Security automation frameworks

General edition

author: Riccardo ten Cate
R1805
Introductionairy

- Security consultant at Xebia
- Hacker, trainer, speaker, security engineer, coder, project leader OWASP S.K.F
- Quirky
Also me

i'm not a procrastinator

i just prefer doing all my work in a deadline-induced panic

invisiblebread.com
The challenge of automation
Why do DevSecOps/Security automation

- Short feedback loop
- Empower your team
- Make applications secure by design
- Eliminate technical dept
- Regular check-ups
The (S)SDLC

- **Test automation (code quality)**
  - Dead end code
  - Over complex code
  - Repudiated code

- **Security test automation**
  - SAST tooling
  - DAST tooling

- **Manual verification**
  - Security code audit
  - Penetration test
Static Analyzer Security Tooling

- Fortify
- Veracode
- Checkmarx
- OWASP Dependency checker
- FindBugs
- Snyk/Retire/Node security
- ...

SAST
Dynamic Analyzer Security Tooling

- Acunetix
- OWASP ZAP
- AppSpider
- HP WebInspect
- Burp
- Nessus
- OpenVAS
CI/CD tooling

Jenkins

GitLab

Travis CI
The setup

1) Pull Project Code

2) Launch Docker(s)

3) Push Findings

4) Validated Findings

Source of Truth

Weaponized Jenkins

Project Repo

Tests run in Docker containers

Zap Docker

Test Web App Projects

Optional Notifications

Project Slack Channel

Project Jira

JIRA
But ideally!
False positives you say? What is that?
OWASP Benchmark project

OWASP WBE Results Interpretation Guide

- Tool reports everything is vulnerable
- Ideal vulnerability detection
- Tool reports vulnerabilities randomly
- Worse than random
- Tool reports nothing is vulnerable
Yeah...

In bird culture that is what we call a "Dick Move"
Defect Dojo

- An OWASP project
- Supports a lot of tools
- Easy to deploy
- False positive suppression
- Delta reporting
Well hello!
Show me metrics plz!

Dependency Check Scan: test_automation

<table>
<thead>
<tr>
<th>Environment</th>
<th>Engagement</th>
<th>Target Start Date</th>
<th>Target End Date</th>
<th>Progress</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>Engagement: DefDev engagement (May 01, 2018)</td>
<td>May 1, 2018</td>
<td>May 1, 2018</td>
<td>100%</td>
</tr>
</tbody>
</table>

Findings

<table>
<thead>
<tr>
<th>Name</th>
<th>Reporter</th>
<th>Mitigation Date</th>
<th>Severity</th>
<th>Verified</th>
<th>Active</th>
<th>Duplicate</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>jruby.jar</td>
<td>admin</td>
<td>None</td>
<td>High</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>[View][Edit][Delete]</td>
</tr>
<tr>
<td>jruby.jar</td>
<td>admin</td>
<td>None</td>
<td>Medium</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>[View][Edit][Delete]</td>
</tr>
<tr>
<td>jruby.jar</td>
<td>admin</td>
<td>None</td>
<td>Medium</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>[View][Edit][Delete]</td>
</tr>
</tbody>
</table>

Potential Findings

Add a potential finding...
**jruby.jar | CVE-2011-4838** Last reviewed today by admin

<table>
<thead>
<tr>
<th>Severity</th>
<th>Status</th>
<th>Type</th>
<th>Date discovered</th>
<th>Age</th>
<th>Reporter</th>
<th>Found by</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Active</td>
<td>Dynamic</td>
<td>May 1, 2018</td>
<td>22 days</td>
<td>admin</td>
<td>Dependency Check Scan</td>
</tr>
</tbody>
</table>

**Affected Endpoints / Systems**

No endpoints.

**Description**

CWE-20 Improper Input Validation

JRuby before 1.6.5.1 computes hash values without restricting the ability to trigger hash collisions predictably, which allows context-dependent attackers to cause a denial of service (CPU consumption) via crafted input to an application that maintains a hash table.
Please handle yer secrets and keys!

KeyWhiz
Problem #1 Secret sprawl

- Plaintext
- Config files
  - Ansible, Puppet, Chef
  - Dockerfile / Entrypoint
- Source code
- Version control systems
  - Github
  - VSTS
  - Bitbucket
Look at all this new attack surface!
If the keys are sprawled!
We have no auditability
We have no good means to revoke keys
We have applications who suck at keeping secrets!
  ○ Logs (splunk, syslog)
  ○ Error handling
  ○ Monitoring
We need a Vault in our lives!
Vault can fix all of this and more!
Vault!

- **Dynamic secrets**
  - Ephemeral
  - So we have key rotation
    - Should an application log creds they are no longer valid

- **Unique tokens**
  - We now have auditability
  - We can revoke keys that were proven to be compromised

- **All the information is encrypted**
  - In rest
  - In transit

- This just scratches the surface of what it could do!
No more talking the talk
Hope y'all had a good time!