AppSec Pipeline

Doug Morato
Sr. Manager
PwC NIS|App-Sec
OWASP Tampa Meeting - 02/19/2016
Who am I again?

Professionally:
Sr. Manager - AppSec team @ PwC

Prior roles:
Sr. Software Sec Consultant @ HP
Sr. Penetration Tester @ Mastercard
App Sec Specialist @ Disney
Independent AppSec Consultant

Certs:
CSSLP, CISSP, GPEN, GCIA, GCFA, GCIH, GSEC,
CCSK, ECSA, CEH Certified

Personally:
Born in Brazil (Yes, I speak Portuguese!)
Happily married
Father of the most awesome 6 year old ever
Live in South Florida (Boca Raton)
Core contributor to the OWASP WebGoat Project

Hobbies:
InfoSec, Travel and Beers

Disclaimer: Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are mine individually and, unless expressly stated to the contrary, are not the opinion or position of my Employer, its staff, or its member. PwC does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.
Acknowledgments

This presentation is an aggregation of multiple presentations and ideas I have seen presented, and customized to our own necessity using my own judgment and my team’s feedback,

There is awesome content out there from people who have been doing and are doing AppSec pipelines, which some of it’s concepts and ideas I mention here.

Thanks and props to the following:

- Matt Tesauro:
  - [http://www.slideshare.net/mtesauro/](http://www.slideshare.net/mtesauro/)

- Matt Konda:
  - [https://speakerdeck.com/mkonda](https://speakerdeck.com/mkonda)

- Aaron Weaver

- Josh Corman
Appsec Pipeline

What’s that all about?

Remember Henry Ford?

Founder of Ford Motor company and sponsor of the development of the assembly line
Why AppSec Pipeline?

Key drivers for us:

10-Fold increase in # of apps
Differences in SDLC maturity across territories.
Some territories and Dev teams require Agile dev and rapid delivery.
Appsec Pipeline

A quick intro to “The Phoenix Project Book” concepts:
The 3 Ways of DevOps

1. **Workflow:** Look at your purpose and those processes which aid it

2. **Improve on feedback:** Open yourself to upstream and downstream information

3. **Continual Experimentation & Learning:** ADAPT. Create a cultoe of innovation and experimentation
How can we securely support the new model of ever-changing, agile initiatives, continuous delivery and DevOps?
Automation

Human capital is the critical resource, but also the most expensive, even when using offshore resources.

Computational resources are cheap.

Automate time-consuming tasks where/when possible.
AppSec in the SDLC
AppSec Pipeline. How it works:

Standard Build

Standard Dev Build
Project goes through standard dev and build process, committing code changes as they go through sprints/cycles.
Scheduled or triggered builds upon code push.

AppSec Tasks

AppSec Pipeline Tasks
Perform AppSec tasks if standard build successful:
- Static Code Analysis (HP Fortify / FOD)
- Dynamic Scans (HP WebInspect, OWASP ZAP...)

Release or Act Upon

Approve artifact or Act
Approve inbound artifact into “blessed” artifact repository if “all good” OR
Trigger alternate workflow, which can be manual review or reassign to AppDev team
I am rugged and, more importantly, my code is rugged. I recognize that software has become a foundation of our modern world. I recognize the awesome responsibility that comes with this foundational role.

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.

I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic and national security.

I recognize these things – and I choose to be rugged.

I am rugged because I refuse to be a source of vulnerability or weakness. I am rugged because I assure my code will support its mission. I am rugged because my code can face these challenges and persist in spite of them. I am rugged, not because it is easy, but because it is necessary and I am up for the challenge.
Technology Stack

Products and Services
Proposed Tools and Vendor Solutions:

<table>
<thead>
<tr>
<th>Svc Delivery</th>
<th>Build</th>
<th>SAST</th>
<th>DAST</th>
<th>Reporting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engagement Tracking</td>
<td>Build Server</td>
<td>Source-Code</td>
<td>Tools</td>
<td>Tools</td>
</tr>
<tr>
<td>JIRA</td>
<td>Bamboo, Jenkins or TFS</td>
<td>HP Fortify</td>
<td>HP WebInspect</td>
<td>Custom Portal</td>
</tr>
<tr>
<td>JIRA Service Desk</td>
<td>Deployment</td>
<td>HP FoD</td>
<td>Acunetix</td>
<td>Kenna/RiskIO</td>
</tr>
<tr>
<td>Bag of Holding*</td>
<td>Puppet, Chef or VSRM</td>
<td>Binary/COTS</td>
<td>OWASP ZAP</td>
<td>SonarQube</td>
</tr>
<tr>
<td>Documentation</td>
<td>Code Repositories</td>
<td>Veracode</td>
<td>Arachni</td>
<td>Archer</td>
</tr>
<tr>
<td>Confluence</td>
<td>GIT/GitHub, TFS, SVN</td>
<td></td>
<td>BURP Suite</td>
<td>Threadfix*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Core Impact</td>
<td></td>
</tr>
</tbody>
</table>

* Or similar
**Possible Offerings:**
- Atlassian Confluence
- Secure Development Training

**Possible Offerings:**
- Atlassian Bamboo, SonarQube, JFrog Artifactory, BlackDuck

**Possible Offerings:**
- Atlassian Bamboo, WASA Portal, HP WebInspect, OWASP ZAP, Burp Suite, Acunetix, Arachni

**Possible Offerings:**
- RASP
- WAF
- Yearly tests

---

**Design**
- Developer gets source-code from code repository.
- Works on backlog.
- Builds locally, runs SAST

**Possible Offerings:**
- Atlassian Jira
- SonarQube
- HP FoD
- Secure Code Library

**Development**
- Standard CI Build.
- Performs general testing.
- Package artifact and store in Artifactory.
- Triggers AppSec Pipeline

**Possible Offerings:**
- Atlassian Bamboo, SonarQube, JFrog Artifactory, BlackDuck

**Build & Package**
- Perform Static Code Analysis automatically from build.
- Break AppSec Pipeline if Critical or High count increase

**Possible Offerings:**
- Atlassian Bamboo
- HP Fortify / HP FoD / Veracode

**SAST**
- Sign-off & Deployment
- Sign-off artifact if all testing successful.
- Move artifact to blessed artifact repo for Orchestration Consumption.
- Report metrics/quality

**DAST**
- Possible Offerings:
  - ThreadFix, PowerBI
  - Dashboard, SonarQube
  - Archer Integration

**Sustainment**
Development Cycles

The proposed workflow demonstrates how the tools and the possible service offerings integrate within the development lifecycle, whether the team is using standard (waterfall) or agile (scrum) methodology.

1. **Get Source**
   - Developer retrieves source code from a repository.

2. **Commit & Push**
   - Developer works on an item from the backlog, makes code changes.

3. **Work on Backlog**
   - Developer builds the project in the local IDE.

4. **Test Changes**
   - Developer tests code changes.

5. **Build on IDE**
   - Developers commit and push code changes to the repository.

6. **Service Offering: Atlassian Jira Ticketing System**
   - CI build is triggered by commit, schedule, or on-demand.

7. **CI Build Successful**
   - If CI build is successful, move to the next step.

8. **Build on Server**
   - If build is successful, passes tests and quality checks, move to next step.

9. **Service Offering: Atlassian Bamboo Build Server**
   - CI build is triggered by commit, schedule, or on-demand.

10. **Standard CI Build**
    - Build is triggered by commit, on-demand, by a specific schedule.

11. **AppSec Pipeline**
    - Start the AppSec pipeline process.

12. **Service Offering: Source Code Repo (GitHub / BitBucket)**
    - Service Offering: Source Code Repo (GitHub / BitBucket)
    - Developer retrieves source code.

13. **Service Offering: Atlassian Jira Ticketing System**
    - Service Offering: Atlassian Jira Ticketing System
    - Developer works on an item from the backlog, makes code changes.

14. **Service Offering: Atlassian Bamboo Build Server**
    - Service Offering: Atlassian Bamboo Build Server
    - CI build is triggered by commit, schedule, or on-demand.

15. **Service Offering: Cloud SAST (HP FoD & Veracode)**
    - Cloud SAST is 1 of the post-build actions. Additional post-build actions might not be “Security Related,” such as code quality, test coverage, dependencies check, etc.

16. **Service Offering: JIRA Ticketing and Service Desk**
    - Service Offering: JIRA Ticketing and Service Desk
    - For all interactions between clients and AppSec, JIRA Service Desk is used to standardize the process, track requests, track MTTR and SLA metrics, accountability & auditing.

JIRA is also used by App Dev teams to create and keep backlog, plan sprints, track defects.
Deployment Cycles

The proposed workflow demonstrates how the tools and the possible service offerings integrate within the deployment lifecycle.

Additionally, this diagram shows possible DevOps and Continuous Delivery integration points pulling “blessed” artifacts from Artifact Repository.
How does it look?
What others are using?

- Source: https://www.linkedin.com/pulse/appsec-pipeline-illustrated-aaron-weaver
No Money?

Open Source it’s your best friend

- Jira -> Bugzilla,
- Confluence -> Tiki Wiki
- Bamboo -> Jenkins
- Artifactory > Open-Source Artifactory
- HP WebInspect -> OWASP ZAP
- ThreadFix Open-Source
- StackStorm Open-Source
- Bag of Holding
- Gauntlt
Our next steps

The road to never ending Continuous Improvement

- “Dockerizing” this approach
- Leverage more Gauntlt
- Define aggregation and reporting strategy
- Create triggers for “auto-release” using chef/puppet
- ChatOps
- Machine Learning for False positives reduction
- BigData and BI for knowledge
- Continuous Improvement
- Rinse and repeat
Questions?
Contact Info:

Doug Morato

doug.morato@owasp.org

https://speakerdeck.com/dougmorato