OWASP
AppSec Europe
London 2nd-6th June 2018

Making Continuous Security a Reality

Aaron Weaver

Matt Tesauro

Matt Tesauro

I am Matt Tesauro

I think AppSec needs to change and

I’m going to tell you how I see it changing

matt.tesauro@owasp.org / @matt_tesauro

Aaron Weaver

Making AppSec a little better each day.

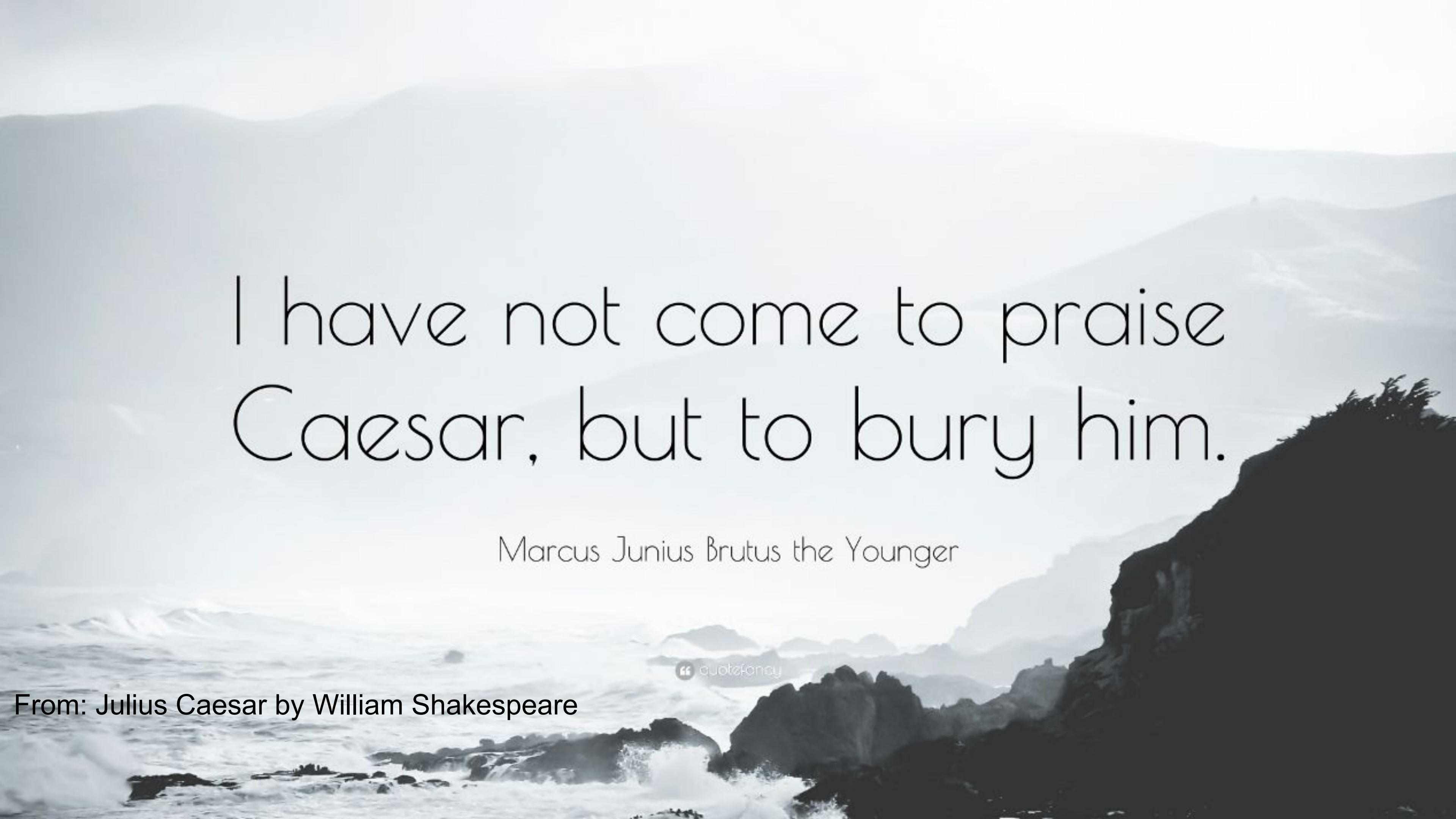
aaron.weaver@owasp.org / @weavera

Principal AppSec Engineer at 10Security

Quick survey...

- Raise your hand if you work in:
  - AppSec
  - Product Security
  - Security Engineering
  - DevOps
    aka DevSecOps,
  - SecDevOps, DevOpsSec,
    OpsDevSec...

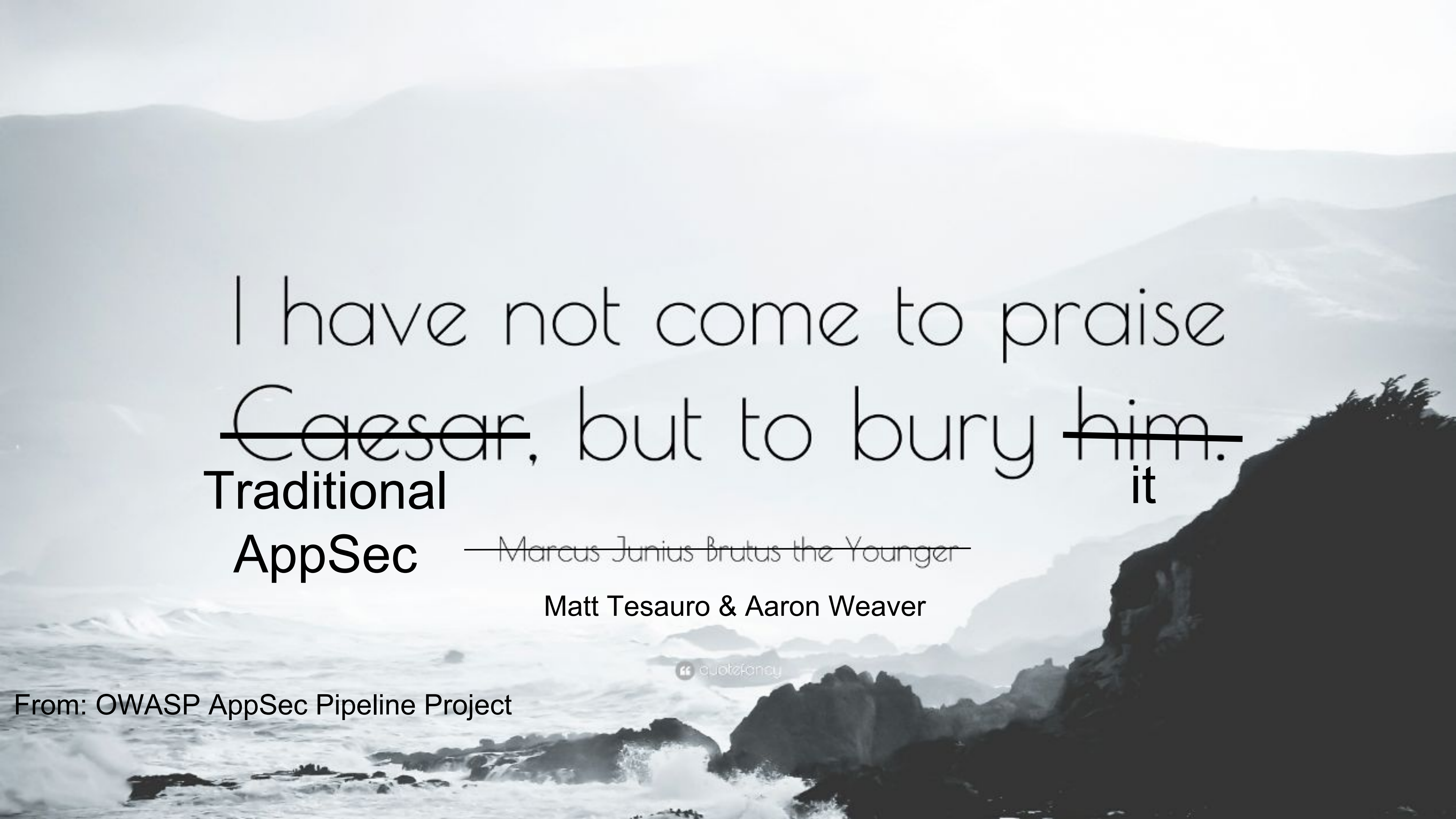
What traditional AppSec Tooling feels like



I have not come to praise Caesar, but to bury him.

Marcus Junius Brutus the Younger

From: Julius Caesar by William Shakespeare



I have not come to praise ~~Caesar~~, but to bury ~~him.~~

Traditional AppSec

it

~~Marcus Junius Brutus the Younger~~

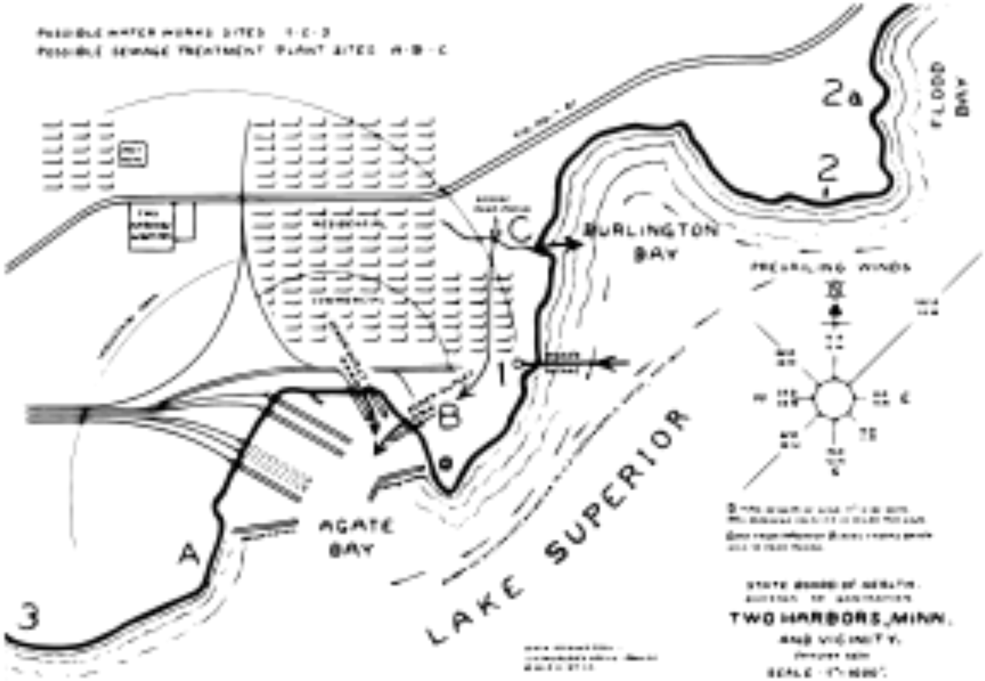
Matt Tesauro & Aaron Weaver

From: OWASP AppSec Pipeline Project

TRADITIONAL
APPLICATION
SECURITY
WE HARDLY KNEW YOU..

AppSec Pipeline

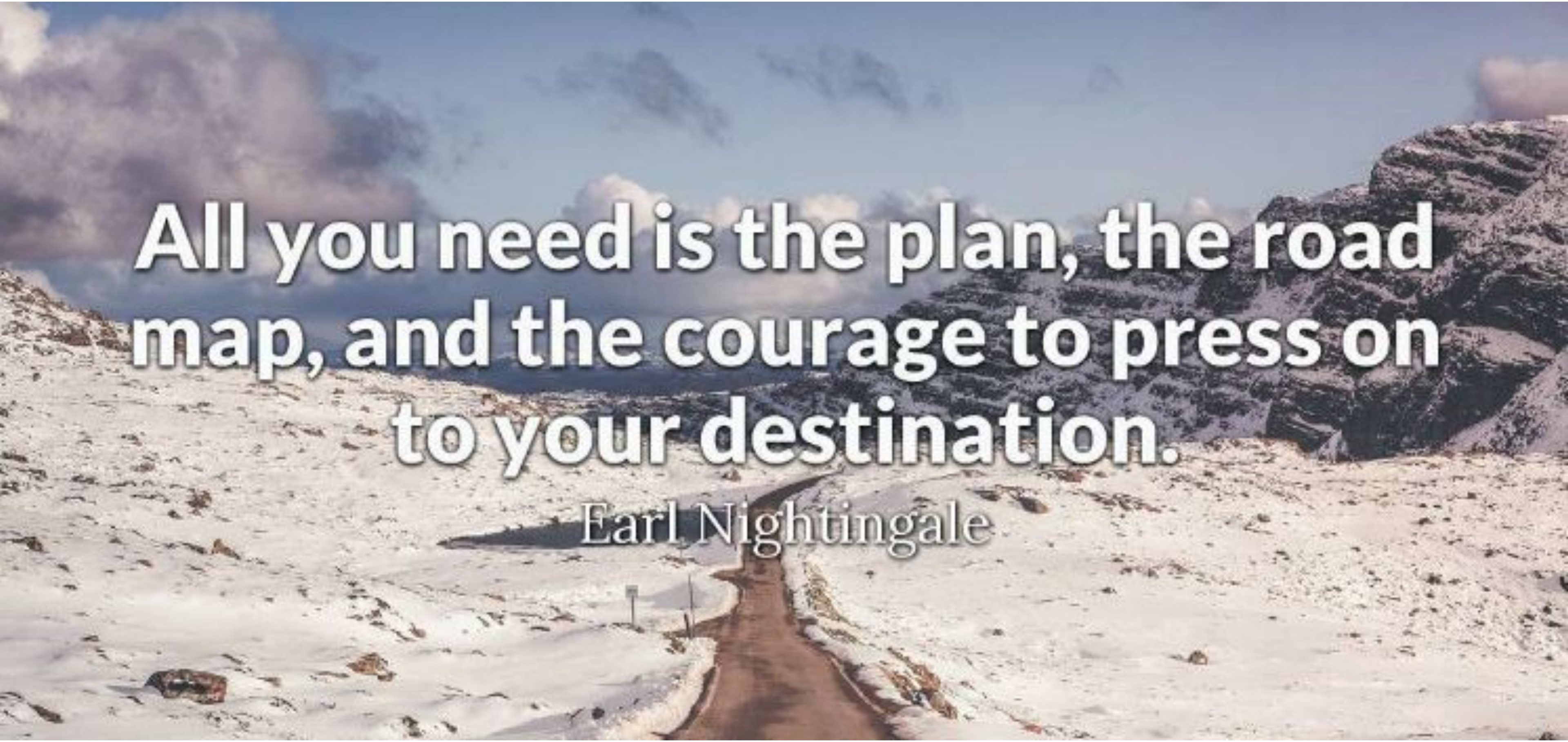
A real life example of an implemented AppSec Pipeline



The purpose of an Application Security program is to **evaluate** the security status of the suite of apps for a business.

Basically, to provide a map to **guide** business decisions

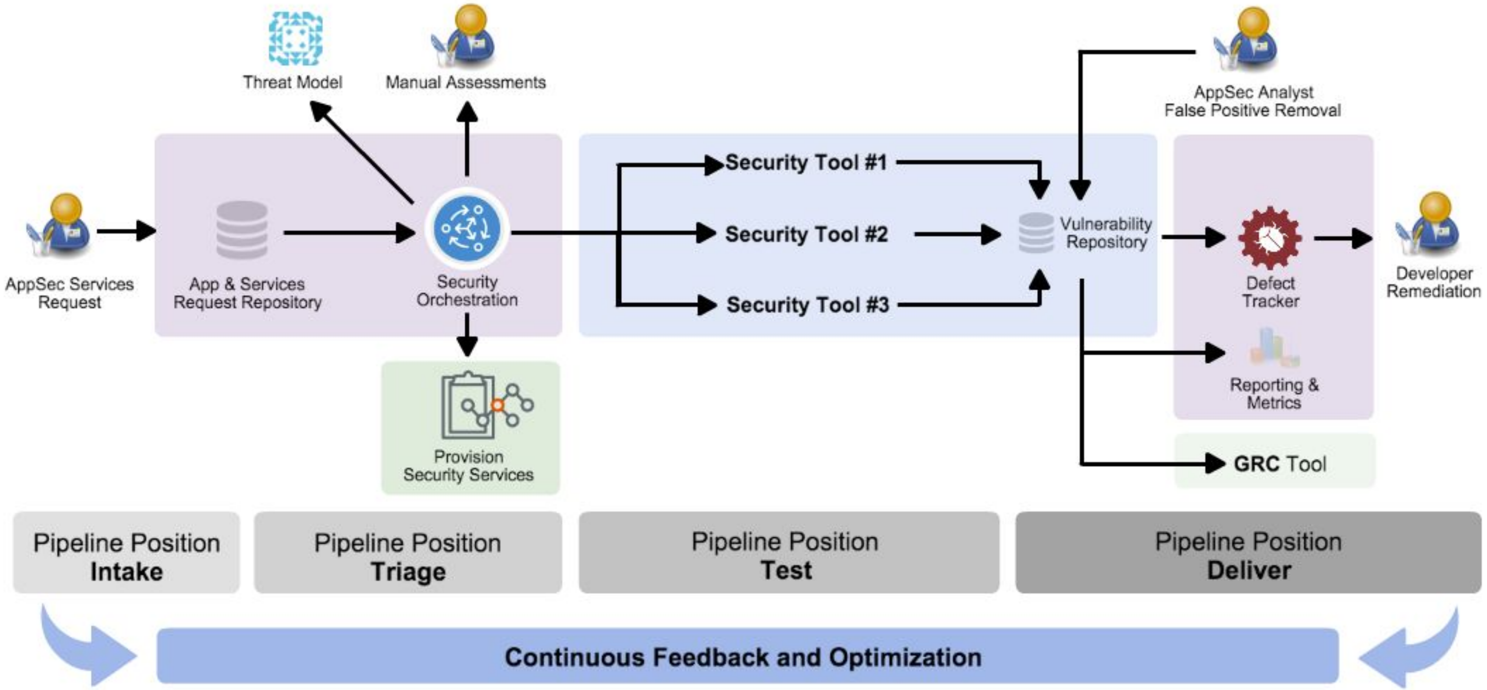
Do you have a full view of your application landscape?



All you need is the plan, the road map, and the courage to press on to your destination.

Earl Nightingale

Rugged Devops - AppSec Pipeline Template



AppSec Services Request

App & Services Request Repository

Threat Model

Manual Assessments

Security Orchestration

Provision Security Services

Security Tool #1

Security Tool #2

Security Tool #3

AppSec Analyst False Positive Removal

Vulnerability Repository

Defect Tracker

Reporting & Metrics

GRC Tool

Developer Remediation

Pipeline Position **Intake**

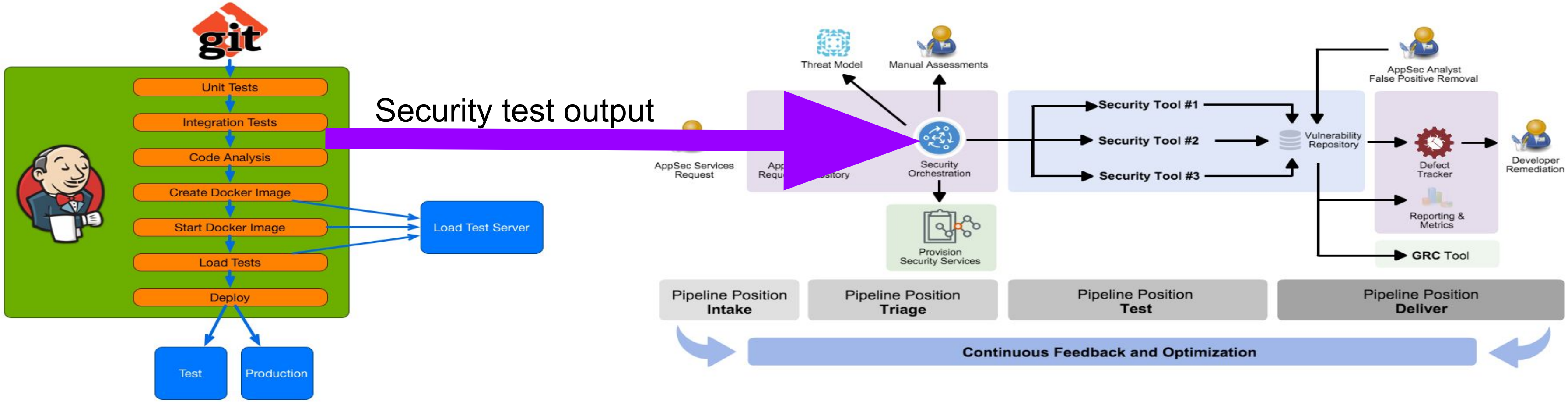
Pipeline Position **Triage**

Pipeline Position **Test**

Pipeline Position **Deliver**

Continuous Feedback and Optimization

- ▨ Partial Automation
- 🟩 Future Automation
- 🟦 Automation



git

Unit Tests

Integration Tests

Code Analysis

Create Docker Image

Start Docker Image

Load Tests

Deploy

Load Test Server

Test

Production

Security test output

Threat Model

Manual Assessments

AppSec Services Request

App Requ

sitory

Security Orchestration

Provision Security Services

Security Tool #1

Security Tool #2

Security Tool #3

Vulnerability Repository

AppSec Analyst False Positive Removal

Defect Tracker

Developer Remediation

Reporting & Metrics

GRC Tool

Pipeline Position **Intake**

Pipeline Position **Triage**

Pipeline Position **Test**

Pipeline Position **Deliver**

Continuous Feedback and Optimization

DevOps Pipeline

AppSec Pipeline

What is an AppSec Pipeline?

- A way to conduct testing in an automated fashion
- Run by the AppSec team
  for the AppSec team
- Get your house in order
  - Then reach out to dev teams

- A way to scale AppSec coverage
  - ‘You must be this high to ride this ride’
  - Pre-calculate a portion of manual testing
  - Create a security baseline across
    the application landscape

What an AppSec Pipeline isn't

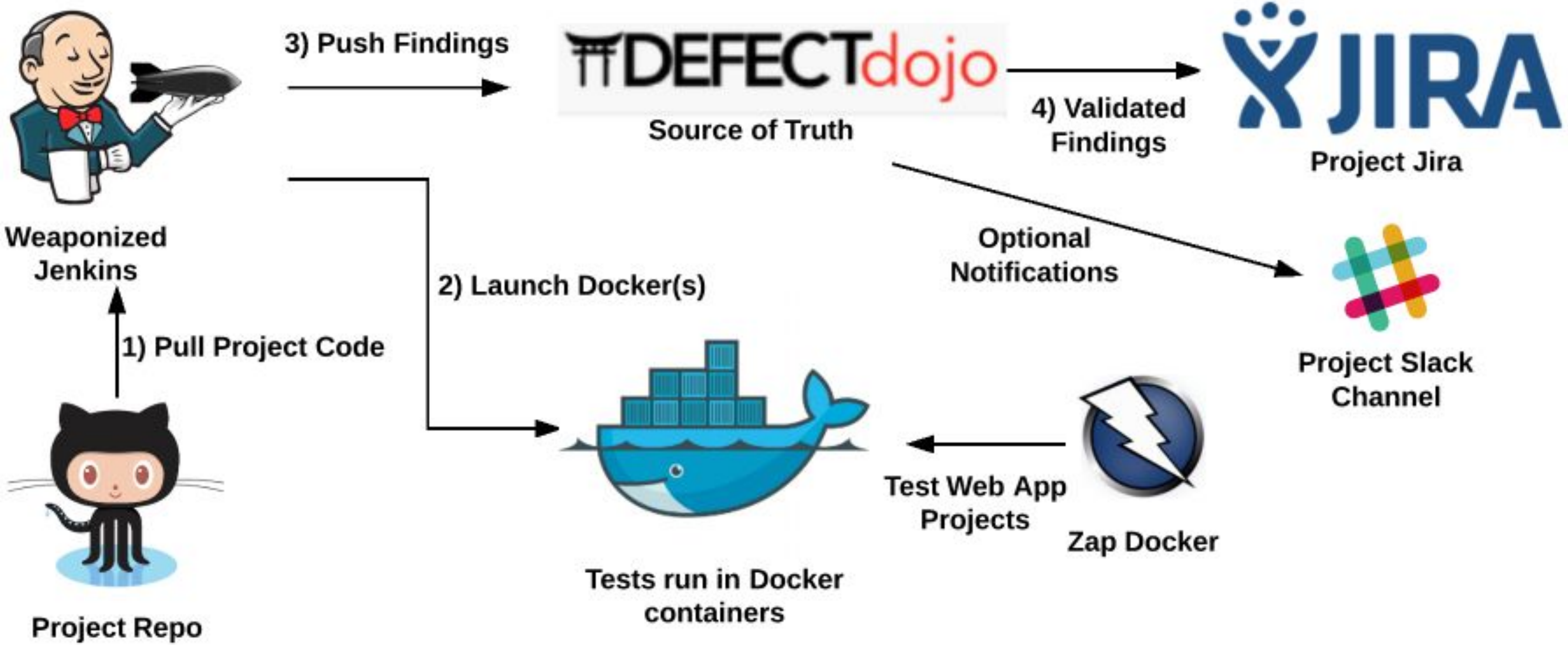
- The ***one thing*** that will fix all your problems
- A gate that blocks deploys
  (especially at first)

- Pipelines create artifact

  - CI/CD artifacts are deployed versions of an app(s)

  - AppSec Pipeline artifacts are security findings

OWASP Projects AppSec Pipeline



Weaponized Jenkins

3) Push Findings

DEFECTdojo

Source of Truth

4) Validated Findings

JIRA

Project Jira

Optional Notifications

Project Slack Channel

2) Launch Docker(s)

1) Pull Project Code

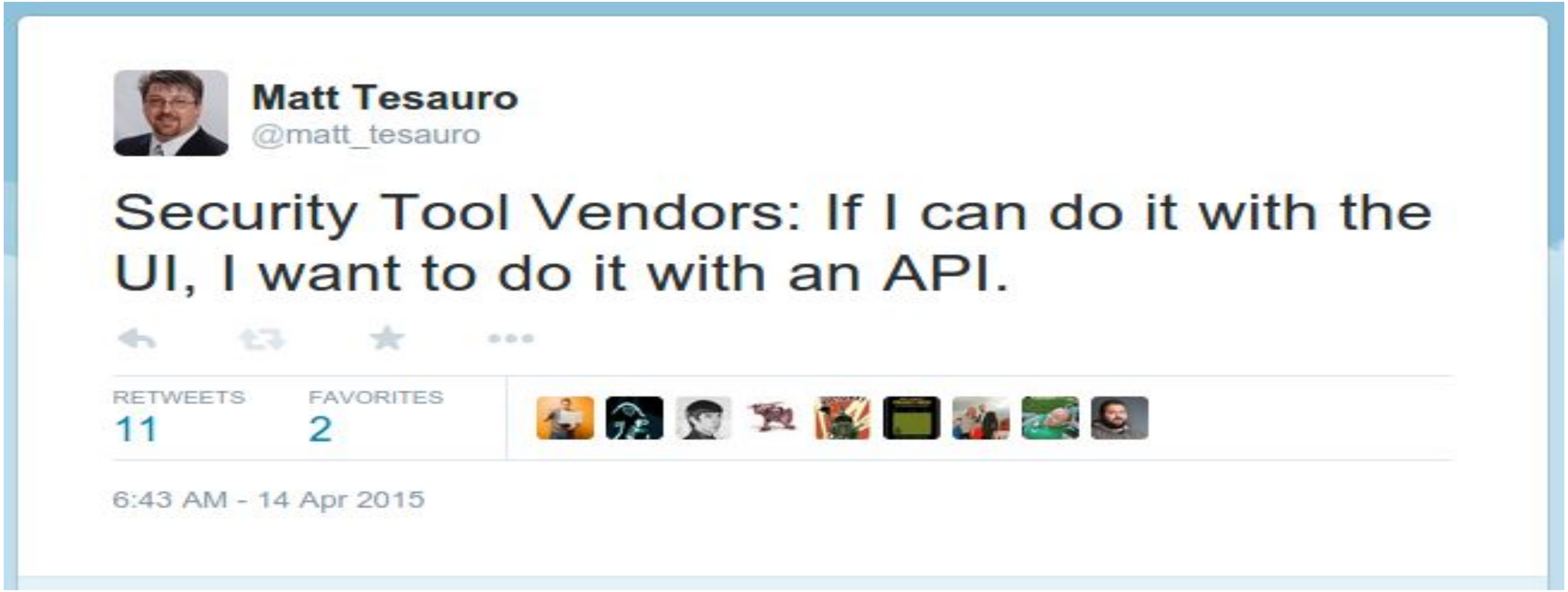
Project Repo

Tests run in Docker containers

Test Web App Projects

Zap Docker

Call to Action



Matt Tesauro
@matt_tesauro

Security Tool Vendors: If I can do it with the UI, I want to do it with an API.

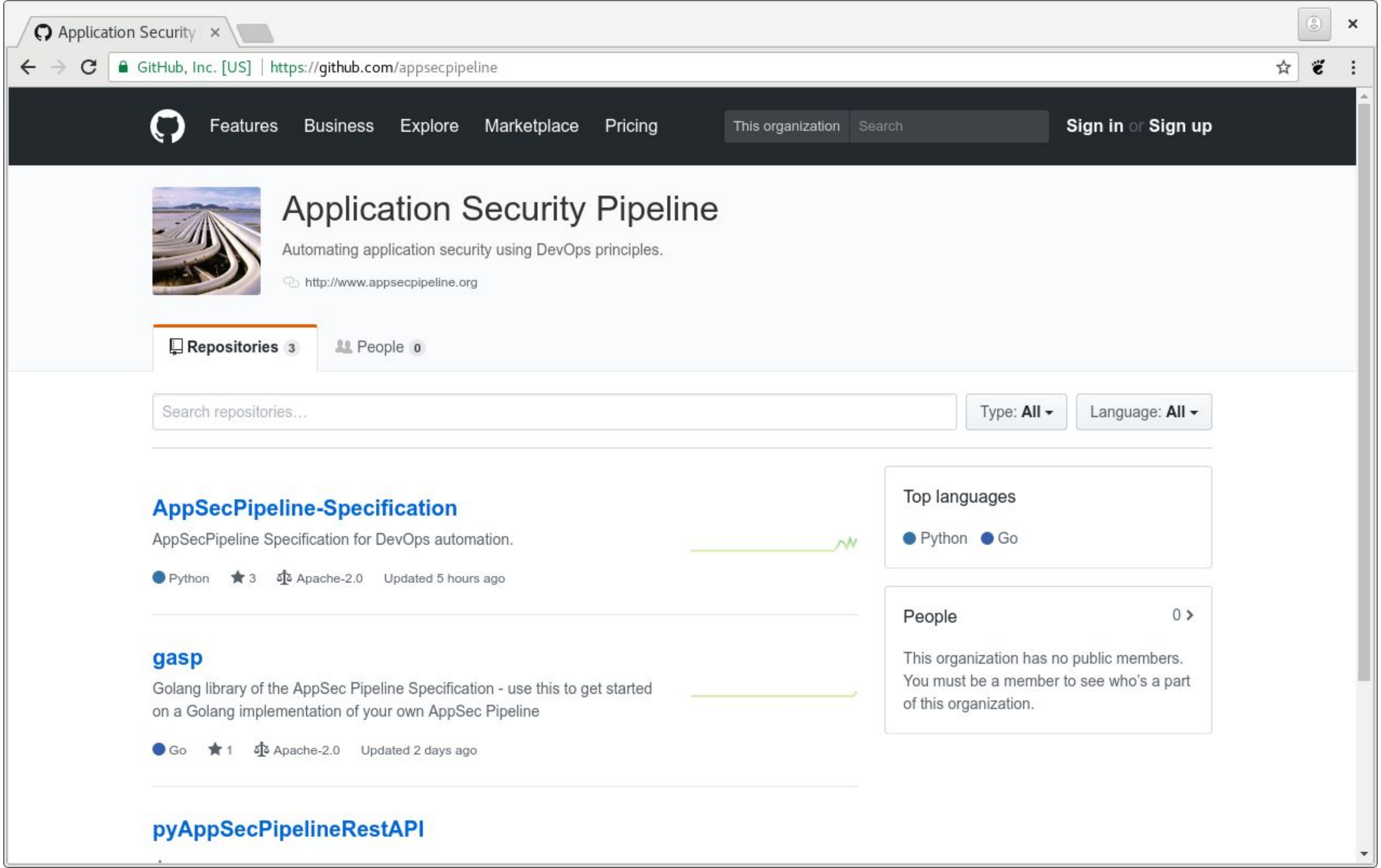
RETWEETS 11

FAVORITES 2

6:43 AM - 14 Apr 2015

Gasp

One implementation of the AppSec Pipeline Spec



Application Security Pipeline

Automating application security using DevOps principles.

http://www.appsecpipeline.org

AppSecPipeline-Specification

AppSecPipeline Specification for DevOps automation.

● Python ★ 3 Apache-2.0 Updated 5 hours ago

gasp

Golang library of the AppSec Pipeline Specification - use this to get started on a Golang implementation of your own AppSec Pipeline

● Go ★ 1 Apache-2.0 Updated 2 days ago

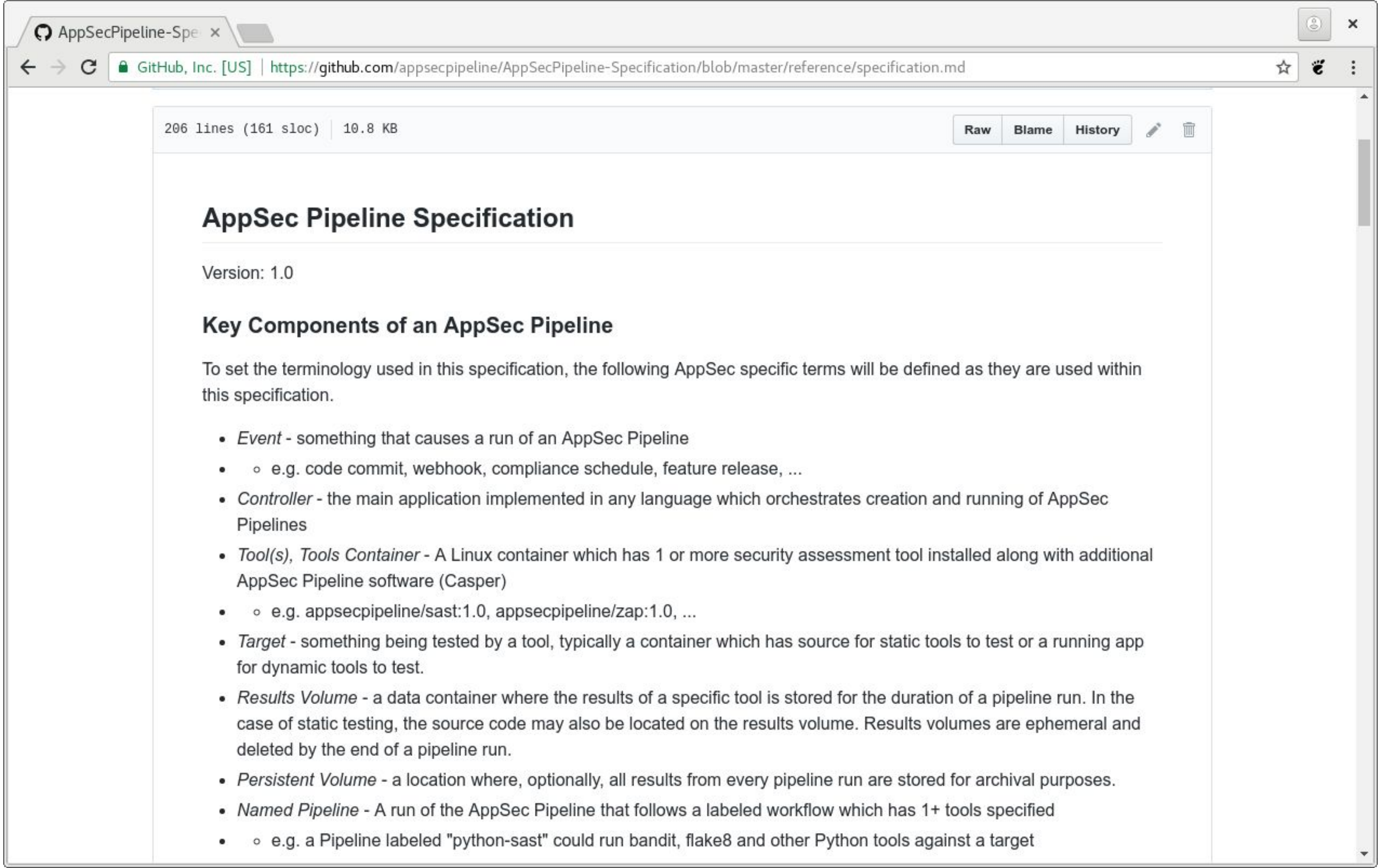
pyAppSecPipelineRestAPI

Top languages

● Python ● Go

People 0 ›

This organization has no public members. You must be a member to see who's a part of this organization.


AppSecPipeline-Spe ×
GitHub, Inc. [US] | https://github.com/appsecpipeline/AppSecPipeline-Specification/blob/master/reference/specification.md
206 lines (161 sloc) | 10.8 KB
Raw
Blame
History

AppSec Pipeline Specification

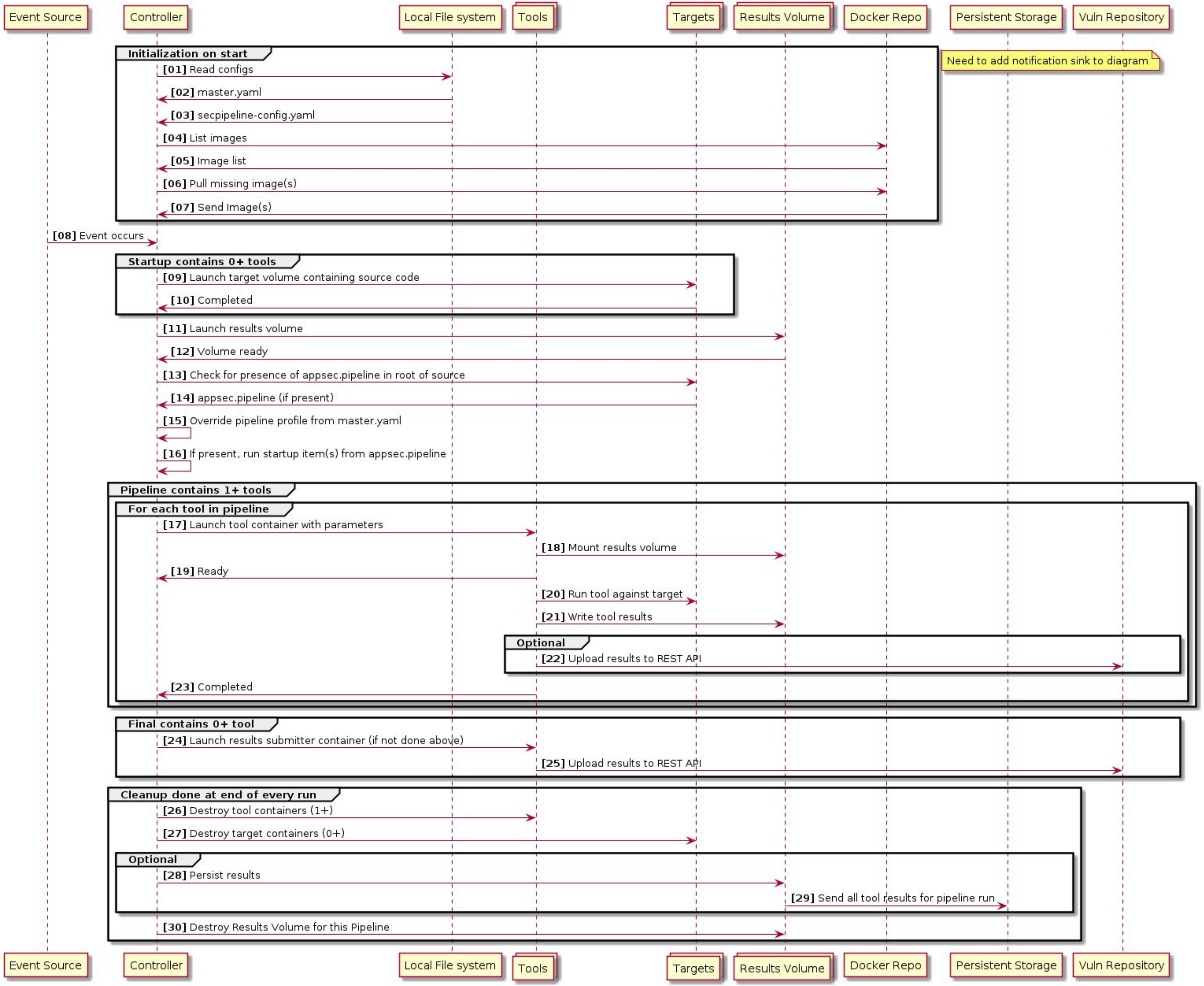
Version: 1.0

Key Components of an AppSec Pipeline

To set the terminology used in this specification, the following AppSec specific terms will be defined as they are used within this specification.

- *Event* - something that causes a run of an AppSec Pipeline
- - e.g. code commit, webhook, compliance schedule, feature release, ...
- *Controller* - the main application implemented in any language which orchestrates creation and running of AppSec Pipelines
- *Tool(s), Tools Container* - A Linux container which has 1 or more security assessment tool installed along with additional AppSec Pipeline software (Casper)
- - e.g. appsecpipeline/sast:1.0, appsecpipeline/zap:1.0, ...
- *Target* - something being tested by a tool, typically a container which has source for static tools to test or a running app for dynamic tools to test.
- *Results Volume* - a data container where the results of a specific tool is stored for the duration of a pipeline run. In the case of static testing, the source code may also be located on the results volume. Results volumes are ephemeral and deleted by the end of a pipeline run.
- *Persistent Volume* - a location where, optionally, all results from every pipeline run are stored for archival purposes.
- *Named Pipeline* - A run of the AppSec Pipeline that follows a labeled workflow which has 1+ tools specified
- - e.g. a Pipeline labeled "python-sast" could run bandit, flake8 and other Python tools against a target

AppSec Pipeline for SAST tools



Need to add notification sink to diagram

Initialization on start

[01] Read configs

[02] master.yaml

[03] secpipeline-config.yaml

[04] List images

[05] Image list

[06] Pull missing image(s)

[07] Send Image(s)

[08] Event occurs

Startup contains 0+ tools

[09] Launch target volume containing source code

[10] Completed

[11] Launch results volume

[12] Volume ready

[13] Check for presence of appsec.pipeline in root of source

[14] appsec.pipeline (if present)

[15] Override pipeline profile from master.yaml

[16] If present, run startup item(s) from appsec.pipeline

Pipeline contains 1+ tools

For each tool in pipeline

[17] Launch tool container with parameters

[18] Mount results volume

[19] Ready

[20] Run tool against target

[21] Write tool results

Optional

[22] Upload results to REST API

[23] Completed

Final contains 0+ tool

[24] Launch results submitter container (if not done above)

[25] Upload results to REST API

Cleanup done at end of every run

[26] Destroy tool containers (1+)

[27] Destroy target containers (0+)

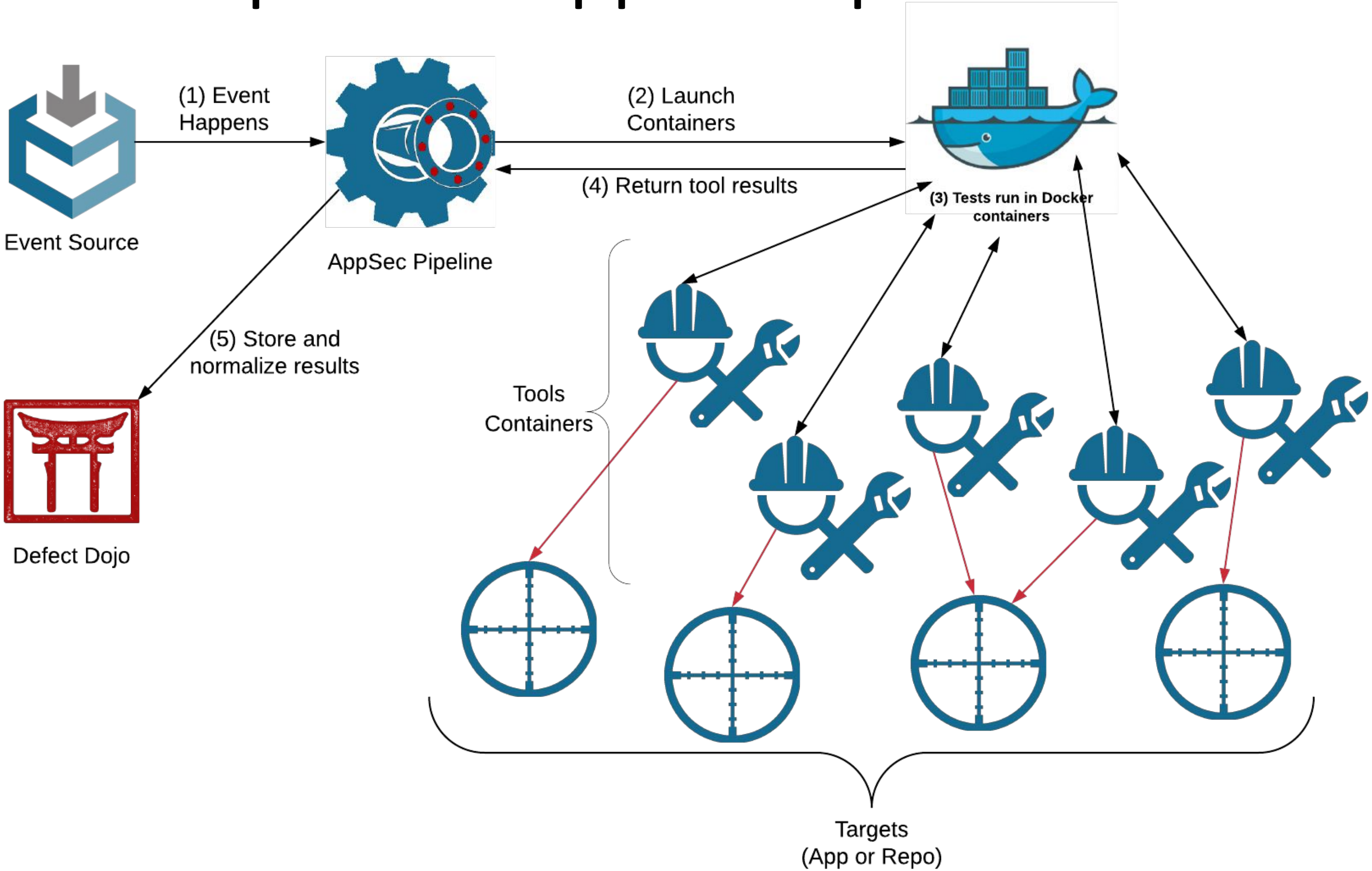
Optional

[28] Persist results

[29] Send all tool results for pipeline run

[30] Destroy Results Volume for this Pipeline

Steps in an AppSec Pipeline run

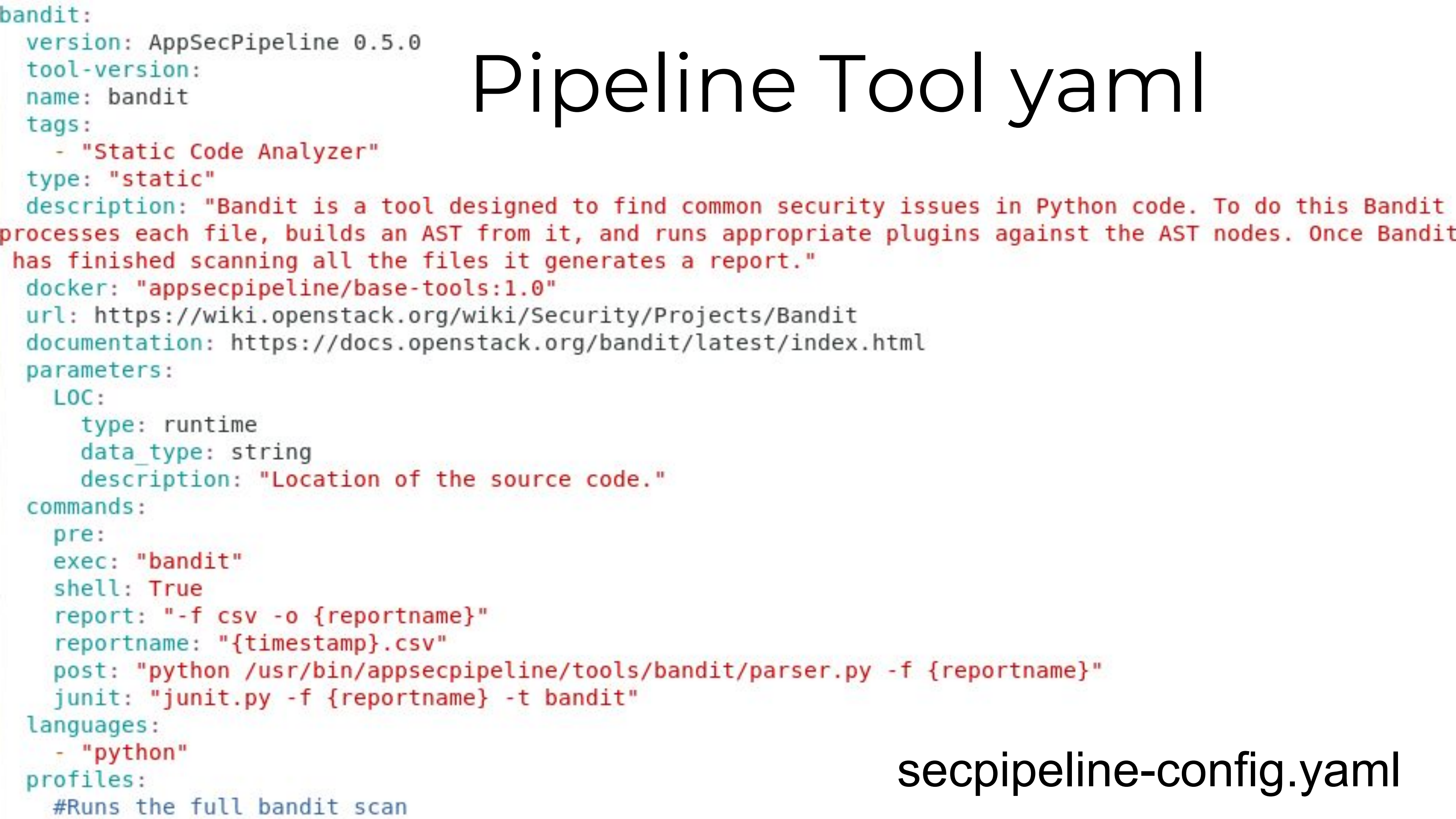


Making containers work for you

- Treat containers like a large binary **executable**
  - Execute **once**, then **discard**

- Each security tool or service is in a **container**
  - Each has a **configuration** file in yaml
  - Yaml contains pre-configured tool **profiles**

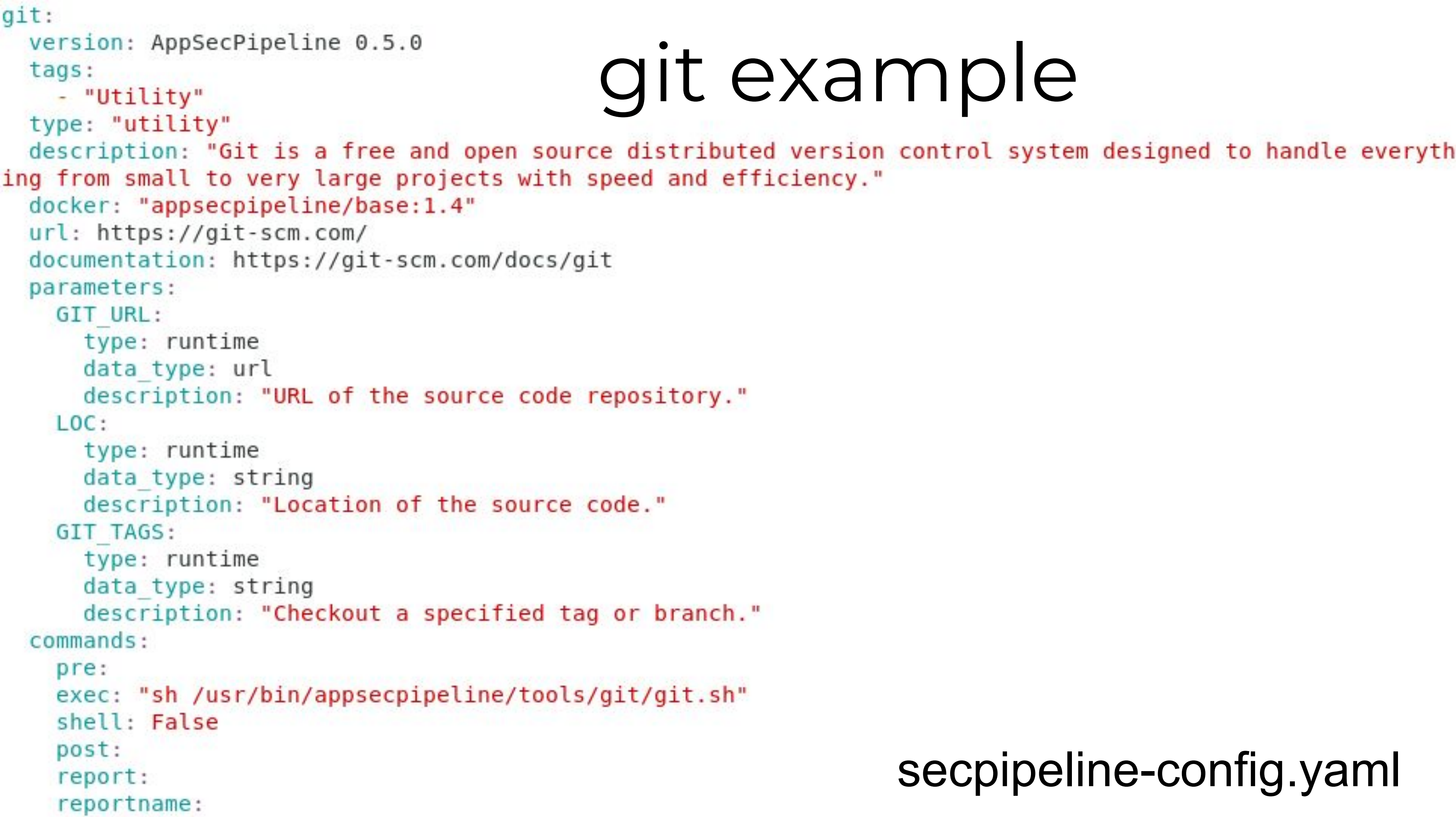
Pipeline Tool yaml



```yaml
bandit:
  version: AppSecPipeline 0.5.0
  tool-version:
  name: bandit
  tags:
    - "Static Code Analyzer"
  type: "static"
  description: "Bandit is a tool designed to find common security issues in Python code. To do this Bandit
processes each file, builds an AST from it, and runs appropriate plugins against the AST nodes. Once Bandit
 has finished scanning all the files it generates a report."
  docker: "appsecpipeline/base-tools:1.0"
  url: https://wiki.openstack.org/wiki/Security/Projects/Bandit
  documentation: https://docs.openstack.org/bandit/latest/index.html
  parameters:
    LOC:
      type: runtime
      data_type: string
      description: "Location of the source code."
  commands:
    pre:
    exec: "bandit"
    shell: True
    report: "-f csv -o {reportname}"
    reportname: "{timestamp}.csv"
    post: "python /usr/bin/appsecpipeline/tools/bandit/parser.py -f {reportname}"
    junit: "junit.py -f {reportname} -t bandit"
  languages:
    - "python"
  profiles:
    #Runs the full bandit scan
```

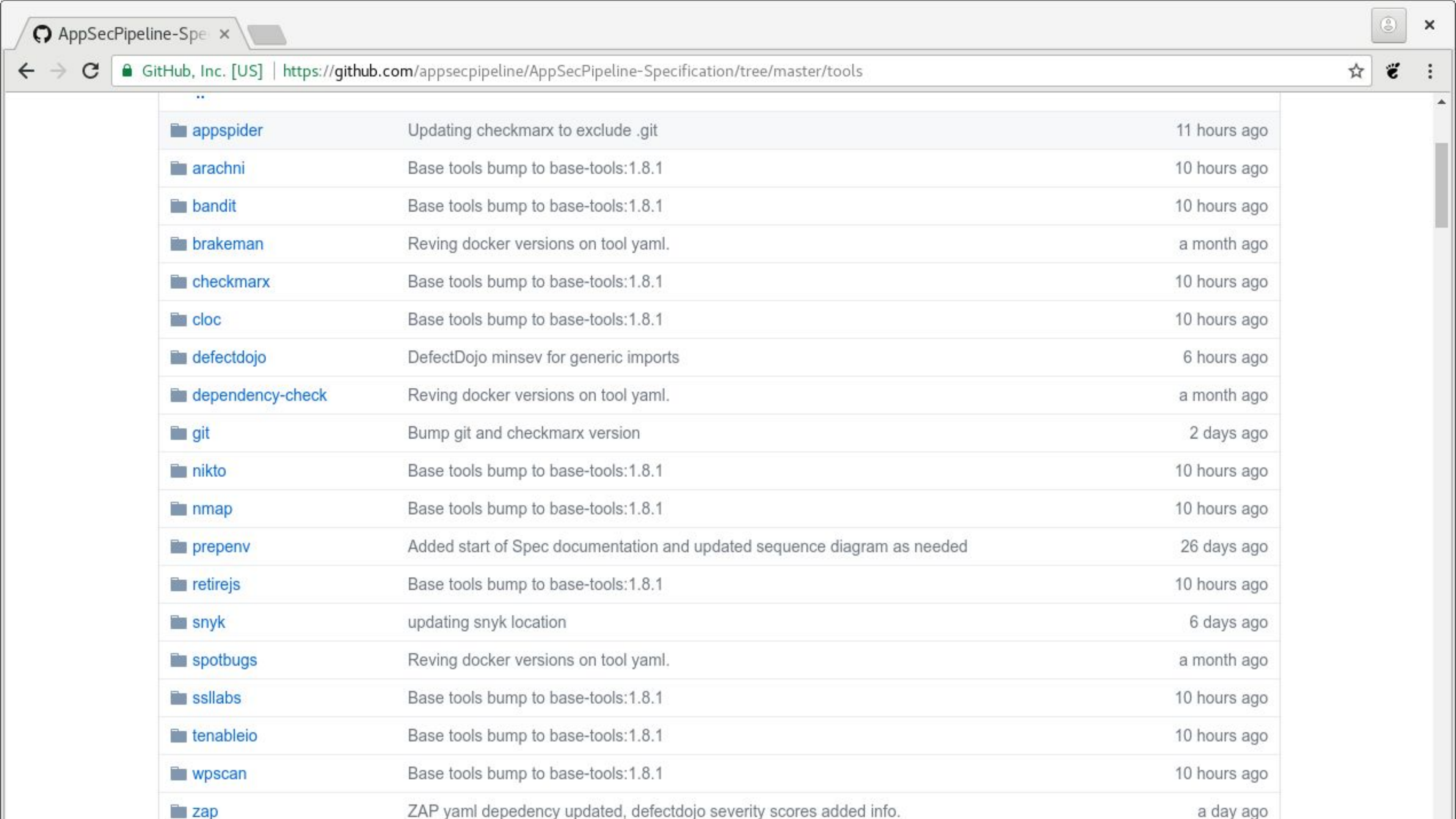
secpipeline-config.yaml

git example



```yaml
git:
  version: AppSecPipeline 0.5.0
  tags:
    - "Utility"
  type: "utility"
  description: "Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency."
  docker: "appsecpipeline/base:1.4"
  url: https://git-scm.com/
  documentation: https://git-scm.com/docs/git
  parameters:
    GIT_URL:
      type: runtime
      data_type: url
      description: "URL of the source code repository."
    LOC:
      type: runtime
      data_type: string
      description: "Location of the source code."
    GIT_TAGS:
      type: runtime
      data_type: string
      description: "Checkout a specified tag or branch."
  commands:
    pre:
    exec: "sh /usr/bin/appsecpipeline/tools/git/git.sh"
    shell: False
    post:
    report:
    reportname:
```

secpipeline-config.yaml



https://github.com/appsecpipeline/AppSecPipeline-Specification/tree/master/tools

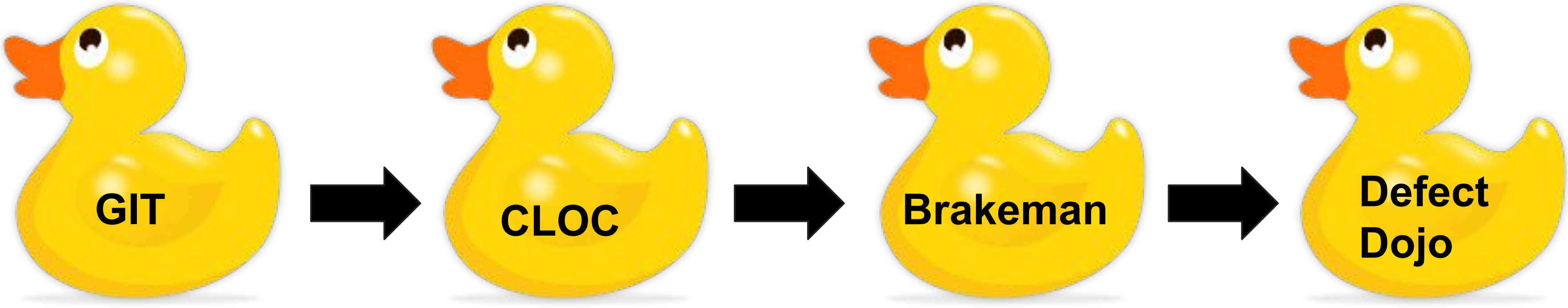
..		
appspider	Updating checkmarx to exclude .git	11 hours ago
arachni	Base tools bump to base-tools:1.8.1	10 hours ago
bandit	Base tools bump to base-tools:1.8.1	10 hours ago
brakeman	Reving docker versions on tool yaml.	a month ago
checkmarx	Base tools bump to base-tools:1.8.1	10 hours ago
cloc	Base tools bump to base-tools:1.8.1	10 hours ago
defectdojo	DefectDojo minsev for generic imports	6 hours ago
dependency-check	Reving docker versions on tool yaml.	a month ago
git	Bump git and checkmarx version	2 days ago
nikto	Base tools bump to base-tools:1.8.1	10 hours ago
nmap	Base tools bump to base-tools:1.8.1	10 hours ago
prepenv	Added start of Spec documentation and updated sequence diagram as needed	26 days ago
retirejs	Base tools bump to base-tools:1.8.1	10 hours ago
snyk	updating snyk location	6 days ago
spotbugs	Reving docker versions on tool yaml.	a month ago
ssllabs	Base tools bump to base-tools:1.8.1	10 hours ago
tenableio	Base tools bump to base-tools:1.8.1	10 hours ago
wpscan	Base tools bump to base-tools:1.8.1	10 hours ago
zap	ZAP yaml depedency updated, defectdojo severity scores added info.	a day ago

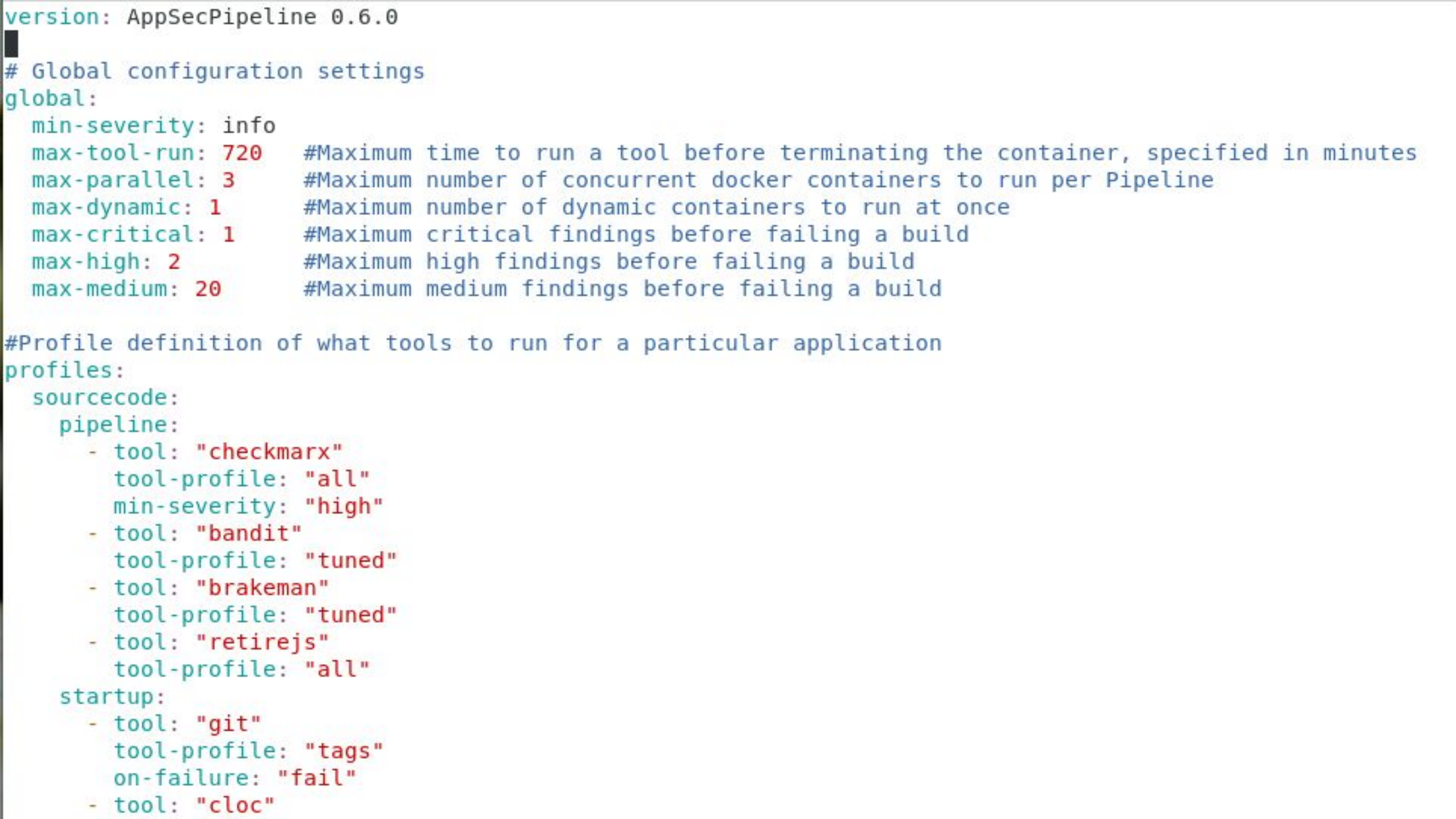
Benefits of Containerizing Tools

- Do a single ***“interesting”*** install once
- Figure out all the arcane tool options once
  - Sane defaults
  - Further refinement for high risk targets
- Tools can be in any language
- Establish a AppSec baseline
  - Run the same tool container + profile against all apps

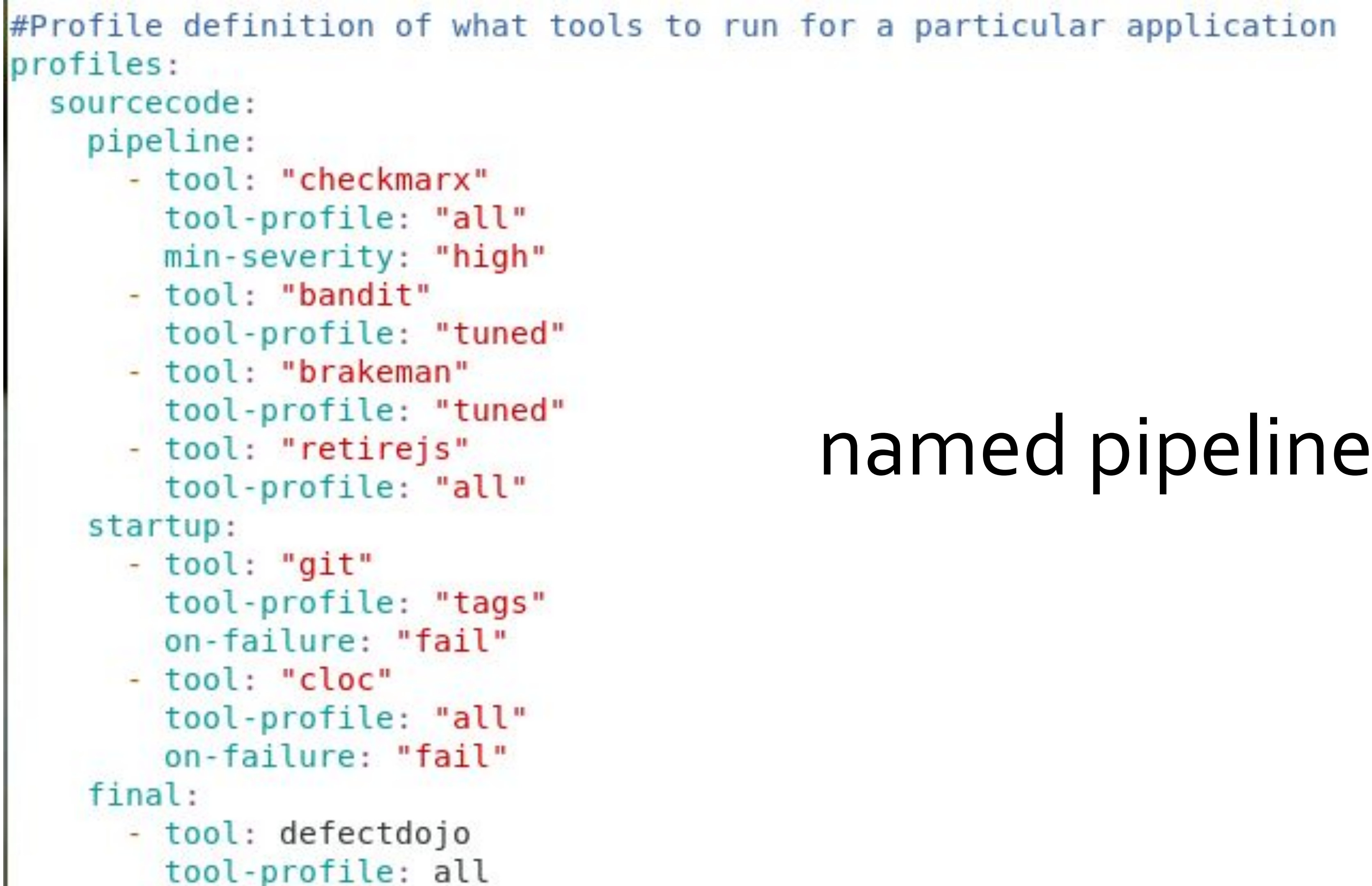
Named pipelines

- Tool configs + containers = pipeline tool
- Run multiple pipeline tools in a specific order to get a "Named pipeline"





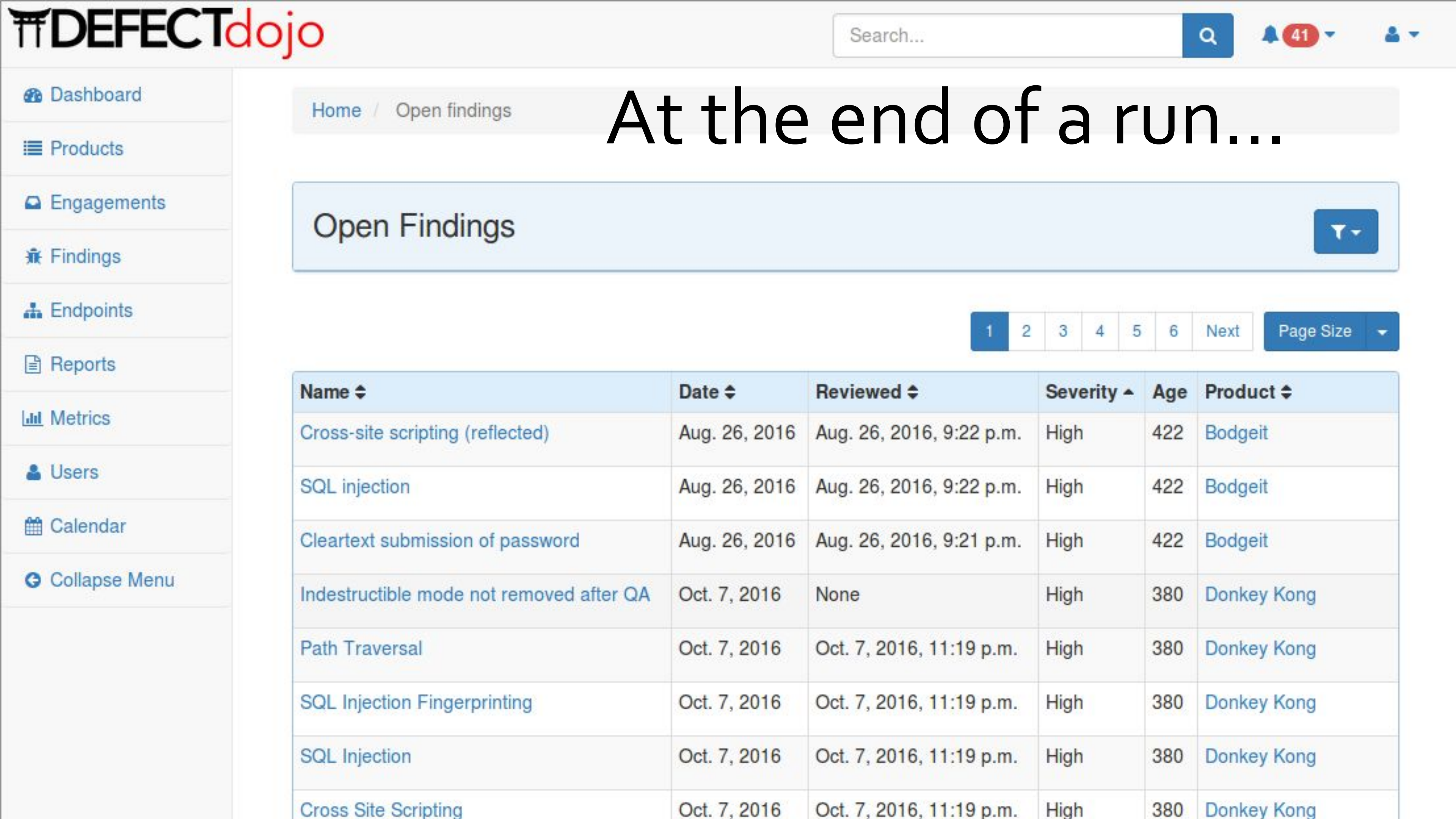
```yaml
version: AppSecPipeline 0.6.0

# Global configuration settings
global:
  min-severity: info
  max-tool-run: 720   #Maximum time to run a tool before terminating the container, specified in minutes
  max-parallel: 3     #Maximum number of concurrent docker containers to run per Pipeline
  max-dynamic: 1      #Maximum number of dynamic containers to run at once
  max-critical: 1     #Maximum critical findings before failing a build
  max-high: 2         #Maximum high findings before failing a build
  max-medium: 20      #Maximum medium findings before failing a build

#Profile definition of what tools to run for a particular application
profiles:
  sourcecode:
    pipeline:
      - tool: "checkmarx"
        tool-profile: "all"
        min-severity: "high"
      - tool: "bandit"
        tool-profile: "tuned"
      - tool: "brakeman"
        tool-profile: "tuned"
      - tool: "retirejs"
        tool-profile: "all"
    startup:
      - tool: "git"
        tool-profile: "tags"
        on-failure: "fail"
      - tool: "cloc"
```



```yaml
#Profile definition of what tools to run for a particular application
profiles:
  sourcecode:
    pipeline:
      - tool: "checkmarx"
        tool-profile: "all"
        min-severity: "high"
      - tool: "bandit"
        tool-profile: "tuned"
      - tool: "brakeman"
        tool-profile: "tuned"
      - tool: "retirejs"
        tool-profile: "all"
    startup:
      - tool: "git"
        tool-profile: "tags"
        on-failure: "fail"
      - tool: "cloc"
        tool-profile: "all"
        on-failure: "fail"
    final:
      - tool: defectdojo
        tool-profile: all
```

named pipeline

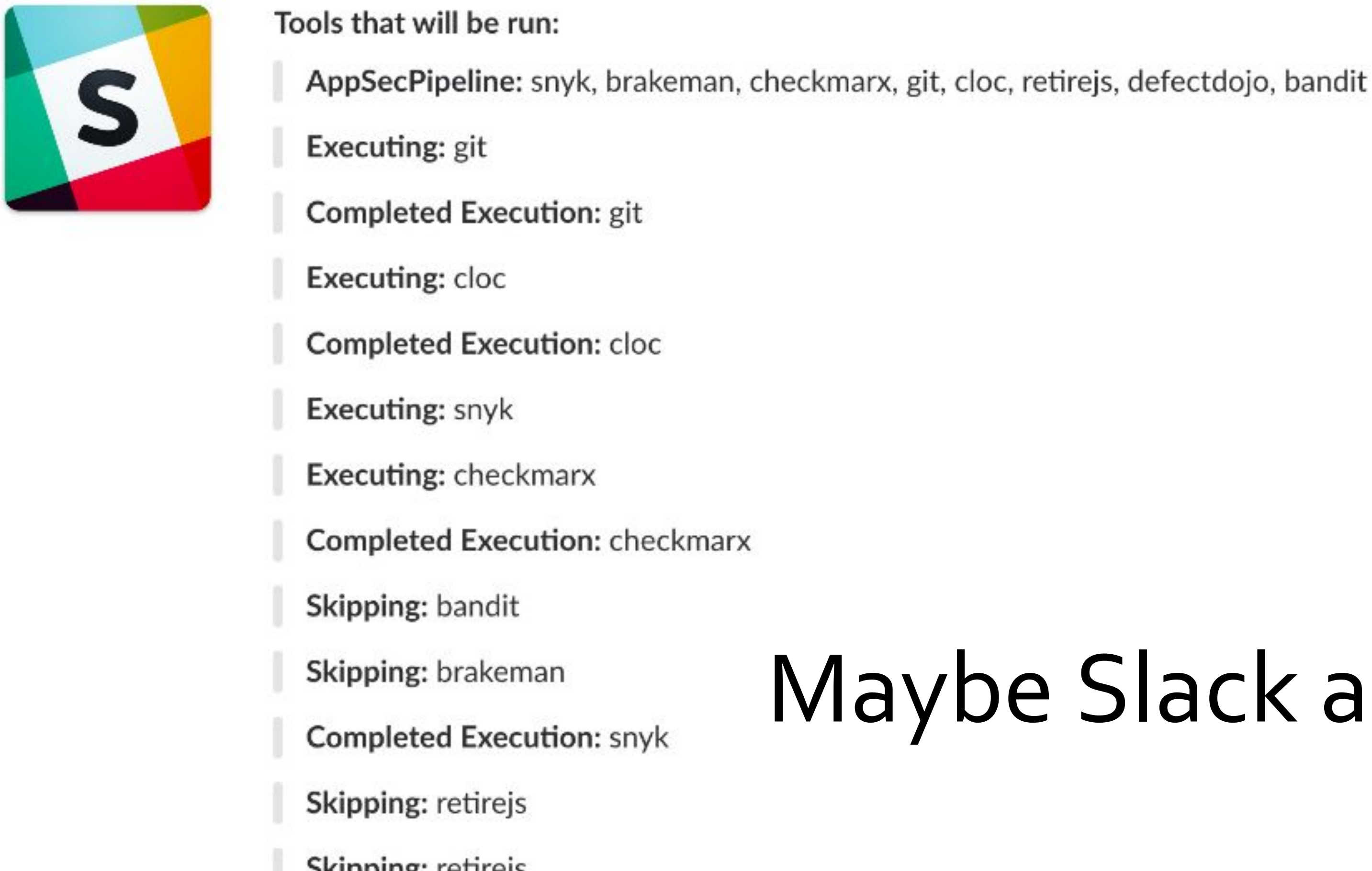
At the end of a run…



Open Findings

Name	Date	Reviewed	Severity	Age	Product
Cross-site scripting (reflected)	Aug. 26, 2016	Aug. 26, 2016, 9:22 p.m.	High	422	Bodgeit
SQL injection	Aug. 26, 2016	Aug. 26, 2016, 9:22 p.m.	High	422	Bodgeit
Cleartext submission of password	Aug. 26, 2016	Aug. 26, 2016, 9:21 p.m.	High	422	Bodgeit
Indestructible mode not removed after QA	Oct. 7, 2016	None	High	380	Donkey Kong
Path Traversal	Oct. 7, 2016	Oct. 7, 2016, 11:19 p.m.	High	380	Donkey Kong
SQL Injection Fingerprinting	Oct. 7, 2016	Oct. 7, 2016, 11:19 p.m.	High	380	Donkey Kong
SQL Injection	Oct. 7, 2016	Oct. 7, 2016, 11:19 p.m.	High	380	Donkey Kong
Cross Site Scripting	Oct. 7, 2016	Oct. 7, 2016, 11:19 p.m.	High	380	Donkey Kong

AppSecPipeline Run



Tools that will be run:

AppSecPipeline: snyk, brakeman, checkmarx, git, cloc, retirejs, defectdojo, bandit

Executing: git

Completed Execution: git

Executing: cloc

Completed Execution: cloc

Executing: snyk

Executing: checkmarx

Completed Execution: checkmarx

Skipping: bandit

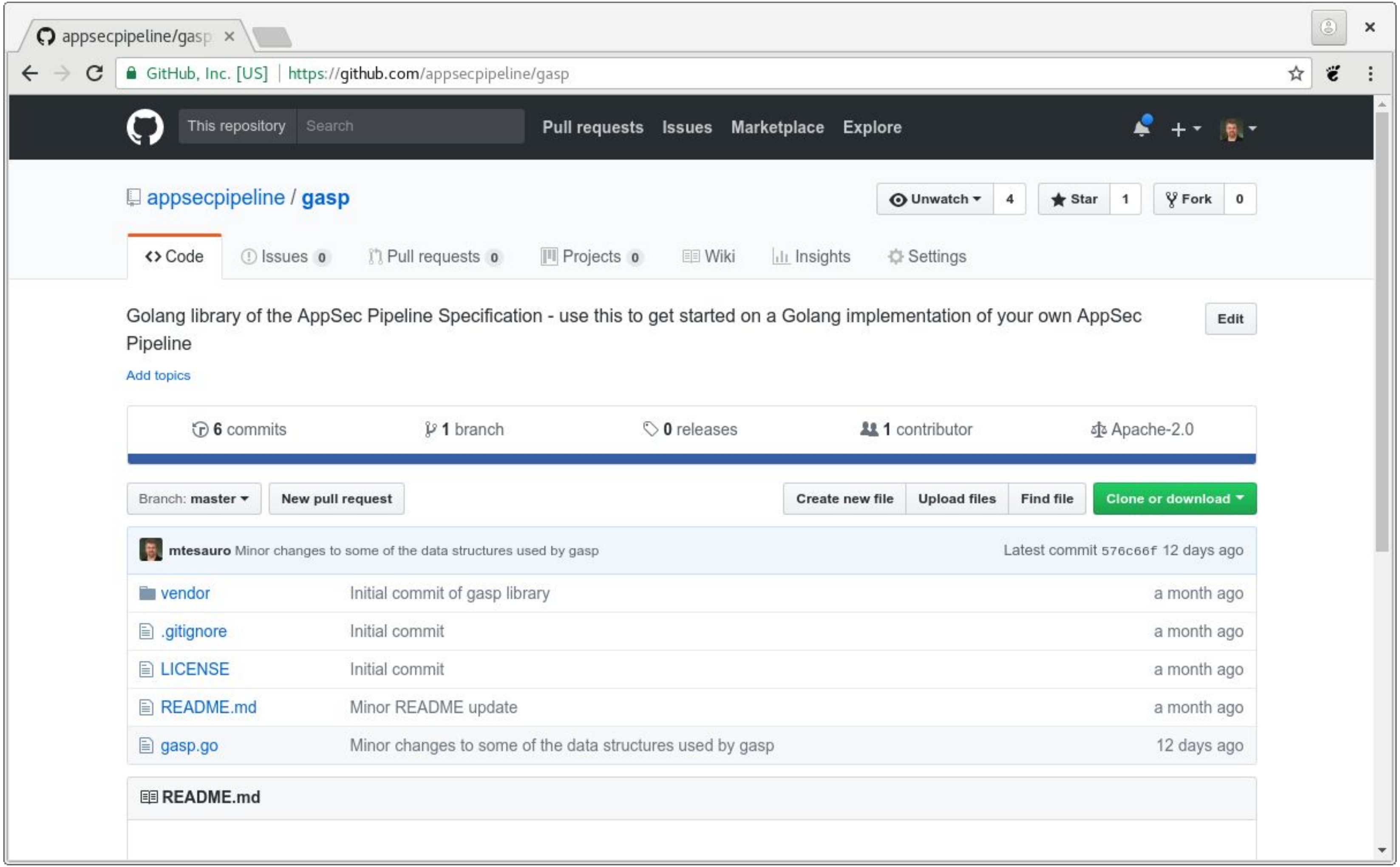
Skipping: brakeman

Completed Execution: snyk

Skipping: retirejs

Skipping: retirejs

Maybe Slack alerts

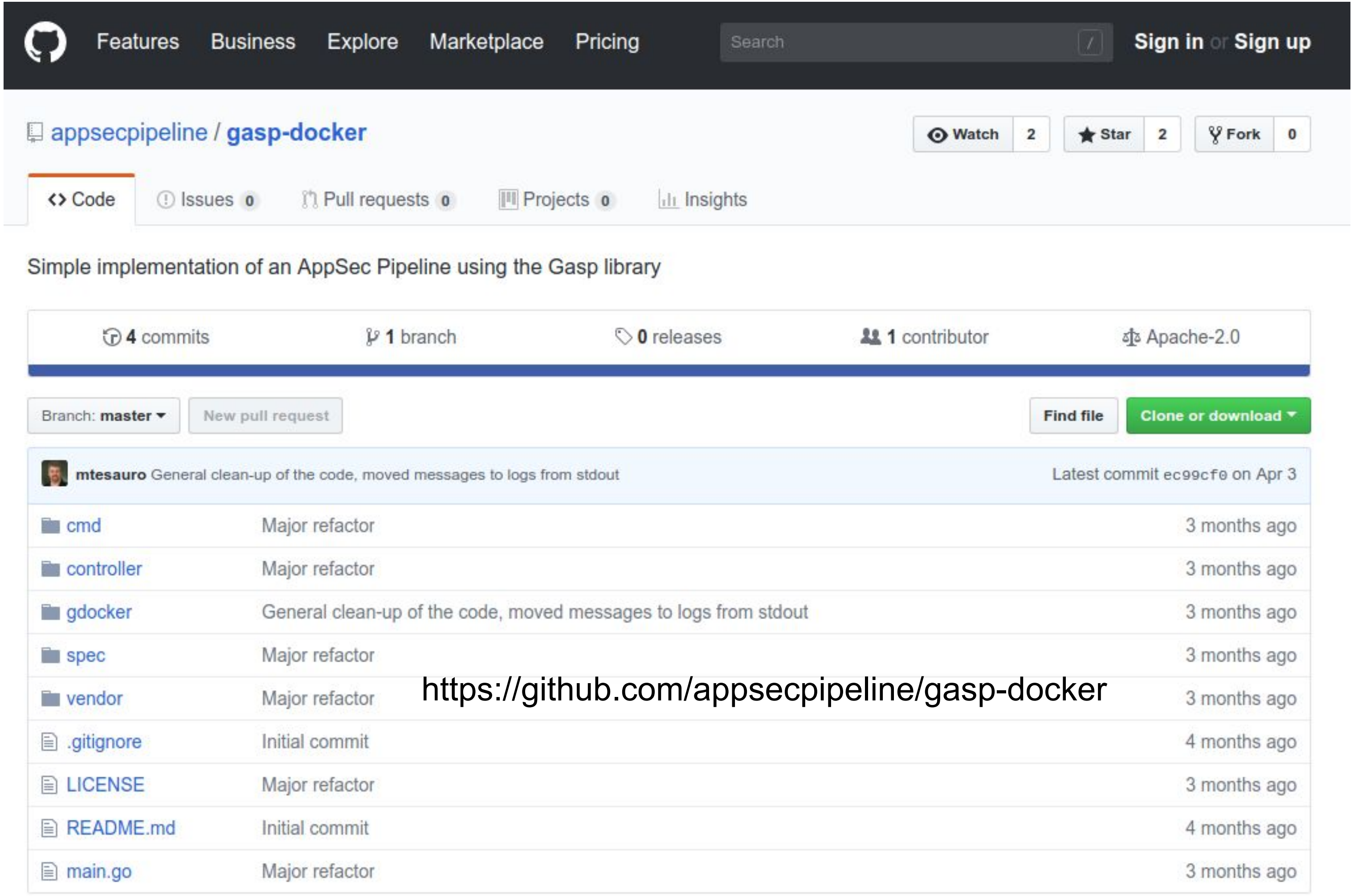


Golang library of the AppSec Pipeline Specification - use this to get started on a Golang implementation of your own AppSec Pipeline

mtesauro Minor changes to some of the data structures used by gasp Latest commit 576c66f 12 days ago

vendor	Initial commit of gasp library	a month ago
.gitignore	Initial commit	a month ago
LICENSE	Initial commit	a month ago
README.md	Minor README update	a month ago
gasp.go	Minor changes to some of the data structures used by gasp	12 days ago

README.md



appsecpipeline / gasp-docker

Simple implementation of an AppSec Pipeline using the Gasp library

4 commits | 1 branch | 0 releases | 1 contributor | Apache-2.0

mtesauro General clean-up of the code, moved messages to logs from stdout — Latest commit `ec99cf0` on Apr 3

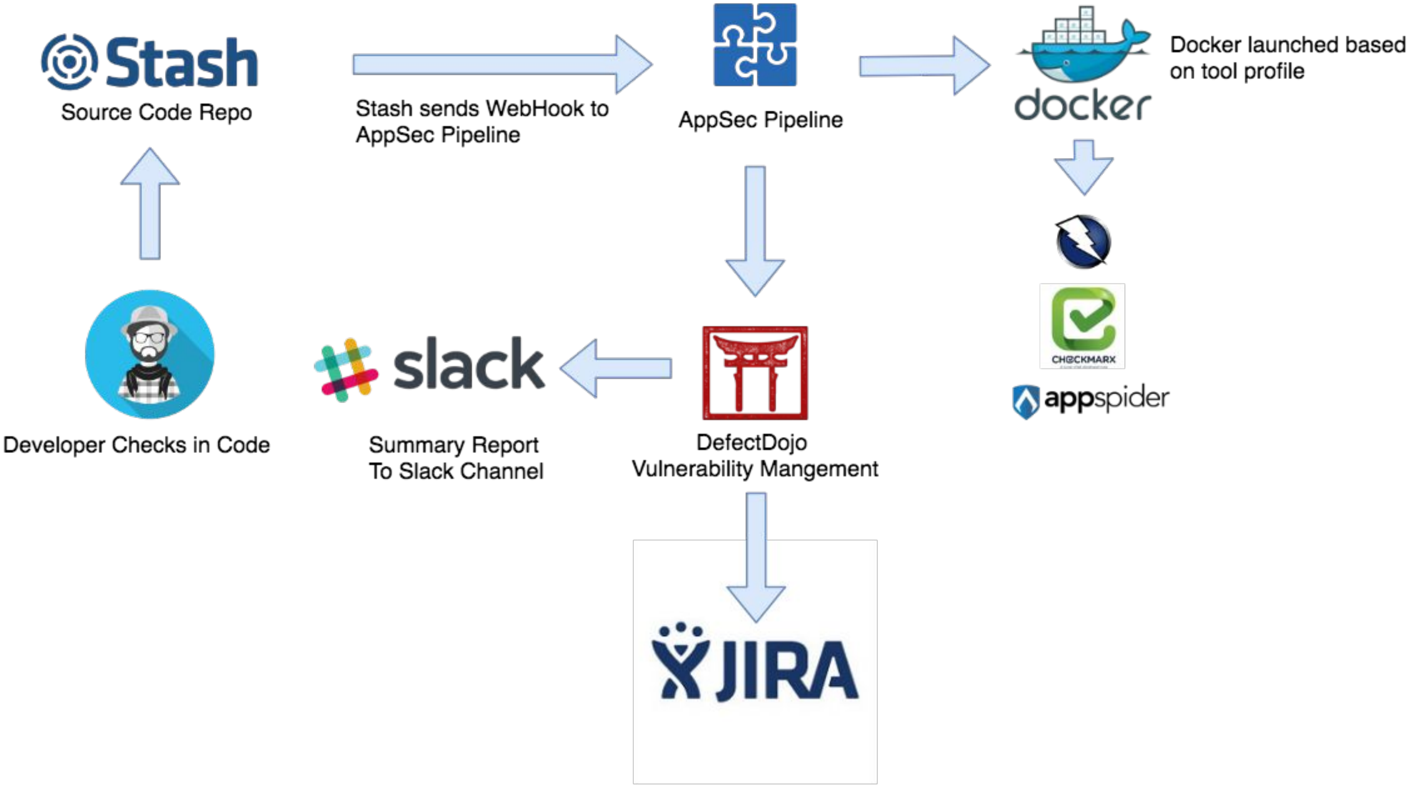
cmd	Major refactor	3 months ago
controller	Major refactor	3 months ago
gdocker	General clean-up of the code, moved messages to logs from stdout	3 months ago
spec	Major refactor	3 months ago
vendor	Major refactor	3 months ago
.gitignore	Initial commit	4 months ago
LICENSE	Major refactor	3 months ago
README.md	Initial commit	4 months ago
main.go	Major refactor	3 months ago

https://github.com/appsecpipeline/gasp-docker

AppSec Pipeline

A real life example of an implemented AppSec Pipeline

My Curent AppSec Pipeline



Source Code Repo

Stash sends WebHook to
AppSec Pipeline

AppSec Pipeline

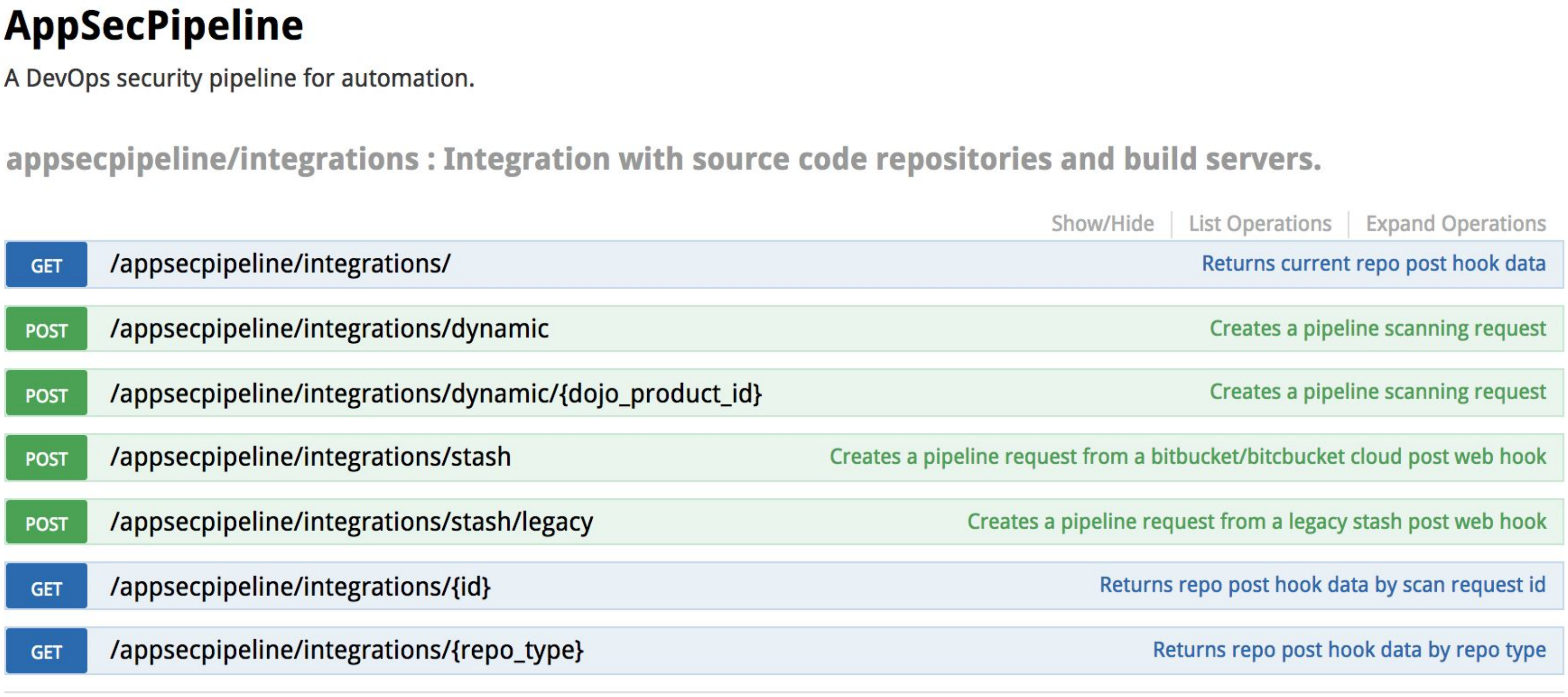
Docker launched based
on tool profile

Developer Checks in Code

Summary Report
To Slack Channel

DefectDojo
Vulnerability Mangement

Lightweight Rest API’s



AppSecPipeline

A DevOps security pipeline for automation.

appsecpipeline/integrations : Integration with source code repositories and build servers.

GET	/appsecpipeline/integrations/	Returns current repo post hook data
POST	/appsecpipeline/integrations/dynamic	Creates a pipeline scanning request
POST	/appsecpipeline/integrations/dynamic/{dojo_product_id}	Creates a pipeline scanning request
POST	/appsecpipeline/integrations/stash	Creates a pipeline request from a bitbucket/bitcbucket cloud post web hook
POST	/appsecpipeline/integrations/stash/legacy	Creates a pipeline request from a legacy stash post web hook
GET	/appsecpipeline/integrations/{id}	Returns repo post hook data by scan request id
GET	/appsecpipeline/integrations/{repo_type}	Returns repo post hook data by repo type

t2.large EC2 Instance

Criteria for Tools

- Runs fairly quickly
- Fast, lightweight dynamic scans
- Static scans with differential
- Third Party Components

AppSec Pipeline Stats

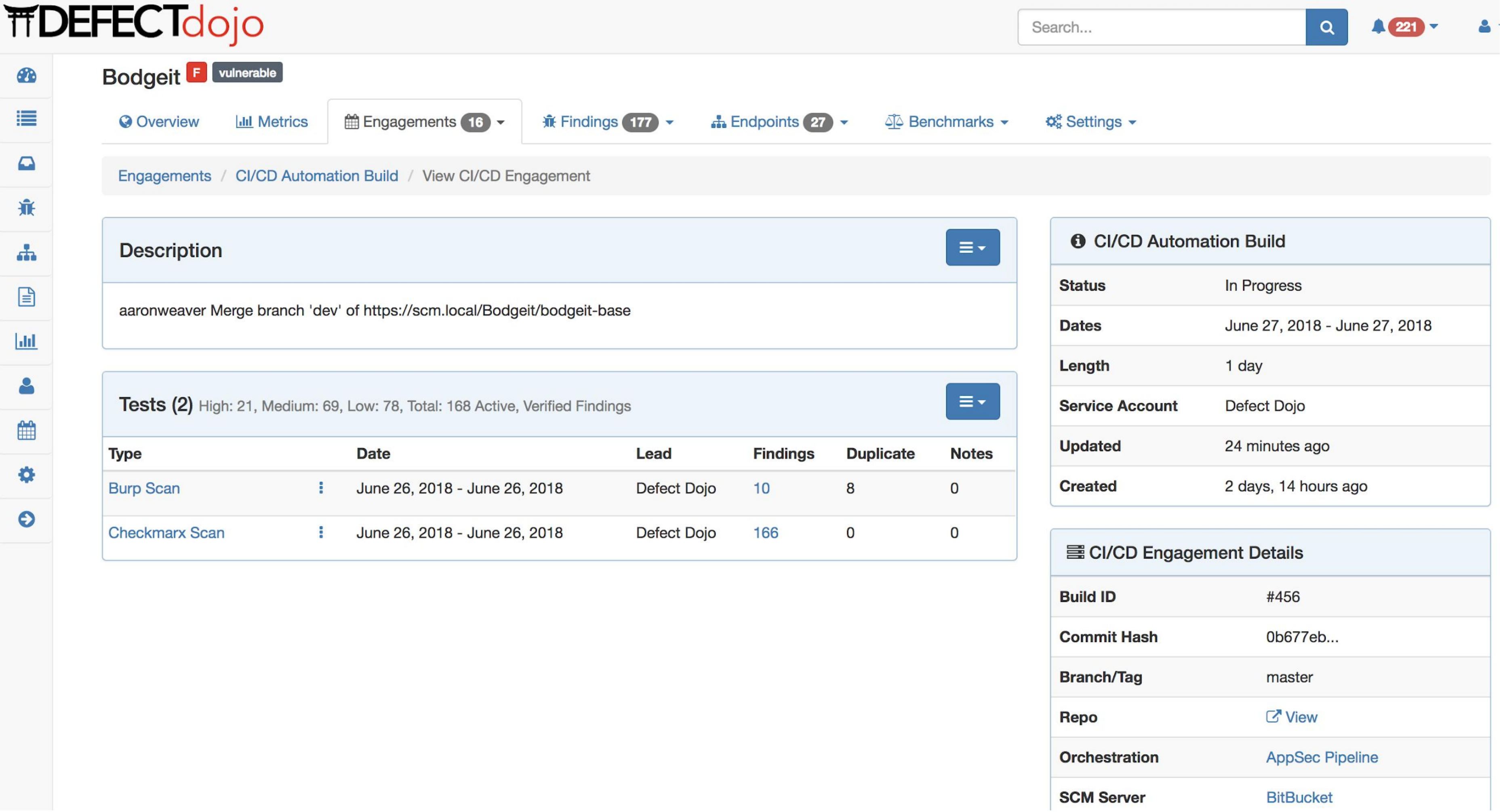
15 Repos

5,100 Runs

4 Months

25,000+

Container Executions



Description

aaronweaver Merge branch 'dev' of https://scm.local/Bodgeit/bodgeit-base

Tests (2) High: 21, Medium: 69, Low: 78, Total: 168 Active, Verified Findings

Type	Date	Lead	Findings	Duplicate	Notes
Burp Scan	June 26, 2018 - June 26, 2018	Defect Dojo	10	8	0
Checkmarx Scan	June 26, 2018 - June 26, 2018	Defect Dojo	166	0	0

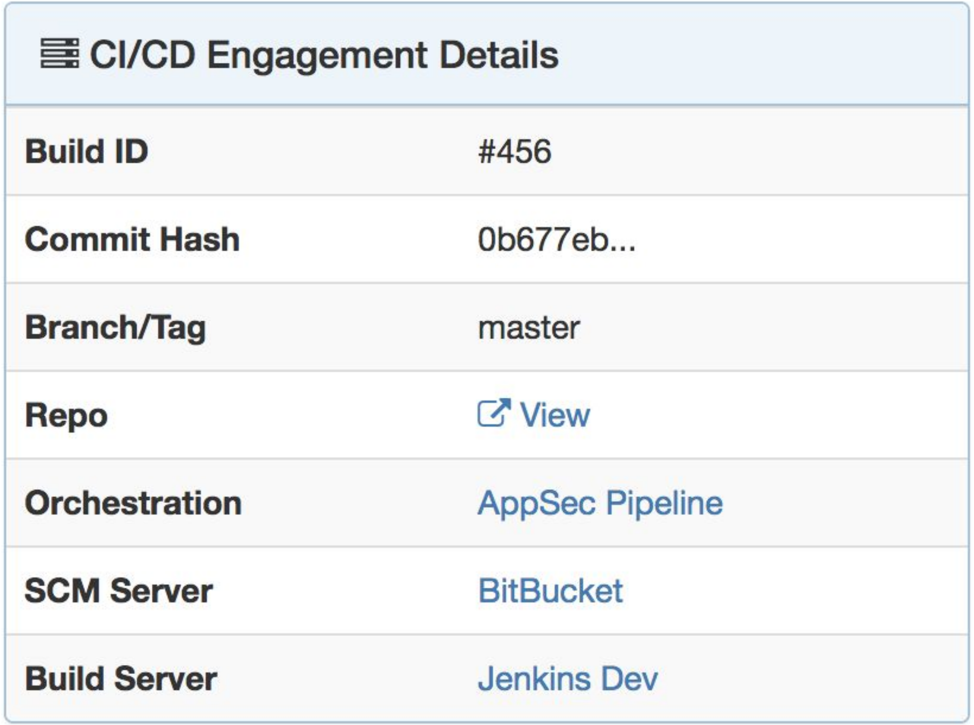
CI/CD Automation Build

Status	In Progress
Dates	June 27, 2018 - June 27, 2018
Length	1 day
Service Account	Defect Dojo
Updated	24 minutes ago
Created	2 days, 14 hours ago

CI/CD Engagement Details

Build ID	#456
Commit Hash	0b677eb...
Branch/Tag	master
Repo	View
Orchestration	AppSec Pipeline
SCM Server	BitBucket

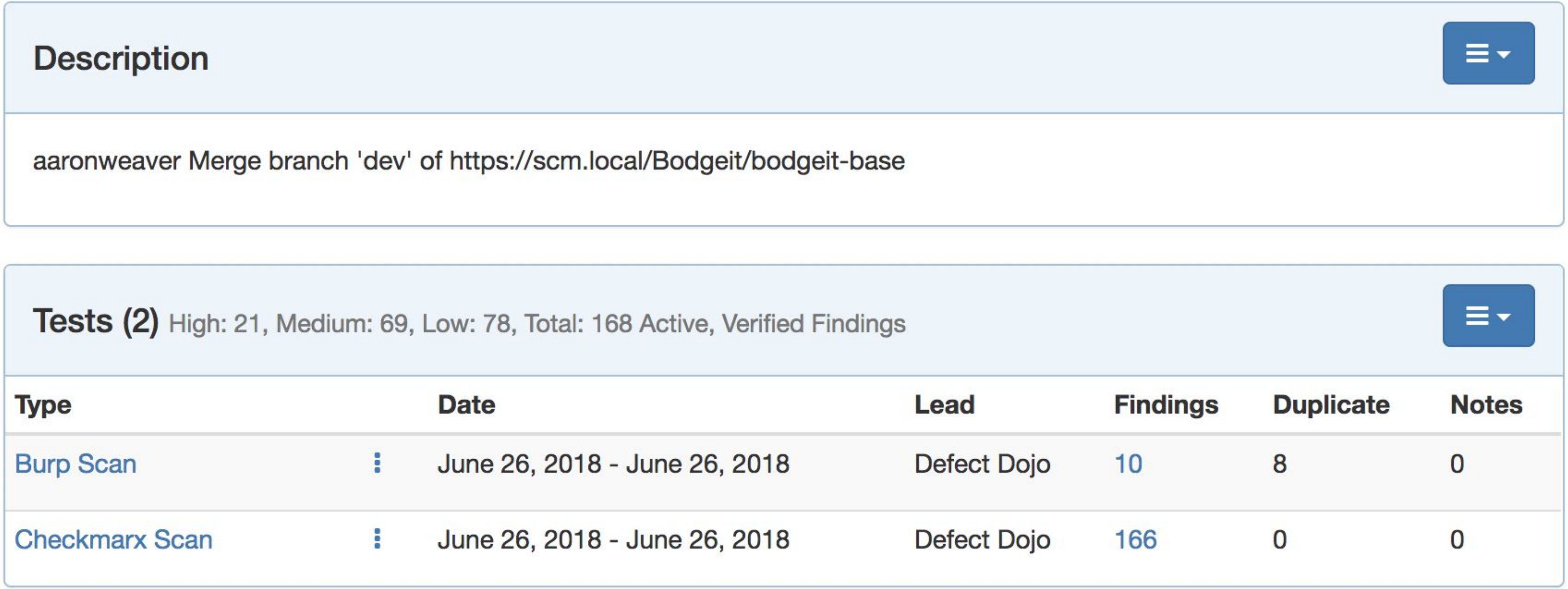
CI/CD Information



CI/CD Engagement Details

Build ID	#456
Commit Hash	0b677eb...
Branch/Tag	master
Repo	View
Orchestration	AppSec Pipeline
SCM Server	BitBucket
Build Server	Jenkins Dev

CI/CD Security Test



Description

aaronweaver Merge branch 'dev' of https://scm.local/Bodgeit/bodgeit-base

Tests (2) High: 21, Medium: 69, Low: 78, Total: 168 Active, Verified Findings

Type	Date	Lead	Findings	Duplicate	Notes
Burp Scan	June 26, 2018 - June 26, 2018	Defect Dojo	10	8	0
Checkmarx Scan	June 26, 2018 - June 26, 2018	Defect Dojo	166	0	0

What have I learned?

After the **first** run of scans the net new vulnerabilities are **low**.

Legacy security* tools will be your biggest pain point.

(Anything that isn’t in a container)

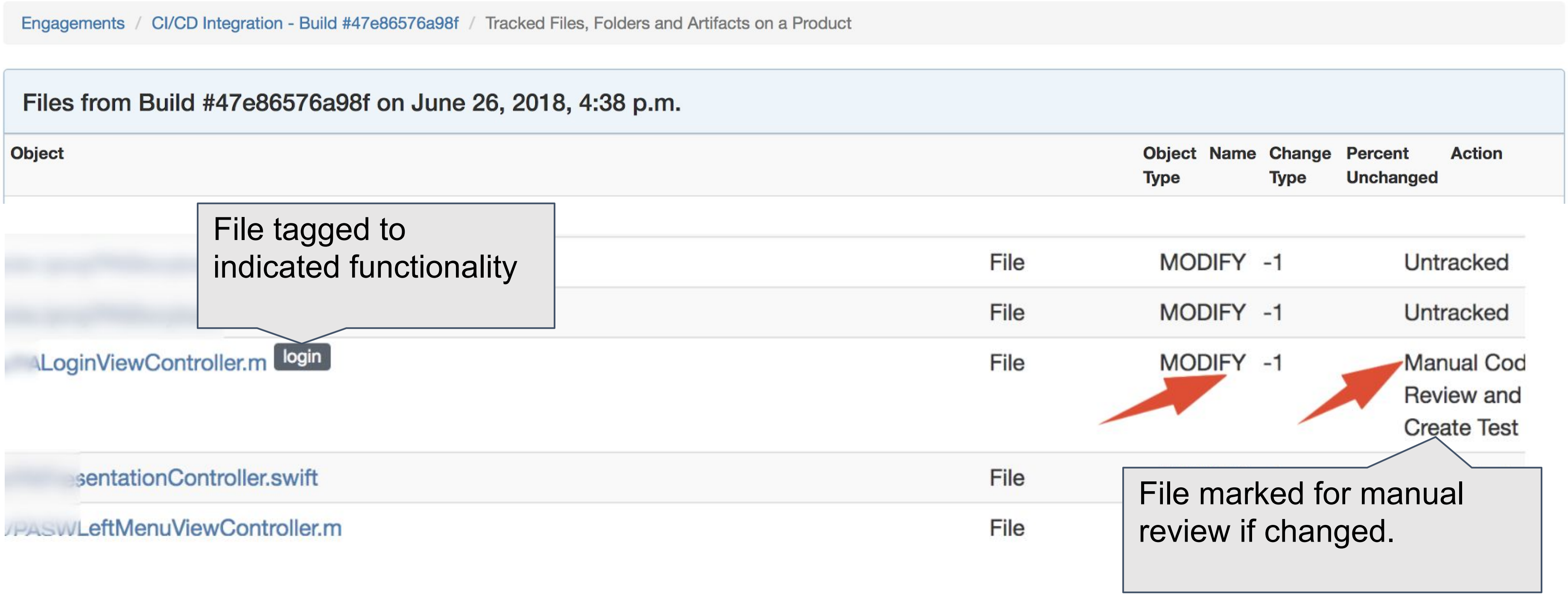
Evaluate what you did and look for the next improvement.

Improvement Idea

SCM Integration: The web post tells me what files have changed.

Manual Review

1. File Tagged for review from build



Engagements / CI/CD Integration - Build #47e86576a98f / Tracked Files, Folders and Artifacts on a Product

Files from Build #47e86576a98f on June 26, 2018, 4:38 p.m.

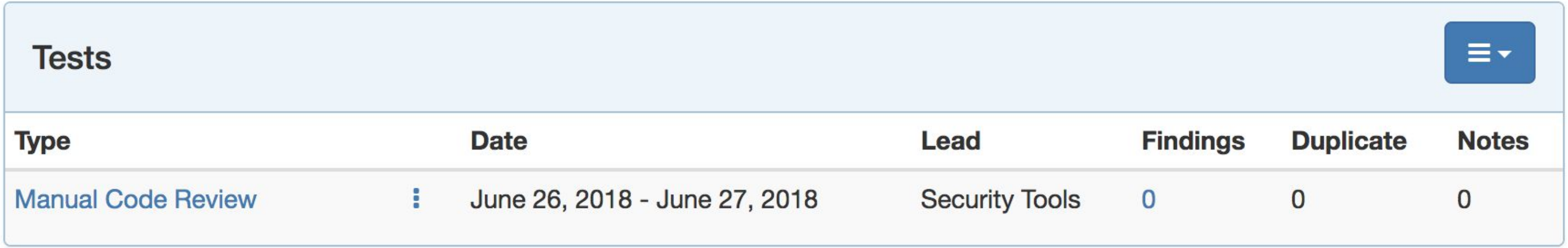
Object	Object Type	Name	Change Type	Percent Unchanged	Action
[redacted]	File		MODIFY	-1	Untracked
[redacted]	File		MODIFY	-1	Untracked
[redacted]LoginViewController.m login	File		MODIFY	-1	Manual Cod Review and Create Test
[redacted]sentationController.swift	File				
[redacted]LeftMenuViewController.m	File				

File tagged to indicated functionality

File marked for manual review if changed.

Manual Review

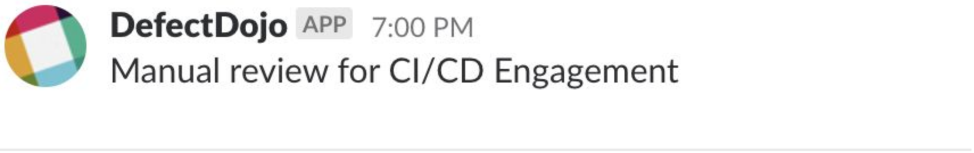
2. Manual Test Created for that Engagement



Tests

Type	Date	Lead	Findings	Duplicate	Notes
Manual Code Review ⋮	June 26, 2018 - June 27, 2018	Security Tools	0	0	0

3. Slack Alert

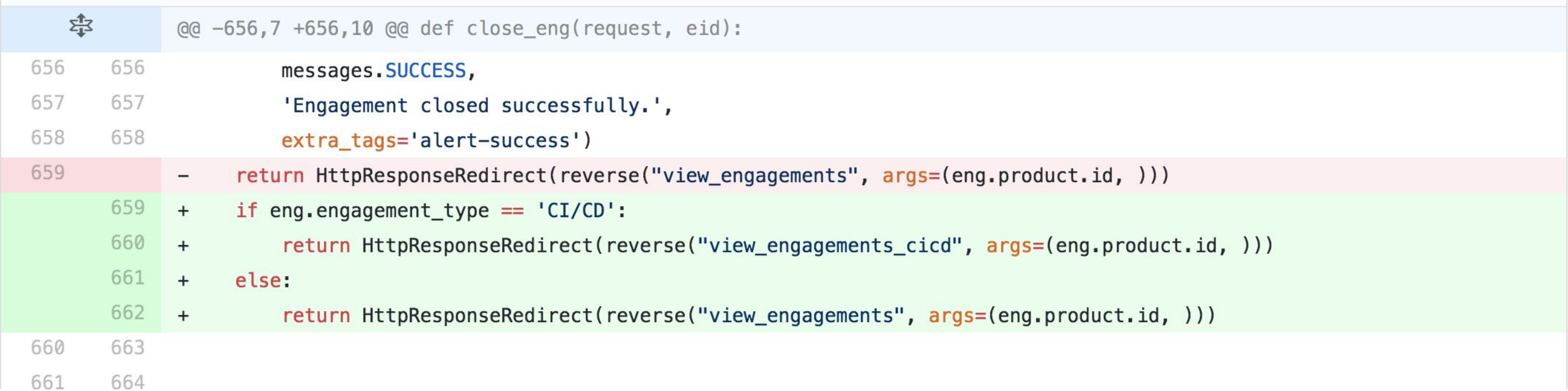


DefectDojo APP 7:00 PM

Manual review for CI/CD Engagement

Manual Review

4. Review changes in SCM

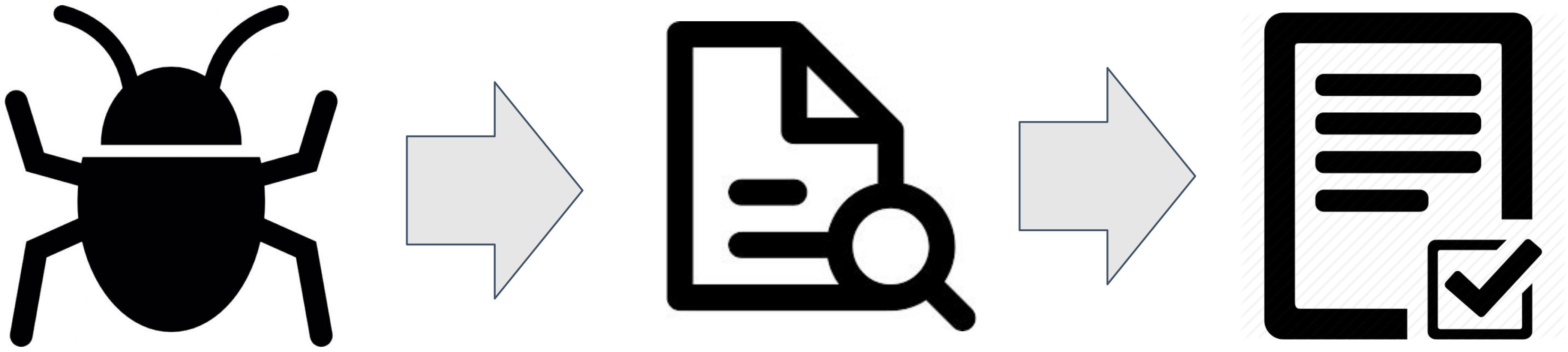


```
@@ -656,7 +656,10 @@ def close_eng(request, eid):
              messages.SUCCESS,
              'Engagement closed successfully.',
              extra_tags='alert-success')
-     return HttpResponseRedirect(reverse("view_engagements", args=(eng.product.id, )))
+     if eng.engagement_type == 'CI/CD':
+         return HttpResponseRedirect(reverse("view_engagements_cicd", args=(eng.product.id, )))
+     else:
+         return HttpResponseRedirect(reverse("view_engagements", args=(eng.product.id, )))
```

False positives:

Can we do better?

Rules Engine



Finding Imported

Analyze

Apply

Rules Engine

CWE Use Case

Title match on XSS →
Update CWE-79

Rules Engine

Scanner Matching

Scanner == SSLLabs →

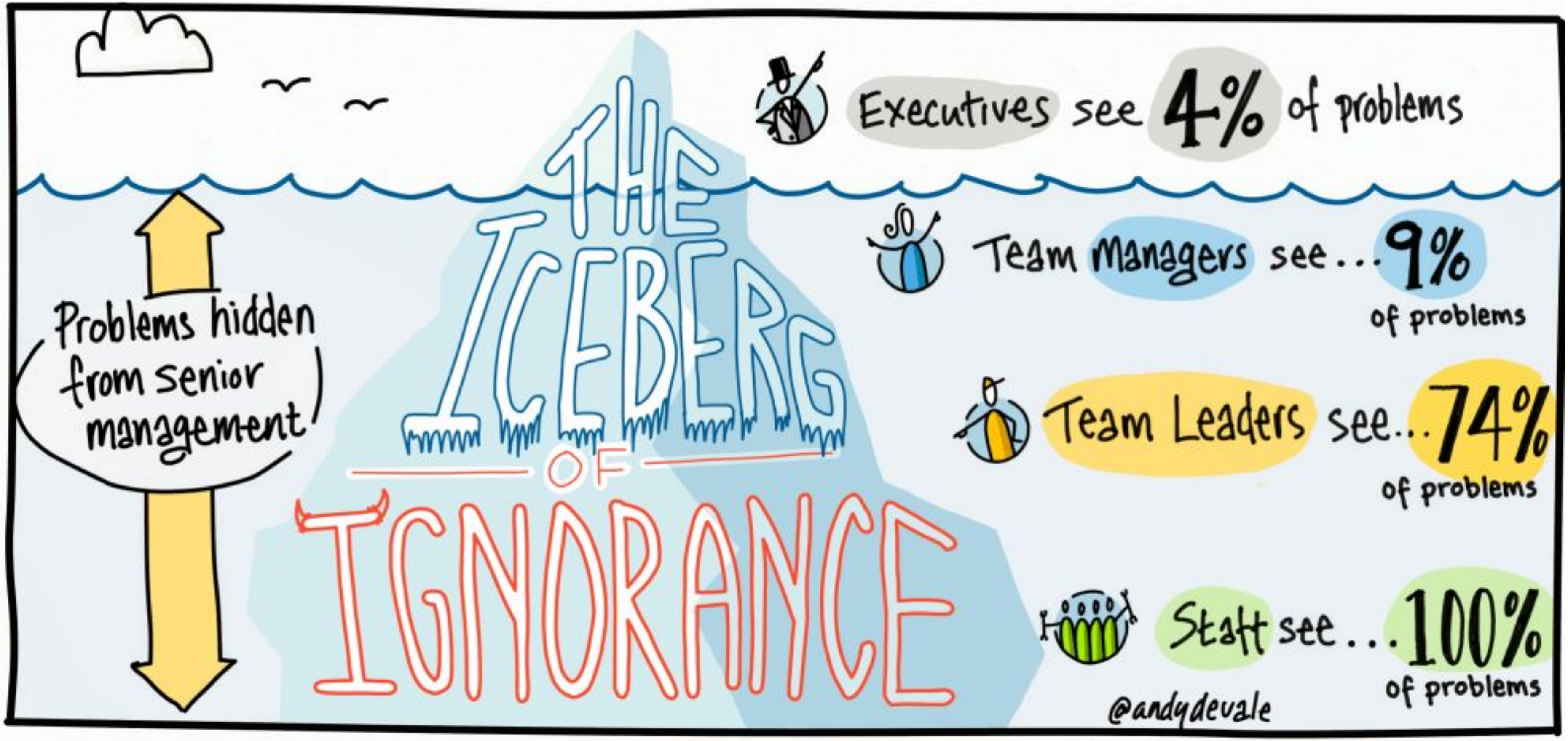
Grade < A →

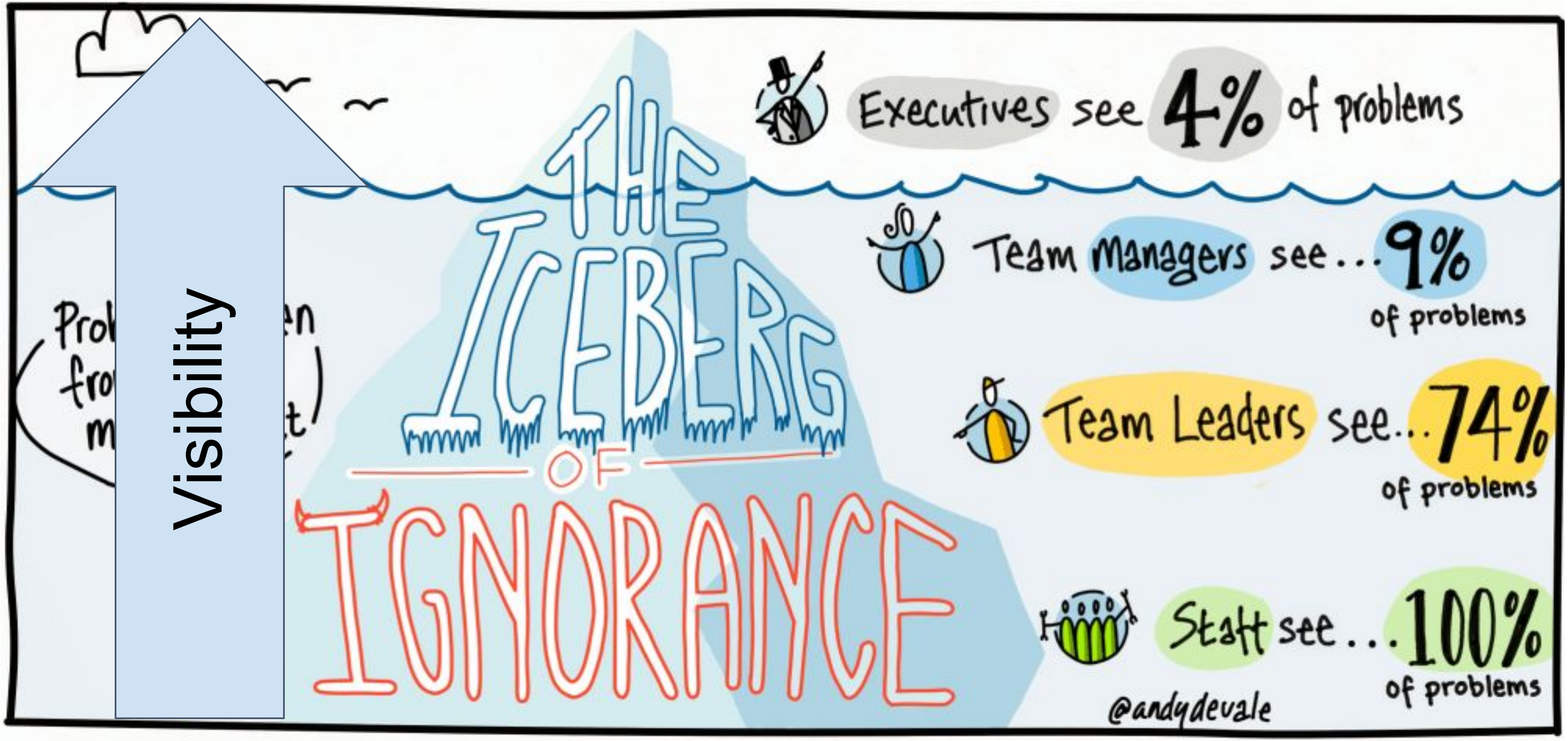
Update Verified

Rules Engine

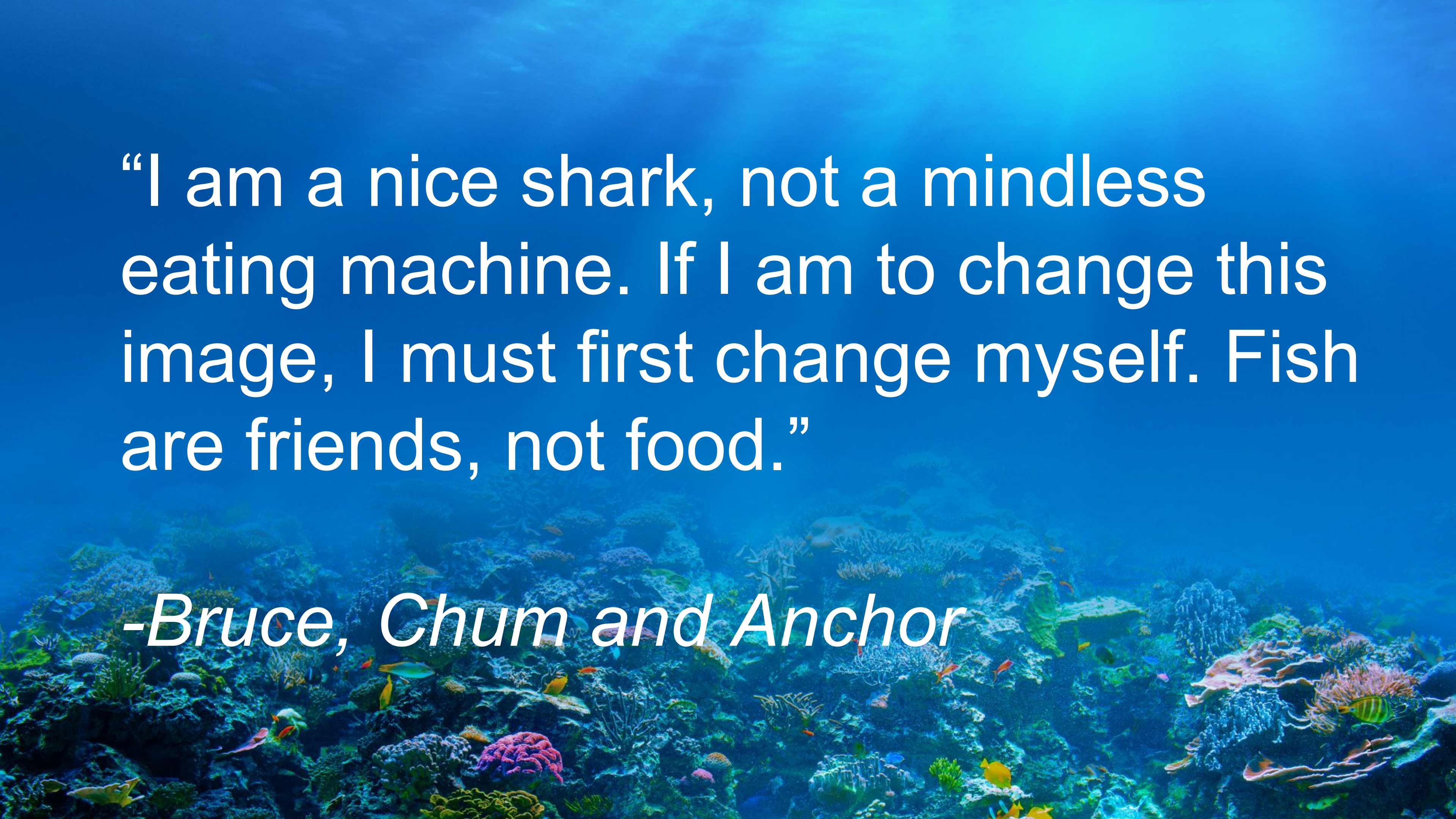
Scanner Confidence

Scanner Confidence == Confirmed → Title == XSS → Update Verified


THE ICEBERG OF IGNORANCE
Problems hidden from senior management
Executives see 4% of problems
Team Managers see... 9% of problems
Team Leaders see... 74% of problems
Staff see... 100% of problems
@andydevale


Visibility
THE ICEBERG OF IGNORANCE
Executives see 4% of problems
Team Managers see... 9% of problems
Team Leaders see... 74% of problems
Staff see... 100% of problems
Prob en
fro
m t
@andydevale

Create an AppSec Pipeline and push visibility north



"I am a nice shark, not a mindless eating machine. If I am to change this image, I must first change myself. Fish are friends, not food."

-Bruce, Chum and Anchor

“I am a nice **security professional**, not a mindless **vulnerability spewing** machine. If I am to change this image, I must first change myself. **Developers** are friends, not **fools**.”

-Bruce, Aaron and Matt

I’m with Bruce

@BruceSecDevOps

#BruceSecDevOps™

Matt Tesauro

matt.tesauro@owasp.org / @matt_tesauro

Questions & Thanks

Aaron Weaver

aaron.weaver@owasp.org / @weavera

10SECURITY

References

- Confused panda: https://openclipart.org/detail/69289/confusedpanda
- Jousting Snails - a random twitter post I lost the URL for, sorry
- Julius Caesar quote image: https://quotefancy.com/quote/1740243/Marcus-Junius-Brutus-the-Younger-I-have-not-come-to-praise-Caesar-but-to-bury-him
- Map image: https://openclipart.org/detail/823/two-harbours-map
- Roadmap quote: https://www.brainyquote.com/quotes/earl_nightingale_159044
- Gandoff “Shall pass”: https://shirt.woot.com/offers/halfling-height-requirement
- Pixie dust: http://www.disneyeveryday.com/bottle-of-tinker-bells-pixie-dust-necklace/
- Easy button: https://xposehope.com/2016/11/02/hit-the-easy-button/
- Jar factory: https://www.youtube.com/watch?v=YVqiEMQ1HgA
- Iceberg of Ignorance: https://corporate-rebels.com/iceberg-of-ignorance/