Added test for CollectorClassFileVisitor (#74)	6 days ago
checkstyle.xml	Allow continuous upper case letters as variable names	last month
codecov.yml	Change codecov comments on PRs	2 days ago

About

DepClean automatically detects and removes unused dependencies in Maven projects

java | bytecode | analysis | dependencies | bloatware | lifecycle

Readme

MIT License

Releases 5

Version 2.0.0 Latest
on Mar 9

+ 4 releases

.NET Third-Party Management Tools

OWASP dependency-check

http://jeremylong.github.io/DependencyCheck/

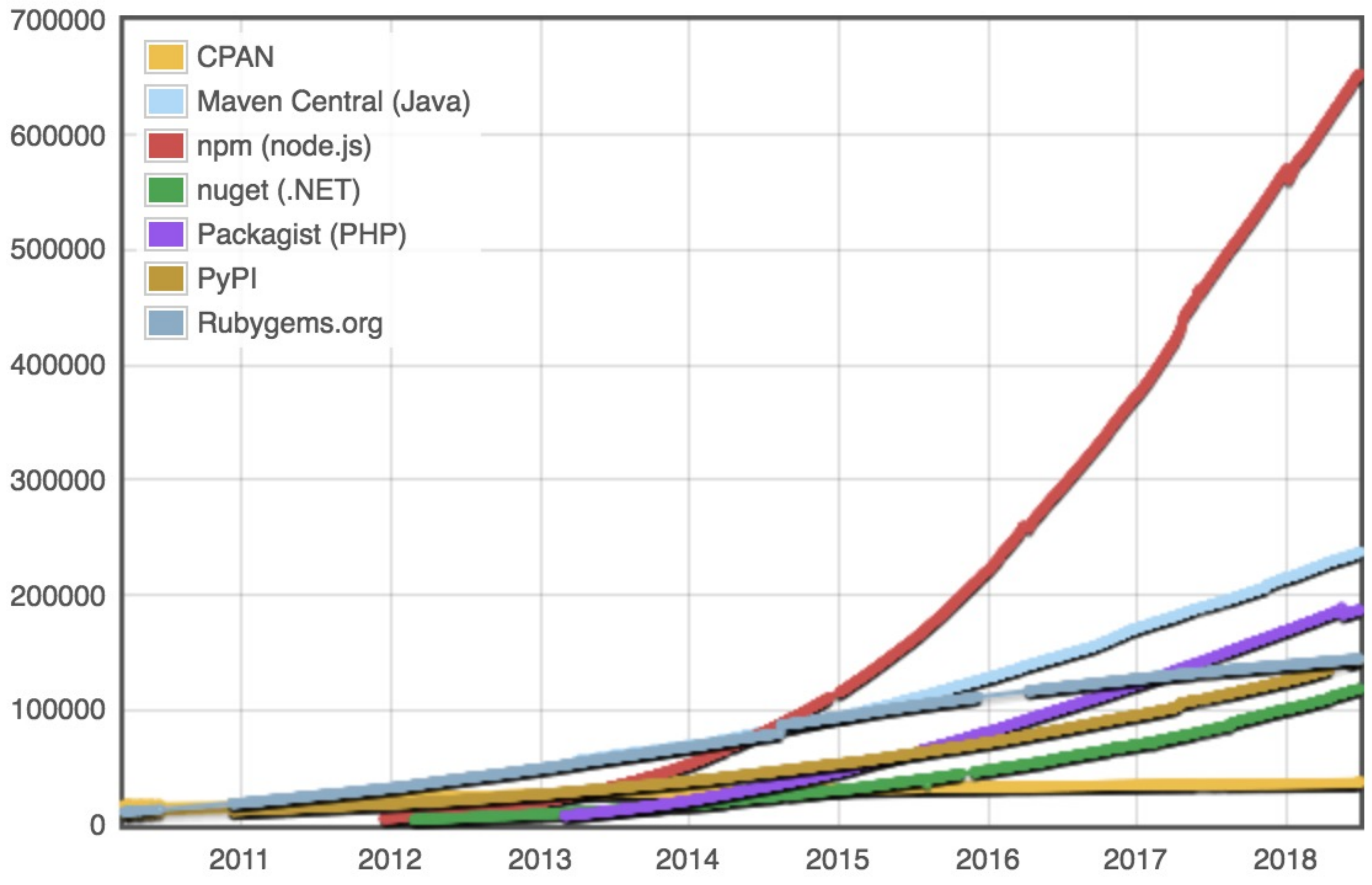
NuGetDefense

https://github.com/digitalcoyote/NuGetDefense

.Visual Studio has a built-in third-party lib scanner

https://devblogs.microsoft.com/nuget/how-to-scan-nuget-packages-for-security-vulnerabilities/

Module Counts



JavaScript Third-Party Management Tools

Retire.js (JavaScript third-party library analysis)

https://retirejs.github.io/retire.js/

Scan your project for vulnerabilities

https://docs.npmjs.com/cli/audit

Create PRs for your dependencies automatically

https://dependabot.com/

Python and Go Third-Party Management Tools

PyUp Safety

https://pypi.org/project/pyupio/

GoLang

https://github.com/golang/vuln

Ruby on Rails Third-Party Management Tools

Bundler Audit

https://github.com/rubysec/bundler-audit

OWASP dependency-check

http://jeremylong.github.io/DependencyCheck/

Create PRs for your dependencies automatically

https://dependabot.com/

Subresource Integrity

Subresource Integrity

```
<script

src="https://example.com/example-framework.js"

integrity="sha384-
oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"

crossorigin="anonymous">

</script>
```

https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity

Process Tradeoffs

Completely Ignore the Problem of Third-Party Libs

- Less updates needed (good)
- When updates ***are*** needed for security or other reasons, it's more work the longer you wait (bad)
- Slow update processes do not pick up unpublished security updates (bad)
- You never know when you have a problem (bad)
- *Slow updates are less vulnerable to supply chain attacks (good)*

Update Only When Told to by SCA Tools

- Less updates needed (good)
- When updates ***are*** needed for security or other reasons, it's more work the longer you wait (bad)
- Slow update processes do not pick up unpublished security updates (bad)
- Slow updates are less vulnerable to supply chain attacks (good)

Update Super Fast – Stay at the Bleeding Edge

- More update time needed (bad)
- Easier to update if you do it often (good)
- More vulnerable to supply chain attacks (bad)
- Bleeding edge updates pick up unpublished security updates (good)

Update 1 or 2 months behind the Bleeding Edge

- More update time needed (bad)
- Requires special tooling (bad)
- Limits supply chain attacks (good)
- If there is a vulnerability in your current version you may need to update to the bleeding edge version right away (bad)
- Easier to update if you do it often (good)

Use Less Third-Party Libraries

- Only use when needed (good/bad)
- *More of your code can be scanned by SAST (good)*
- Less opaque code in your system (good)
- *Easier on the updates (good)*
- Longer development times (?)

Tools

- Consider having more than one SCA tool scanner, they often have overlap
- Consider next generation SCA tools that understand if the vulnerable library you use effects your actual software
- Negotiate HARD with Snyk, they are one of the best but charge unreasonable prices that vary wildly

Emerging technology

- If you wish to minimize updating, keep an eye on SCA tools that determine exploitability in your codebase
- But don't do this if you believe in unpublished security updates
- Consider assured open source libraries such as Google's https://cloud.google.com/assured-open-source-software
- Be aware that no matter what choice you make in this area you are going to struggle, there is no clean solution here

What about your role in others supply chains?

- Who is using your software components?
- How does your product security update cycle effect others?
- If there is a security bug in one of your components, how fast can you update for your customers?
- Are you ready to provide reporting and public messaging when there is a security bug in your software?

Conclusion

Takeaways

- Make developers aware of the massive risk of using third party libraries
- Use less third party libraries and create a more formal vetting process before allowing new libraries
- Err on the side of updating your libraries more often than not
- Be ware of libraries that are end of life and swap them out with new libraries
- Build regression testing around your use of third party libraries

Third-Party Library Learning Objectives

Learn the security significance of the use of third-party libraries

Learn how to vet third-party libraries before using them

Learn how to conduct security verification of your third-party libraries

Learn process tradeoffs regarding the management of third-party libraries

MANICODE

SECURE CODING EDUCATION

It’s been a pleasure

jim@manicode.com