Five Ways to Boost Cyber Security with DevOps

Aaron Volkmann & Doug Reynolds
Agenda

• Cross-Team Collaboration
• Unified Data
• Platform Hardening
• Application Security
• Monitoring
Five Ways to Boost Cyber Security with DevOps

Collaboration
Waterfall

- Requirements
- Design
- Implementation
- Verification
- Maintenance
Water - Scrum - Fall

Business

Research

Budget

Document

Development

QA and Operations

Integrate

Test

Release

Jez Humble, https://youtu.be/L1w2_AY82WY
Silos Block Collaboration
Silos Reinforce Waterfall

Teams have moved to Agile methodologies, but roles still align with waterfall methods.
DevOps is an Extension of Agile Thinking

**Agile**

**Embrace** constant change

**Embed Customer** in team to internalize expertise on requirements and domain

**DevOps**

**Embrace** constant testing, delivery

**Embed Operations** in team to internalize expertise on deployment and maintenance
DevOps Aims to Increase...

...the pace of innovation

...responsiveness to business needs

...collaboration

...software quality
DevOps Tenets

1. Reduce Silos
2. Use Automation as Much as Possible
3. Build a Little Test a Little
4. Continually Improve
5. Linked Data Between Systems
Five Ways to Boost Cyber Security with DevOps

Unified Data
Integration and communication, even among tools, is key to assuring security!
What changed since my last security scan?
Who was involved in a peer review of a change?
Is there unusual activity happening right now?
Do I have this particular vulnerable piece of software deployed in my system?
Five Ways to Boost Cyber Security with DevOps

Security Hardening
Platform Hardening Workflow

- STIG Hardening Guides
- Customization + Add New Rules
- Remove Not Applicable Rules
- Identify Exceptions
- Security Database
- Automated Implementation
- Hosts
- Automated Verification
- Continuous Monitoring

Legend
- Configuration
- Scan
- Results
Platform Hardening Workflow

STIG Hardening Guides

Customization + Add New Rules
- Remove Not Applicable Rules

Identify Exceptions

Security Database

Automated Implementation

Hosts

Automated Verification

Continuous Monitoring

https://iase.disa.mil/stigs

Covers Various Software & General

Great Starting Point for Rules
## STIG Rule Example

<table>
<thead>
<tr>
<th>ID</th>
<th>V-38451</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severity</td>
<td>Medium</td>
</tr>
<tr>
<td>Title</td>
<td>The <code>/etc/passwd</code> file must be group-owned by root.</td>
</tr>
<tr>
<td>Discussion</td>
<td>The <code>/etc/passwd</code> file contains information about the users that are configured on the system. Protection of this file is critical for system security.</td>
</tr>
</tbody>
</table>
| Fix Text | To properly set the group owner of `/etc/passwd`, run the command:  

```
# chgrp root /etc/passwd
```

[https://www.stigviewer.com/stig/red_hat_enterprise_linux_6/](https://www.stigviewer.com/stig/red_hat_enterprise_linux_6/)
Platform Hardening Workflow

- Add New Rules
- Customize How They Apply to Your System
- Remove Rules that Do Not Apply
Platform Hardening Workflow

- Rules that apply to certain hosts in the system, but not all
Platform Hardening Workflow

- Persist the plan and results
- Spreadsheets work, but become cumbersome

Legend

- Configuration
- Scan
- Results

STIG Hardening Guides → Customization + Add New Rules • Remove Not Applicable Rules → Identify Exceptions → Security Database → Automated Implementation → Automated Verification → Continuous Monitoring
Platform Hardening Workflow

- Store in Source Control
- Practice Orchestration
- Harden and unharden
Platform Hardening Workflow

- Scan for vulnerabilities
- Persist Results to Monitor Changes Over Time

Legend
- Configuration
- Scan
- Results
Platform Hardening Workflow

- Check for Unexpected Changes
- Integrate with Alerting System

Legend:
- Configuration
- Scan
- Results

STIG Hardening Guides → Customization + Add New Rules • Remove Not Applicable Rules → Identify Exceptions → Security Database → Automated Implementation → Hosts → Automated Verification

Continuous Monitoring
Platform Hardening Workflow

- Start Small
- Harden Development and Test Environments
- Practice Maintenance Activities

Legend
- Configuration
- Scan
- Results

STIG Hardening Guides

Customization + Add New Rules • Remove Not Applicable Rules

Identify Exceptions

Security Database

Automated Implementation

Automated Verification

Continuous Monitoring

Hosts
Five Ways to Boost Cyber Security with DevOps

Application Security
Secure DevOps Lifecycle
Secure DevOps Lifecycle
Secure DevOps Lifecycle

- Pausing for manual steps is typical
- Optimize the manual work!
- Persist the output of any tools / work
Dependency Management

Due to security, bug fixes, and new features, third-party dependencies keep changing.

Software dependencies can range from a large list of items:

- JavaScript Libraries from npm
- CSS Frameworks
- Python packages from PyPI
- Nuget packages
- Maven JARs
- Operating system packages (glibc/libxml2/libxslt)
- Operating system kernel versions
Dependency Management Workflow

1. Developer
2. Dependency Repository
3. INFO SEC
4. New Release
5. CI / CD
Dependency Management: Why?

Relying on third-party packages repos can be troublesome for many reasons:

• Security and integrity
• “Angry author” scenario
• Archive retention for older packages
• Uptime, connectivity, and speed
• Not suitable for internal or “proprietary” packages
Dependency Management: Security

Typosquatting, a common problem with domain names, is now available in your favorite package manager!

Malicious code was uploaded to PyPI using commonly misspelled package names.

Ten Malicious Libraries Found on PyPI - Python Package Index

By Catalin Cimpanu

https://bit.ly/2xCTGHT
Dependency Management: More Security

External packaging is signed, so it is okay, right?
In a perfect world, yes. In today’s world, maybe not!

- D-Link
- Yahoo!
- Linux Mint
Dependency Management: “Angry author”

March 22, 2016 was the day that the Internets broke.
One author decided to remove his JavaScript packages from npm.

The absence of the left-pad package broke many dependency chains.
The author was within his rights to remove the packages from npm.
Dependency Management: Maintenance Mode

Eventually, project development is completed, leaving a finished product. Two years in the future, it has to be deployed to another server. Do you know where all of your dependencies are stored, because they aren’t available from the original external repo!

- Open-source projects generally move very fast or very slow.
- Slow moving projects will have the same version for years as they are feature complete.
- Fast moving projects will often release major version every year, and the API will not be compatible with the previous major release.
Five Ways to Boost Cyber Security with DevOps

Monitoring
Monitoring: Be the all-seeing eye

Once an application is deployed and running, no news is good news, right? Unfortunately, that is often a metric that is used to measure an application’s uptime or functionality. While everything is “working,” the following things are chipping away at your application’s security:

- Running out of disk space, memory, or swap space
- HTTP 401, 403, and 500 responses are going unnoticed
- Malicious network probes are trying to find a way into your network
- Malware is trying to find its way out of your network
- Your app dependencies have new security fixes
Monitoring: One Screen to Rule Them All

Collaboration is the key to DevOps, and likewise with monitoring.

- All of your monitoring statistics and alerts need to be visible from one place.
  - Avoids monitoring fatigue
  - Allows easy review of metrics
  - Prevents scrambling for the correct tool

- Most monitoring functionality can be achieved with a combination of open-source tools and extended with plugins.

Along with StatsD/Graphite and ElasticSearch/Logstash/Kibana (ELK)
Monitoring: Storage Space

Storage is relatively inexpensive. However, running out of storage could be costly.

“Things” that burn memory or disk space:

- Logs
- Data / database journals
- Backups
- OS Patches
- Swap / page files
- Message queues
- Core / crash / heap dumps
- User uploads

An out of space or memory issue could be a signal that something is out of the ordinary.

- DoS/DDoS attack
- Coding errors
- Buffer overrun/underrun
- Software exploits
- Malware
- System configuration issue
Monitoring: The HTTP Request

Monitoring request status and the quantity of requests of your application provide a base line measurement. An increase of requests or certain types of requests can indicate problems:

- Password dictionary attacks (HTTP 401)
- Directory traversal attacks (HTTP 401/403)
- Application error or misuse (HTTP 400/500)
- DDoS/DoS - exponential increase in requests (HTTP 401/403/500)
- Code deployment error – decrease in requests or increase in errors
Monitoring: Network

The network is the gateway in and out of your enterprise. State roads and highways have traffic and stoplight cameras. Your network should be no different! Monitoring some items will help you establish thresholds for alerts so to avoid “alert” fatigue.

- Failed DNS lookups
- Proxy bypass attempts
- Destinations dropped by your outbound firewall
- Login failures
- Network intrusion detection
Monitoring: Dependencies

Applications and program libraries are constantly being fixed for security issues. Using a dependency database, you can scrape data feeds for updates or CVEs:

- Python Package Index (PyPI) [https://pypi.python.org/pypi?%3Aaction=rss](https://pypi.python.org/pypi?%3Aaction=rss)
- node package manager (npm) [https://registry.npmjs.org/-/rss](https://registry.npmjs.org/-/rss)
- RubyGems [https://rubygems.org/gems/package_name/versions.atom](https://rubygems.org/gems/package_name/versions.atom)
- Packagist (php) [https://packagist.org/feeds/releases.atom](https://packagist.org/feeds/releases.atom)
- CPAN (perl) [https://metacpan.org/feed/recent](https://metacpan.org/feed/recent)
Contact Information

Presenters

Aaron M. Volkmann
DevOps Team Lead
Telephone: +1 412.268.8993
Email: amvolkmann@cert.org

Douglas J. Reynolds
Software Engineer
Telephone: +1 412.268.2824
Email: djreynolds@sei.cmu.edu