DevSecOps

Essentials

Will begin shortly •••

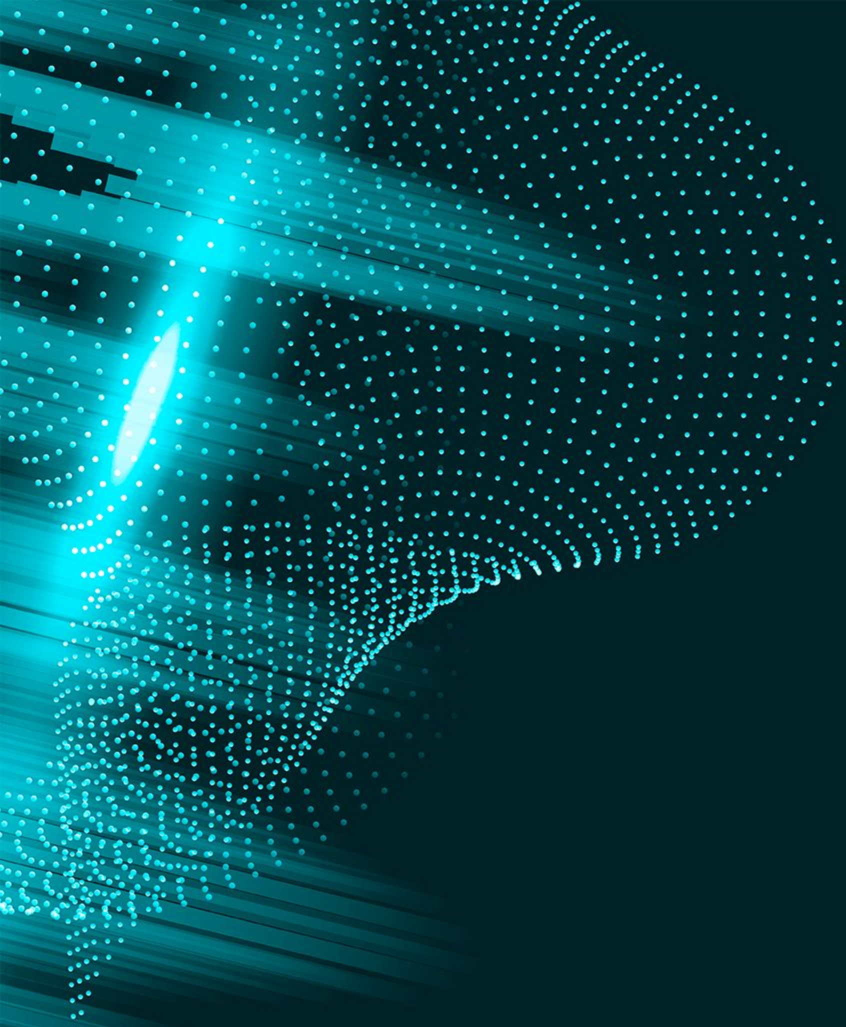
Andrew Krug
andrewkrug@gmail.com
@andrewkrug

Head of
Security
Advocacy @
Datadog

Antisiphon
Instructor for
Cloud Security

AUDIENCE WARNING

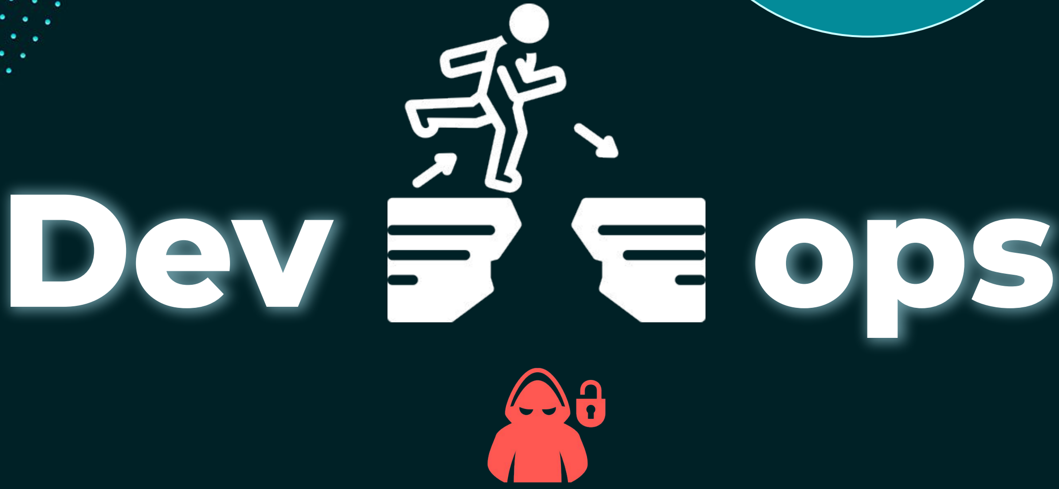
This presentation is full of jargon



Devops

A movement that began in about 2007

Something was missing



Dev ops

Two Movements

The agile software movement

- Speedy Delivery
- Customer Focus
- Tight small feedback loops

The DevOps Movement

- Value automation
- Speed delivery
- Integrate continuous

“When a measure becomes a target, it ceases to be a good measure.”

—Charles Goodhart

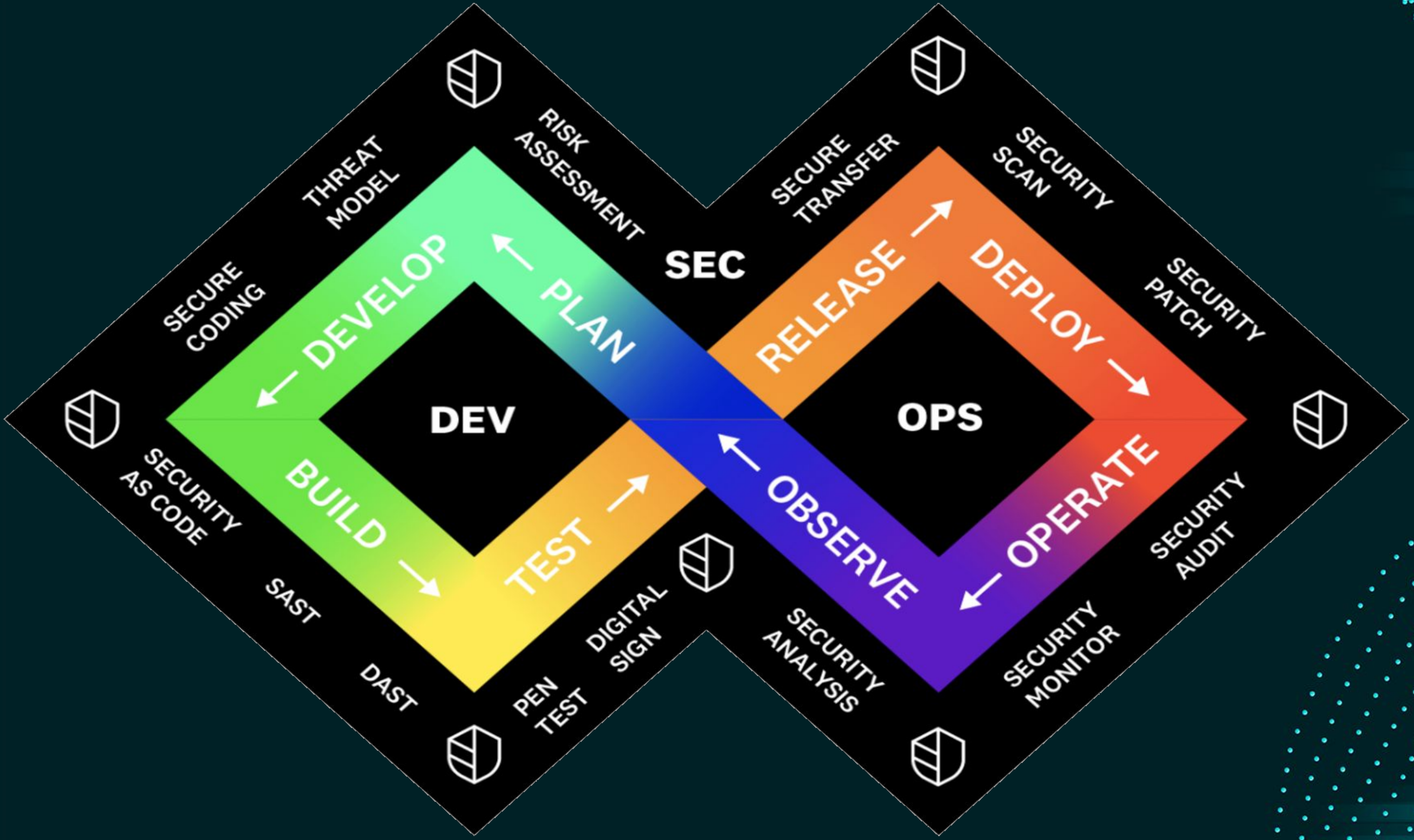
Security

Development

We’ve already seen this …

Purple teaming is a lot of the same within the mico-chasm of security

- Blue teams and red teams working together
- Accelerating feedback
- Using empathy



SEC

DEV

OPS

PLAN →

DEVELOP →

BUILD →

TEST →

RELEASE →

DEPLOY →

OPERATE →

OBSERVE →

THREAT MODEL

RISK ASSESSMENT

SECURE CODING

SECURITY AS CODE

SAST

DAST

PEN TEST

DIGITAL SIGN

SECURE TRANSFER

SECURITY SCAN

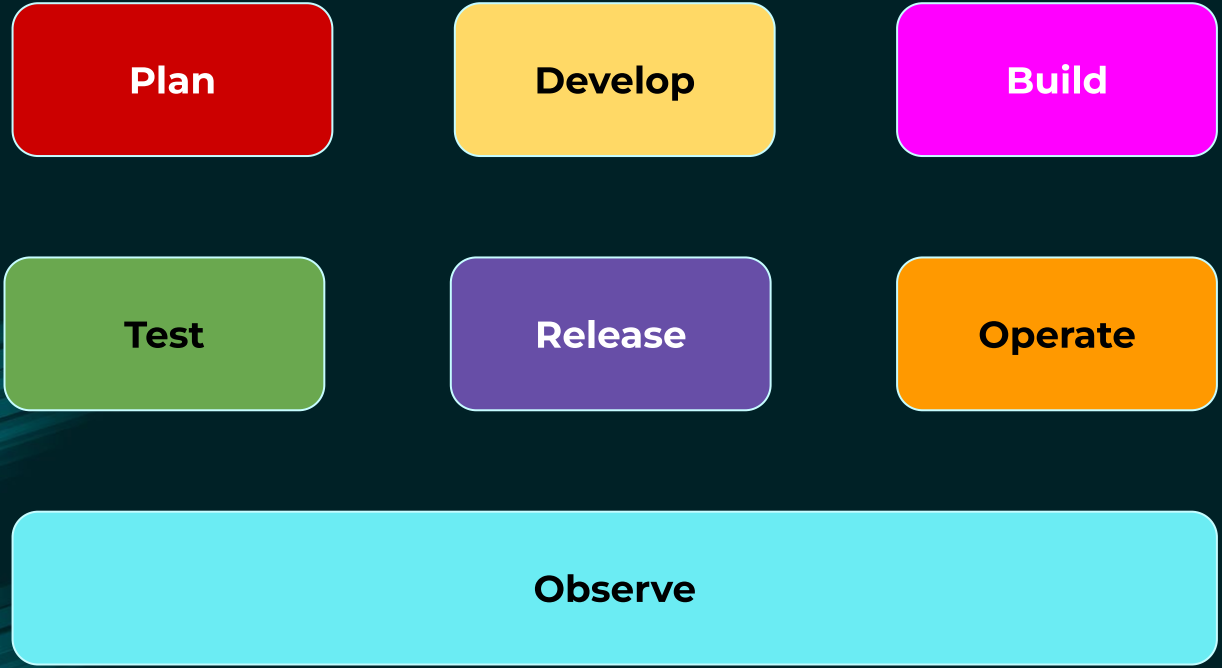
SECURITY PATCH

SECURITY AUDIT

SECURITY MONITOR

SECURITY ANALYSIS

Phases of DevSecops



Plan

Develop

Build

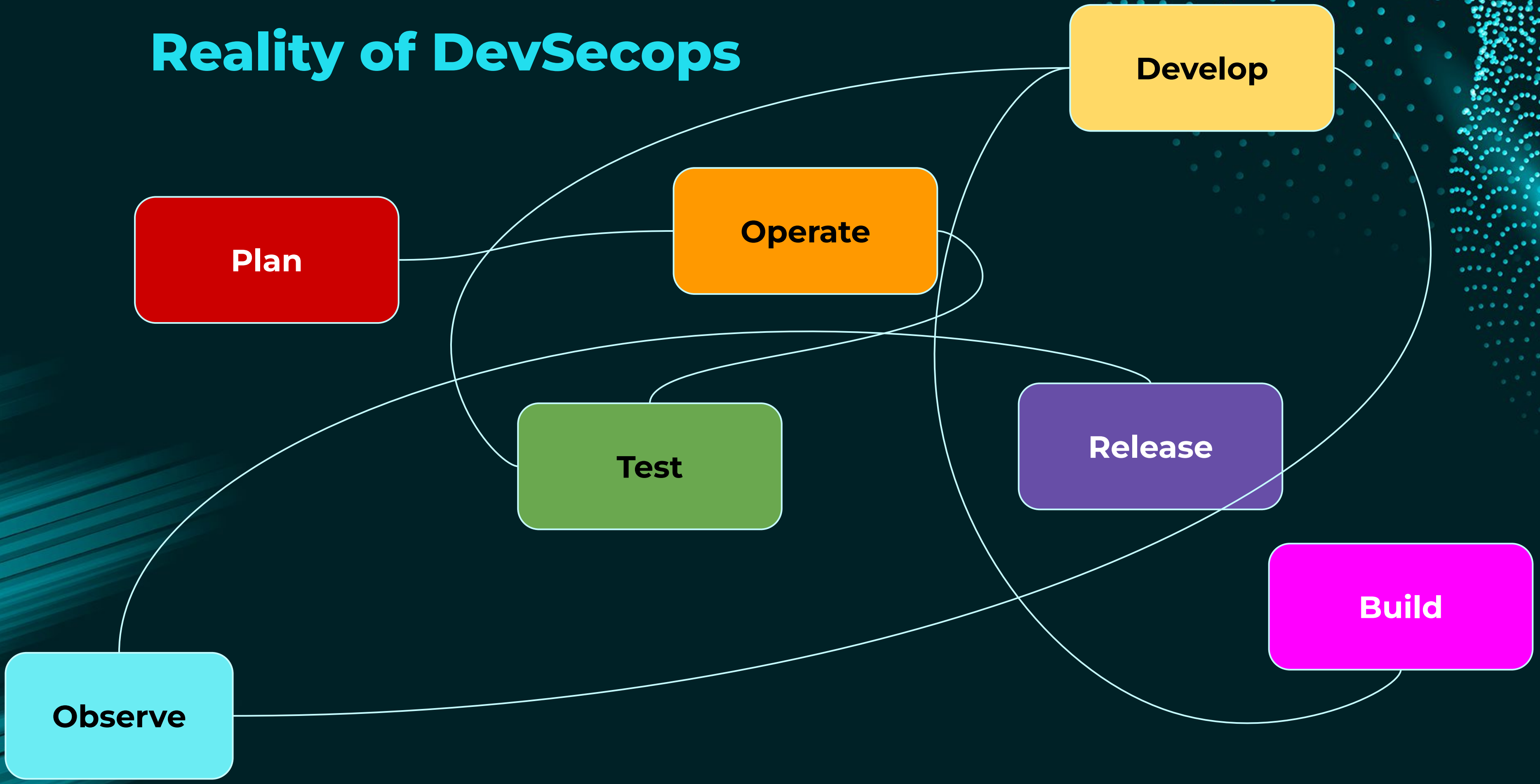
Test

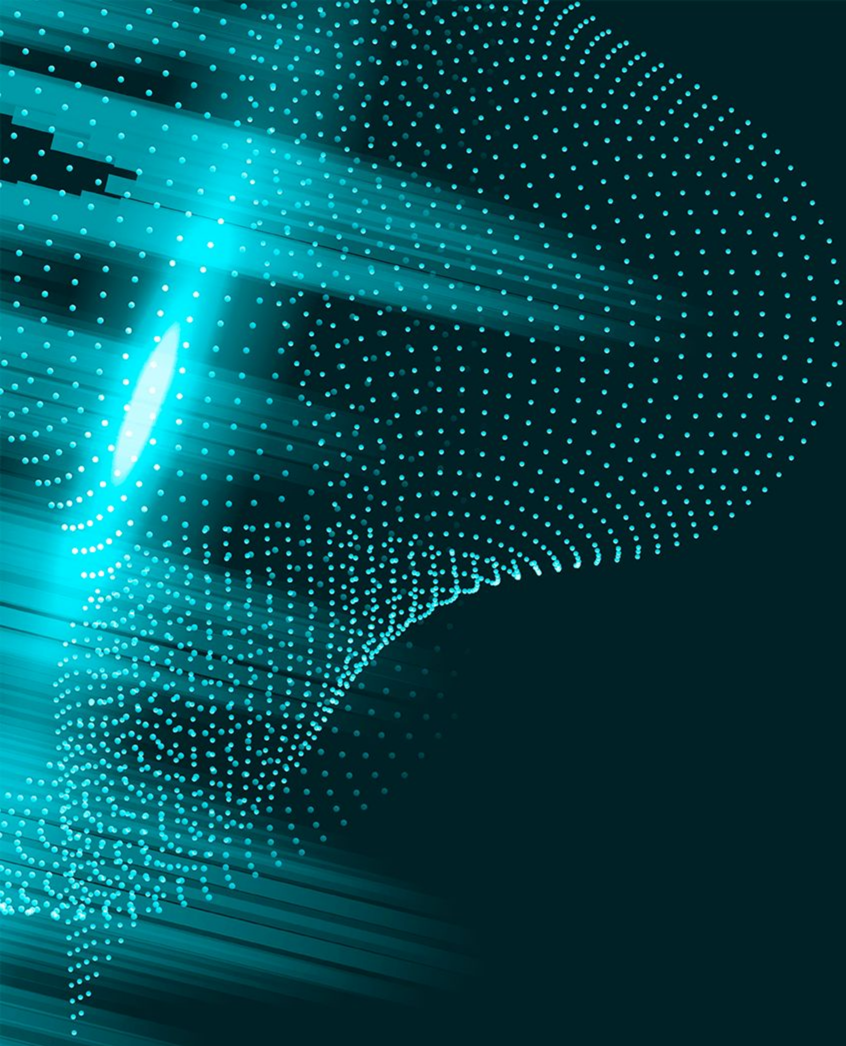
Release

Operate

Observe

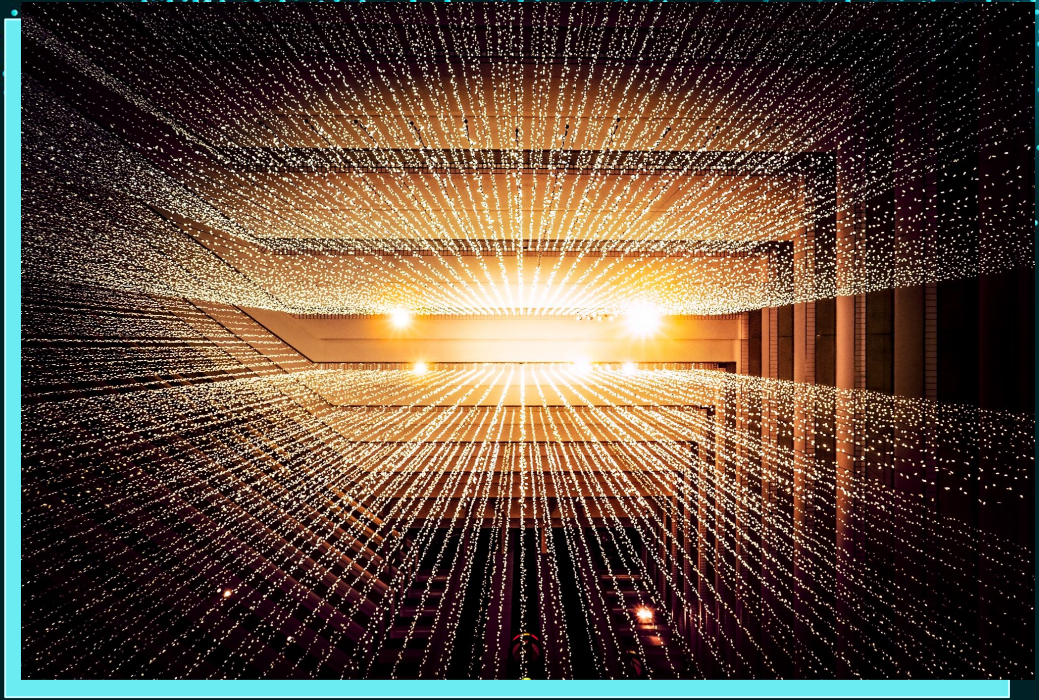
Reality of DevSecops



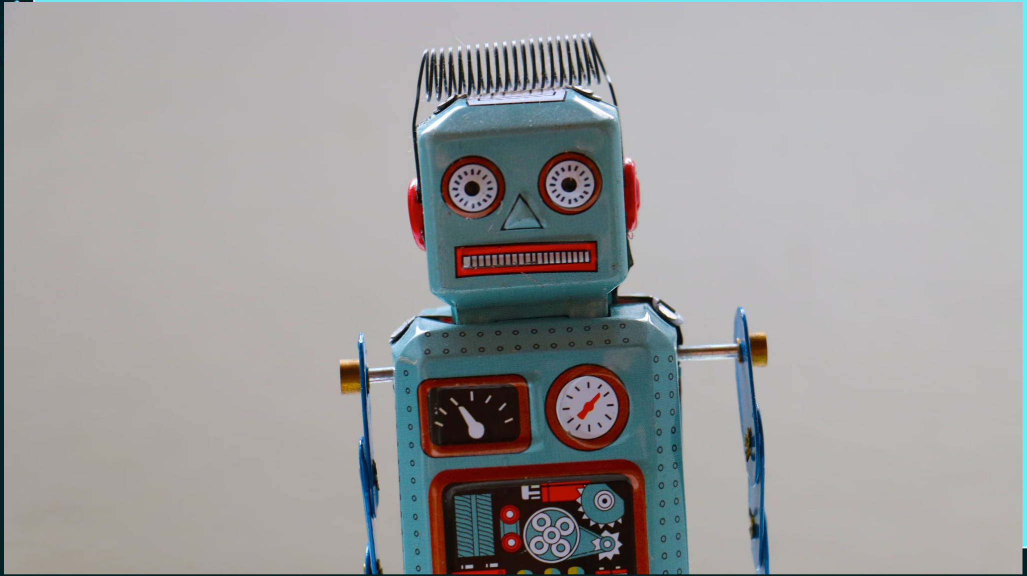


4 Truths of DevSecOps

There is a massive benefit to development and security working together



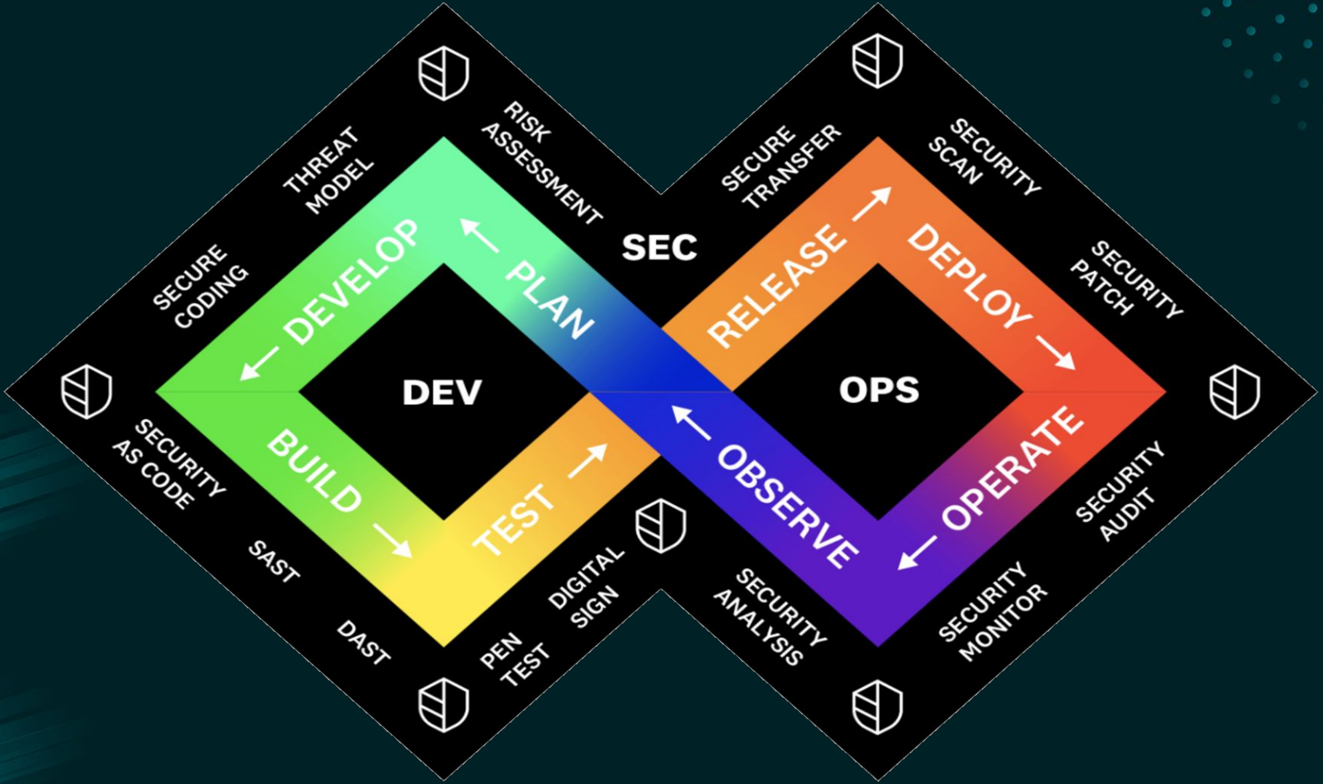
Culture is more important than technology



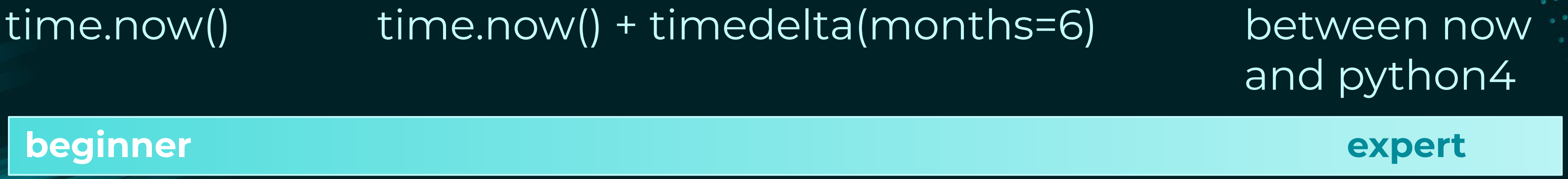
The wrong technology is worse than no technology

You can’t spell DevSecOps without DevOps

What's the biggest bang for the $$$ and time?



Take a maturity approach



time.now()

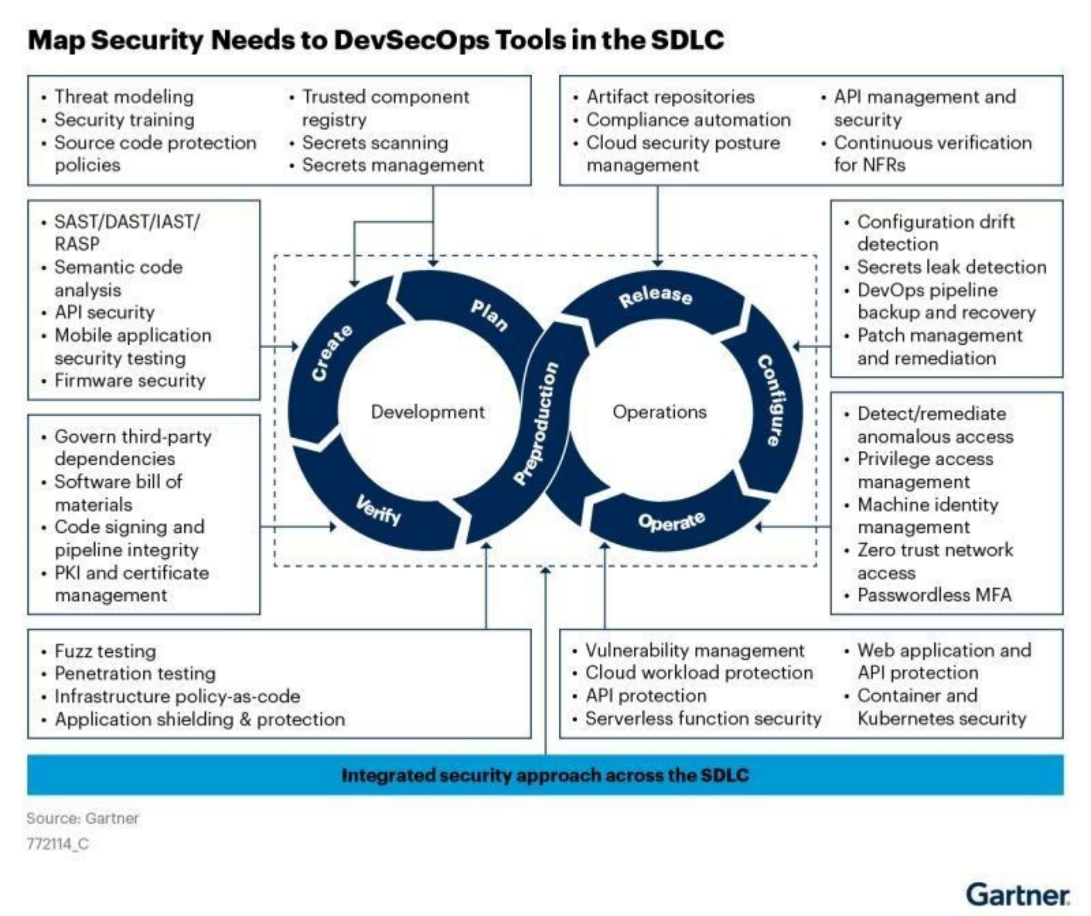
time.now() + timedelta(months=6)

between now and python4

beginner

expert

Trying to do everything can be a bit overwhelming



Map Security Needs to DevSecOps Tools in the SDLC

- Threat modeling
- Security training
- Source code protection policies
- Trusted component registry
- Secrets scanning
- Secrets management

- Artifact repositories
- Compliance automation
- Cloud security posture management
- API management and security
- Continuous verification for NFRs

- SAST/DAST/IAST/ RASP
- Semantic code analysis
- API security
- Mobile application security testing
- Firmware security

- Govern third-party dependencies
- Software bill of materials
- Code signing and pipeline integrity
- PKI and certificate management

- Configuration drift detection
- Secrets leak detection
- DevOps pipeline backup and recovery
- Patch management and remediation

- Detect/remediate anomalous access
- Privilege access management
- Machine identity management
- Zero trust network access
- Passwordless MFA

- Fuzz testing
- Penetration testing
- Infrastructure policy-as-code
- Application shielding & protection

- Vulnerability management
- Cloud workload protection
- API protection
- Serverless function security
- Web application and API protection
- Container and Kubernetes security

Integrated security approach across the SDLC

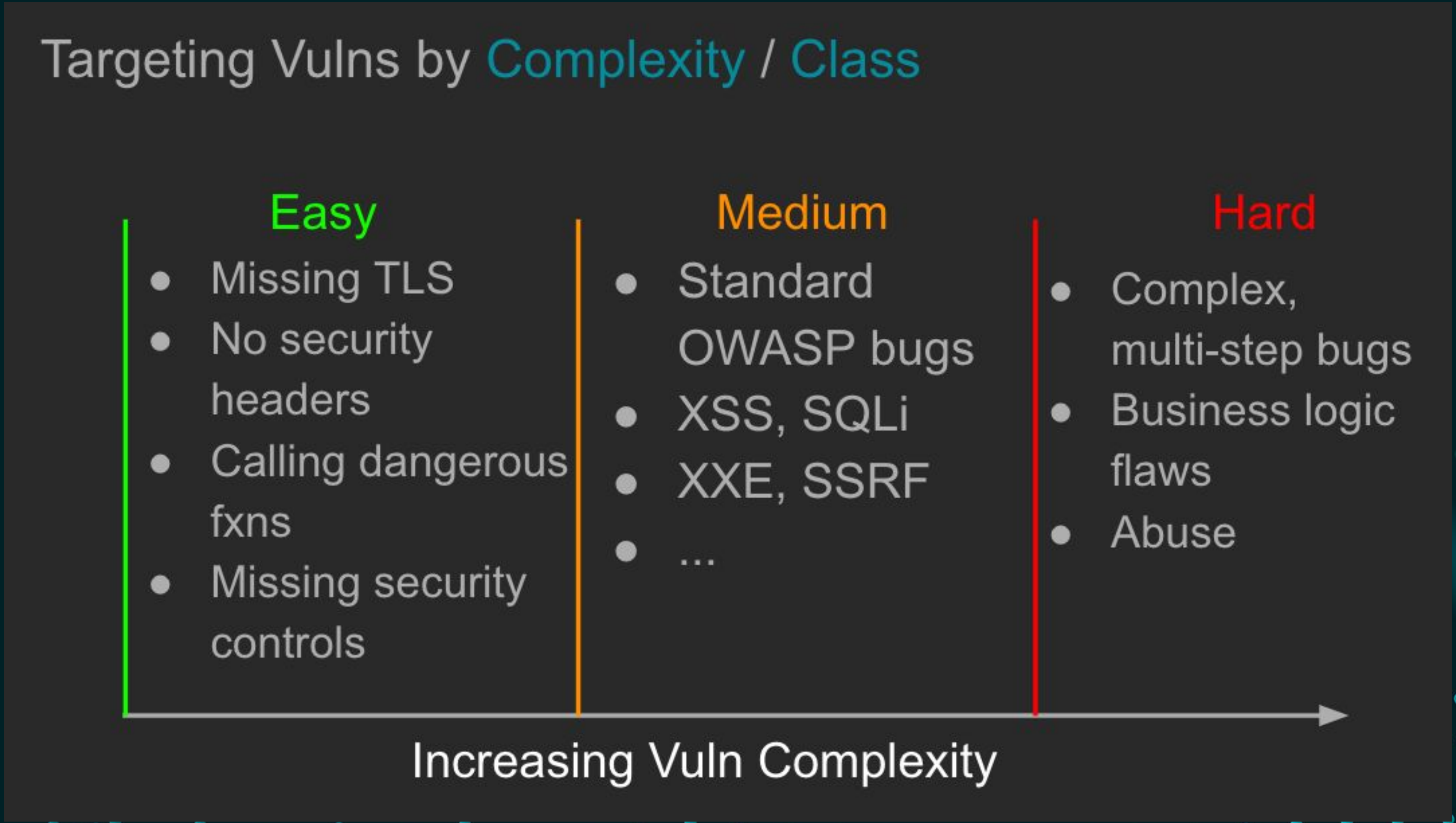
Source: Gartner

772114_C

Gartner

How to 10X Your Security by Clint Gibler 2020

Credit: https://bit.ly/4aEtjzT



Targeting Vulns by Complexity / Class

Easy

- Missing TLS
- No security headers
- Calling dangerous fxns
- Missing security controls

Medium

- Standard OWASP bugs
- XSS, SQLi
- XXE, SSRF
- ...

Hard

- Complex, multi-step bugs
- Business logic flaws
- Abuse

Increasing Vuln Complexity

This is what we’re all still trying to do

This is what we’re all still trying to do

We’re doing at Cloud Scale

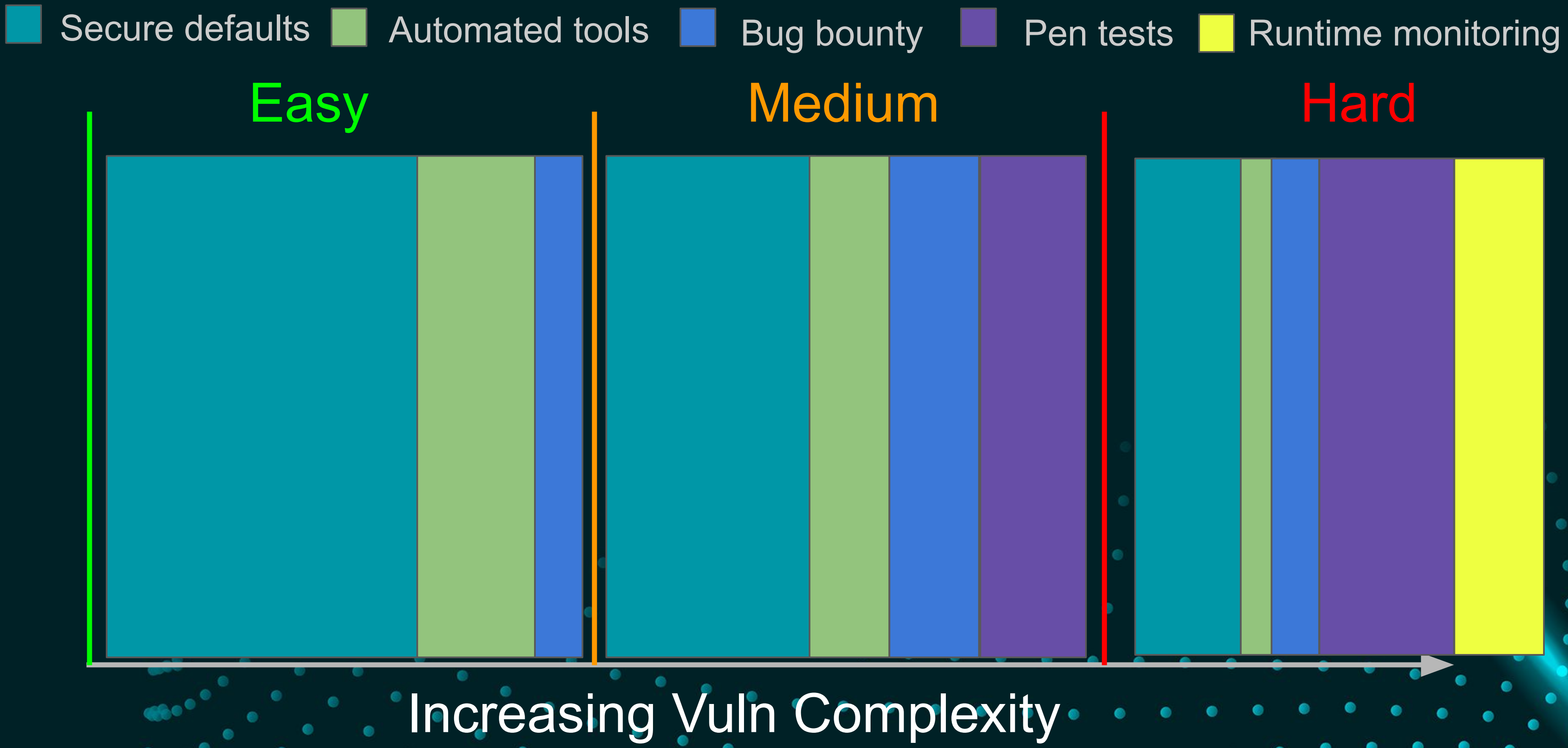
This is what we're all still trying to do

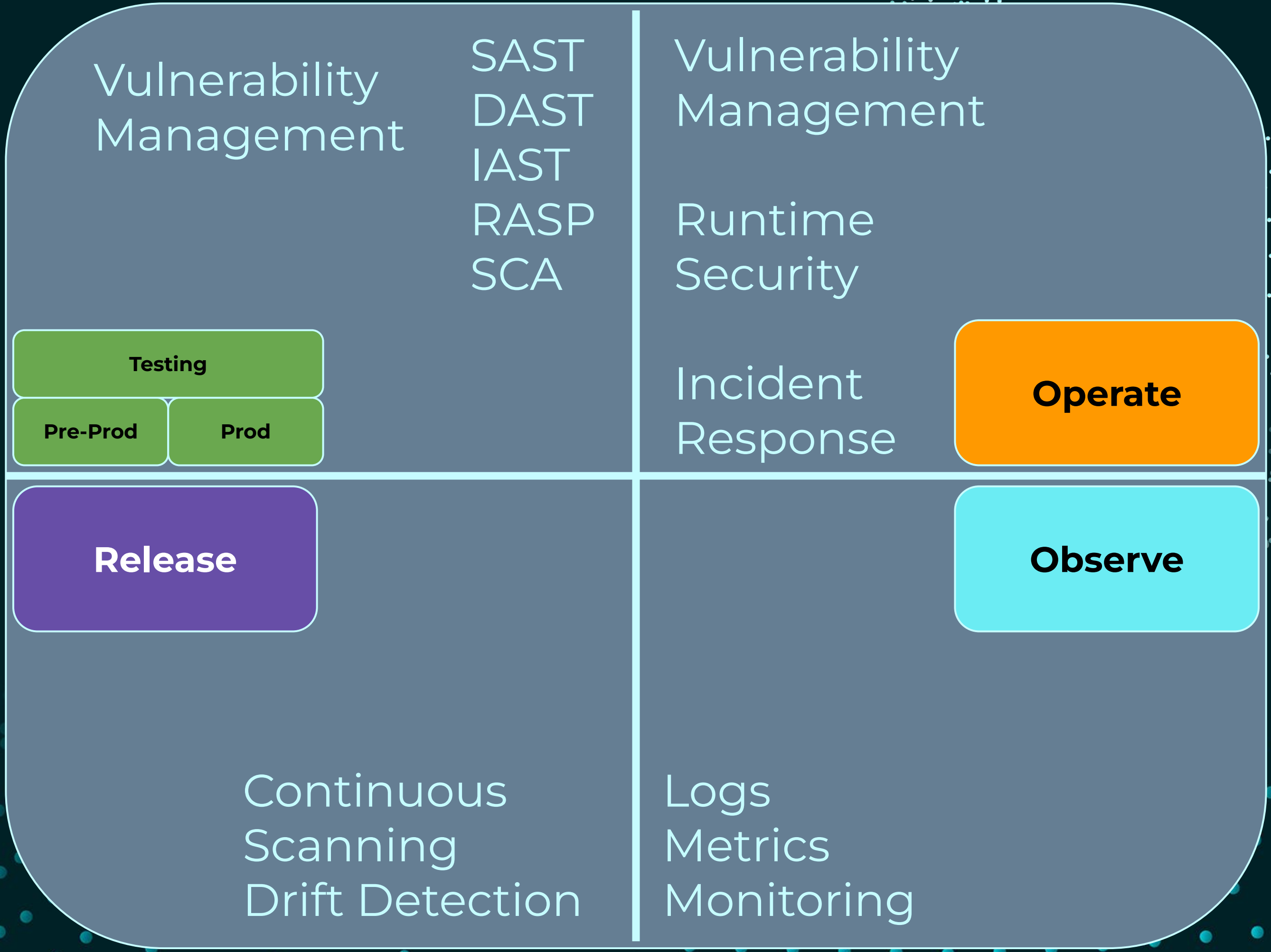
Trying to do it at Cloud Scale

And do it well!

Targeting Vulns by Complexity / Class

Credit: https://bit.ly/4aEtjzT





Vulnerability Management

SAST
DAST
IAST
RASP
SCA

Testing

Pre-Prod

Prod

Vulnerability Management

Runtime Security

Incident Response

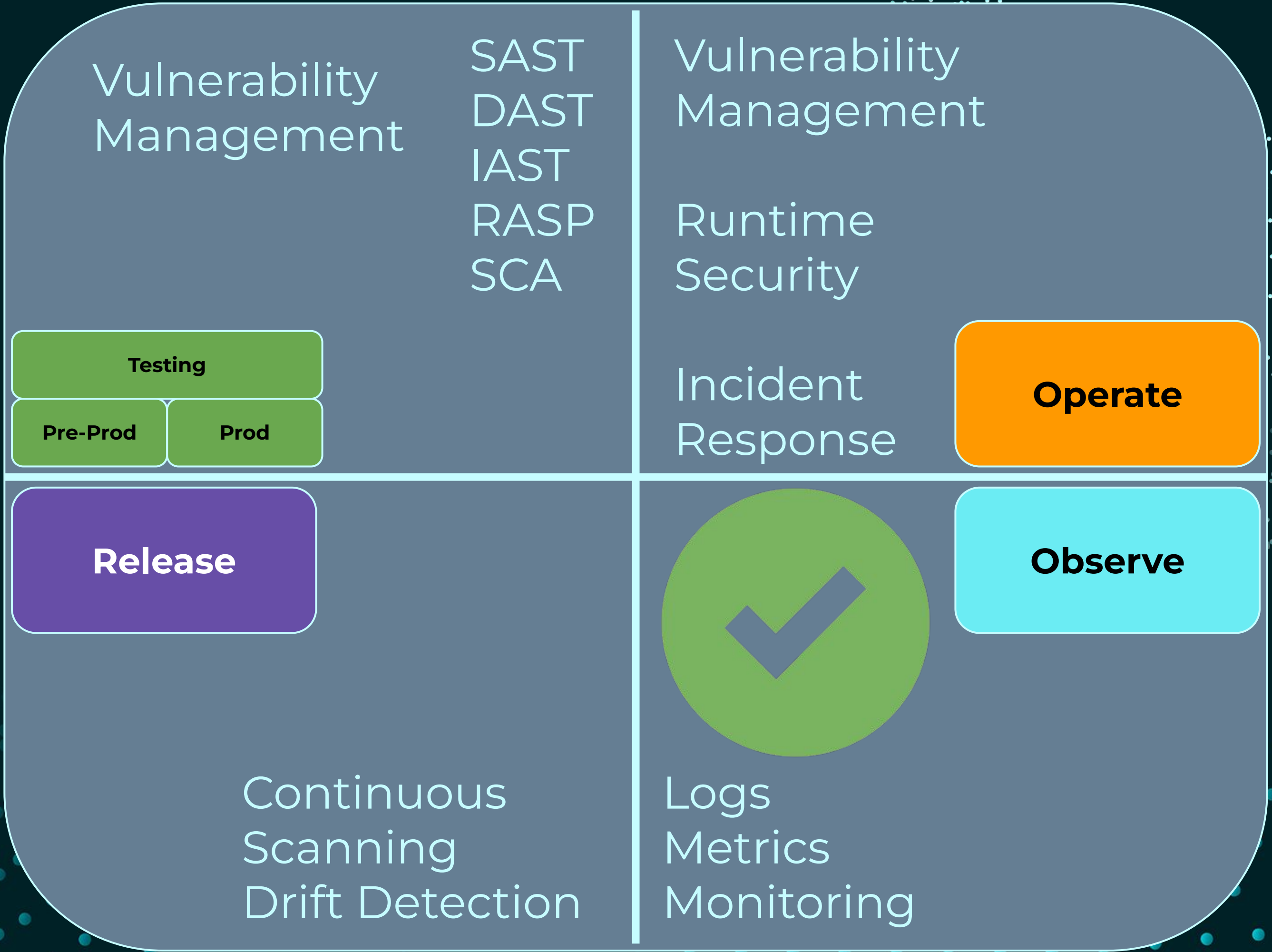
Operate

Release

Continuous Scanning
Drift Detection

Observe

Logs
Metrics
Monitoring



Vulnerability Management

SAST
DAST
IAST
RASP
SCA

Testing

Pre-Prod

Prod

Vulnerability Management

Runtime Security

Incident Response

Operate

Release

Continuous Scanning
Drift Detection

✓

Observe

Logs
Metrics
Monitoring

Table Stakes

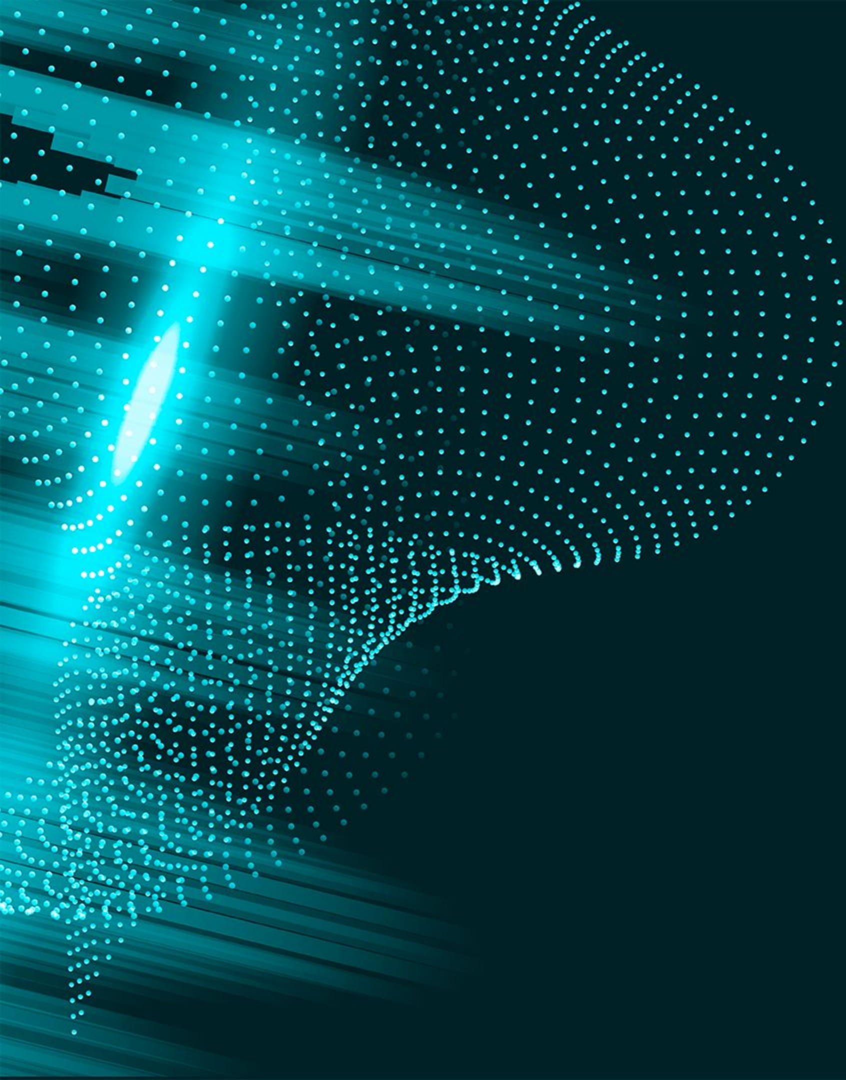
Asset Inventory

Vulnerability Management

- In production and pre-production

Continuous Scanning

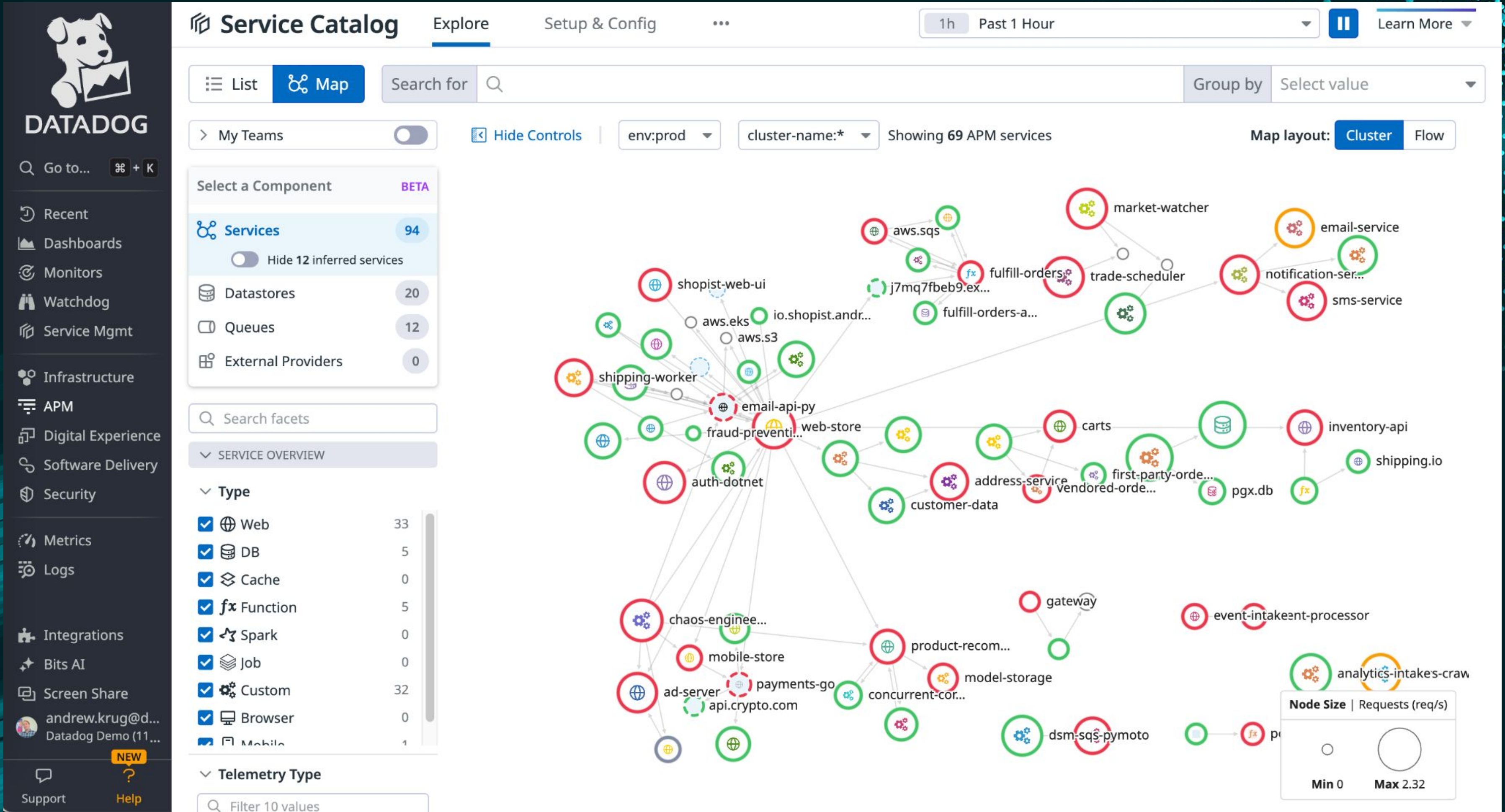
- In code
- Cloud



Asset Inventory

Who you gonna call?

Modern Applications are Complex



Code

.inventory

Keep owner info
close to the source

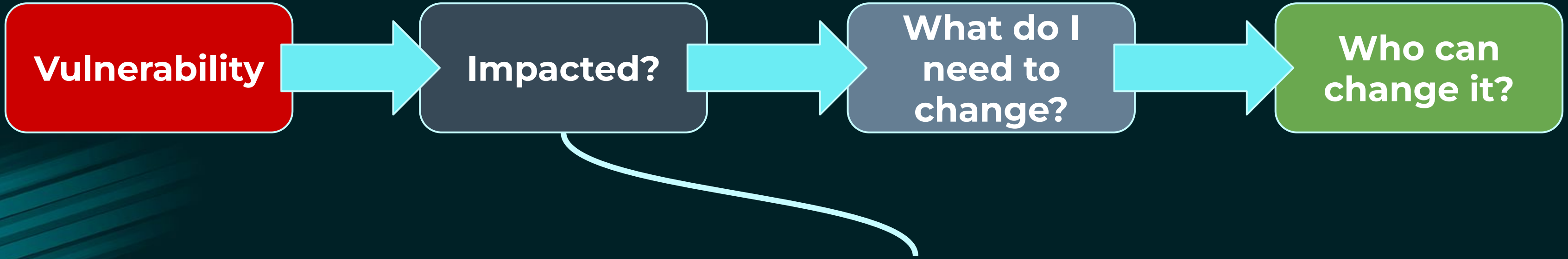
Source code custody
chains if possible

Cloud

Tagging
Data lakes

Owner
Environment
Service

Inventory data should be an accelerator for triage and remediation



Vulnerability

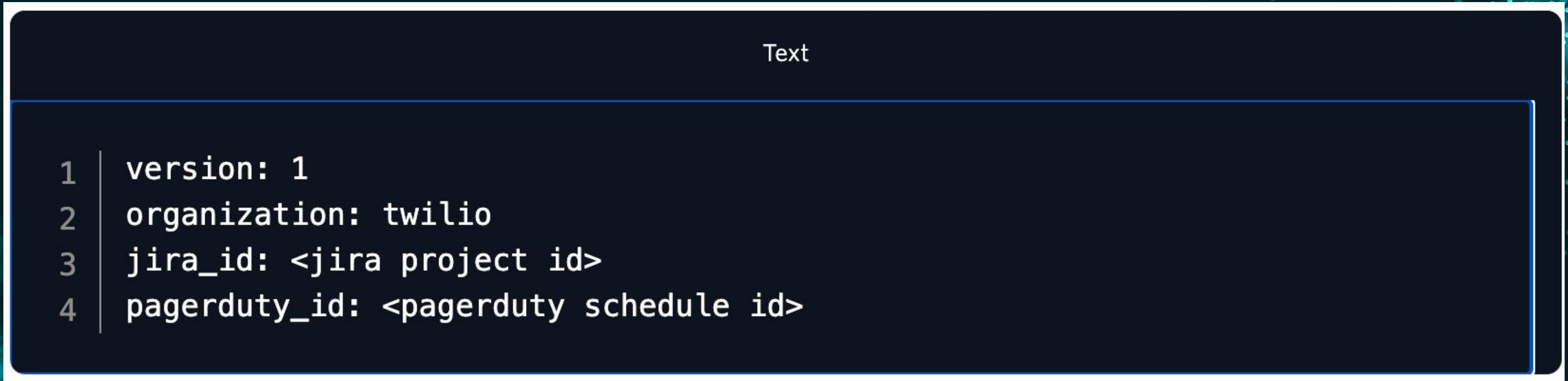
Impacted?

What do I need to change?

Who can change it?

Are my volumes encrypted?
Do I have that XYZ in my cloud?
Does that law apply to this region?

In code inventory metadata



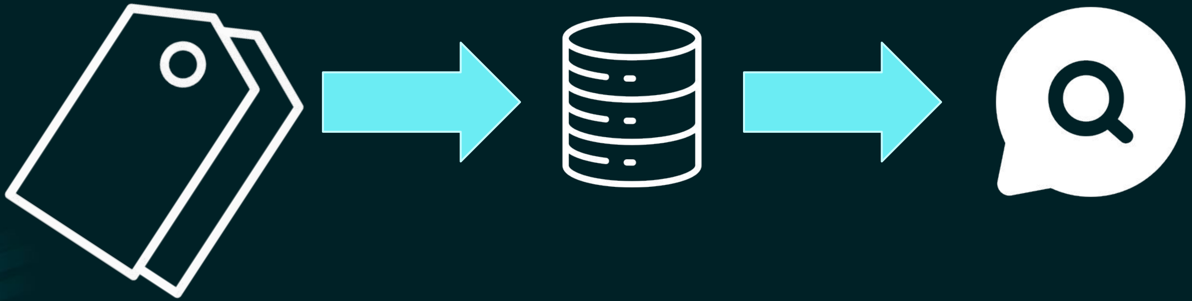
Text

```
version: 1
organization: twilio
jira_id: <jira project id>
pagerduty_id: <pagerduty schedule id>
```

Can The Real Codeowners Please Stand Up? Code Provenance at Scale

https://bit.ly/3J1cPG6

Cloud Inventory



Powered by tags

Stored in a data-lake

Searchable outside of your production account

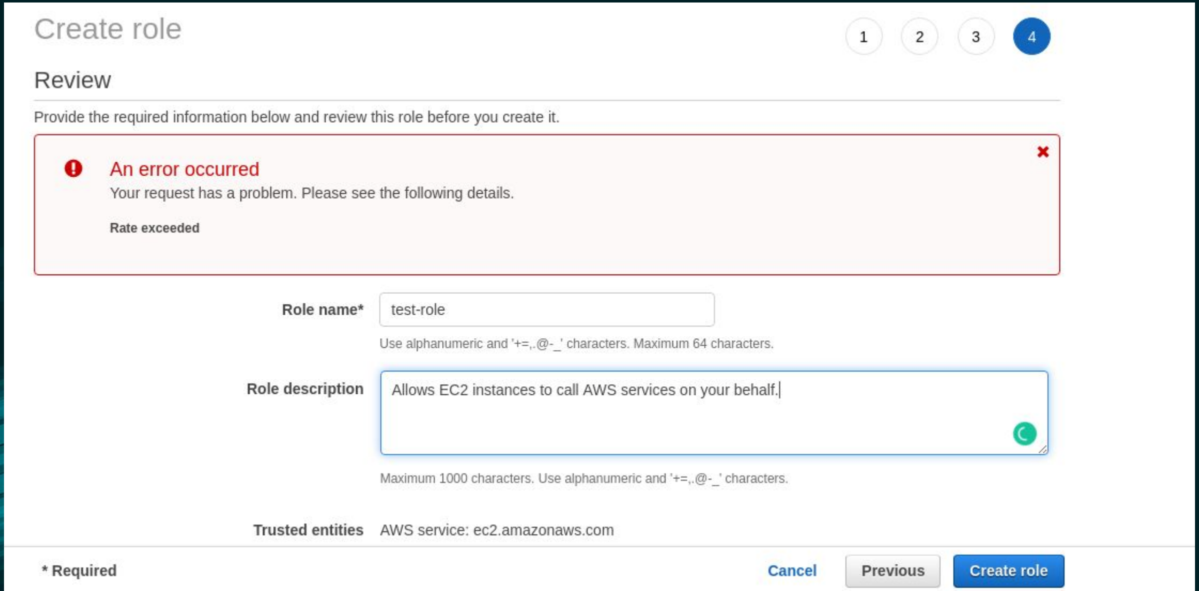
Cloud Inventory

Powered by tags

Stored in a data-lake

Searchable outside of your production account

Cloud Inventory | Why non-production?



Review

Provide the required information below and review this role before you create it.

An error occurred

Your request has a problem. Please see the following details.

Rate exceeded

Role name*	test-role
	Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.
Role description	Allows EC2 instances to call AWS services on your behalf.
	Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.
Trusted entities	AWS service: ec2.amazonaws.com

*** Required**

Inventory systems can DoS Production

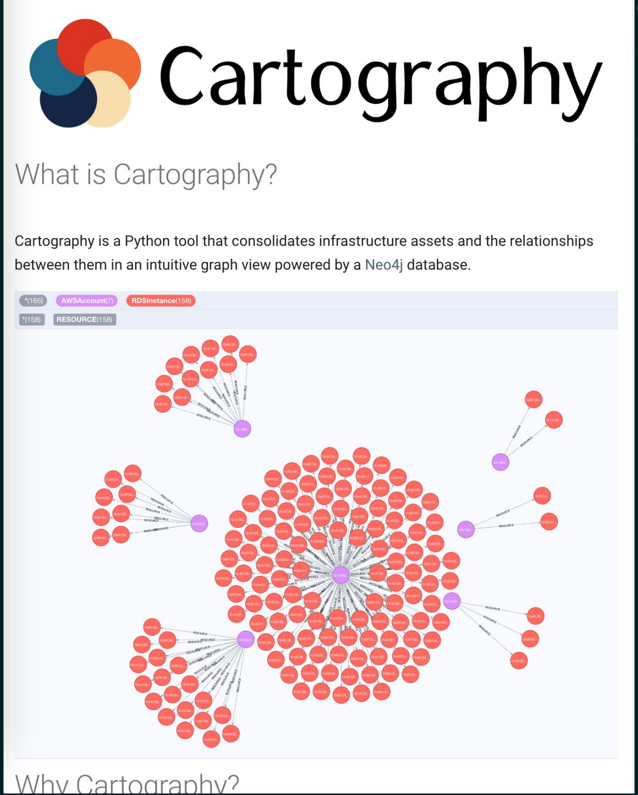
Searchable outside of your production account

Open Source Inventory Solutions

Purpose built for security inquiry

Uses relationships to map relative risk

https://github.com/lyft/cartography



Cartography

What is Cartography?

Cartography is a Python tool that consolidates infrastructure assets and the relationships between them in an intuitive graph view powered by a Neo4j database.

Why Cartography?

This is one area where a commercial option could be better / more reliable

What is the best use of your AppSec team’s limited time?

Vulnerability Management

Pre-Prod and In Prod

Remember the good old days?

Scheduled task:

```
yum update -y

apt-get update && apt-get upgrade -y

wuauclt.exe /updatenow
shutdown -r -t 0
```

Vulnerability Management

Into the great beyond

Vulnerabilities can be:

- Third party dependencies
- Indirect third party dependencies
- Engineering vulnerability

  (your code)
- OS vulnerability
- Cloud provider vulnerability
- Orchestrator vulnerability

Vulnerability Management

Into the great beyond

Vulnerabilities can be:

- **Third party dependencies**
- **Indirect third party dependencies**
- Engineering vulnerability

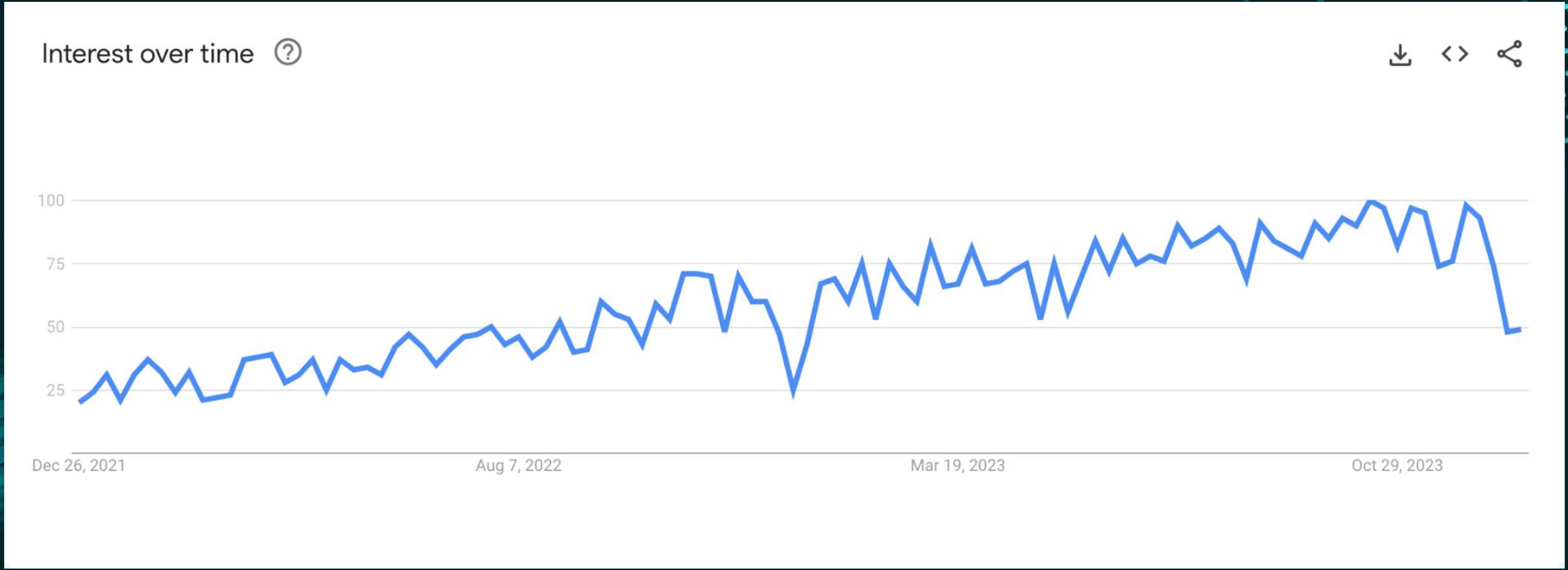
  (your code)
- OS vulnerability
- Cloud provider vulnerability
- Orchestrator vulnerability

Dependency Resolution

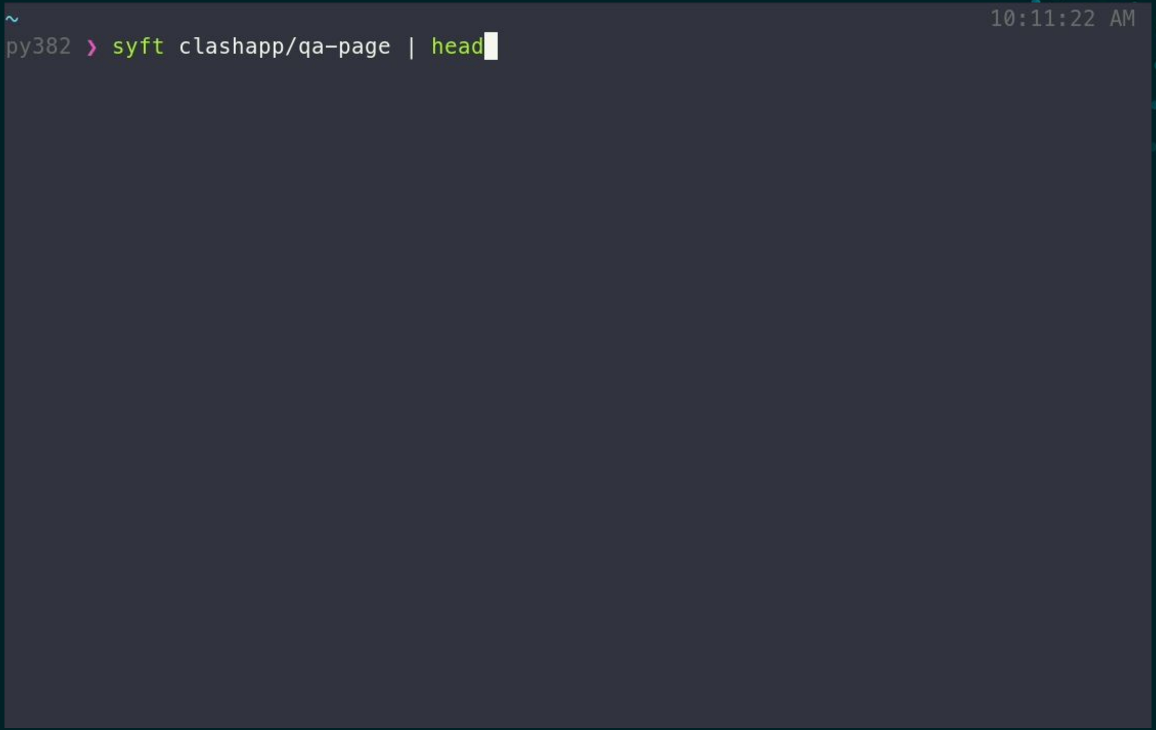
SBOM saves the day?

Executive order 14028 calls for all software vendors to the US government to list the components that they used to create their products with software bill of materials (SBOM) documentation by **September 2023.**

Interest is growing



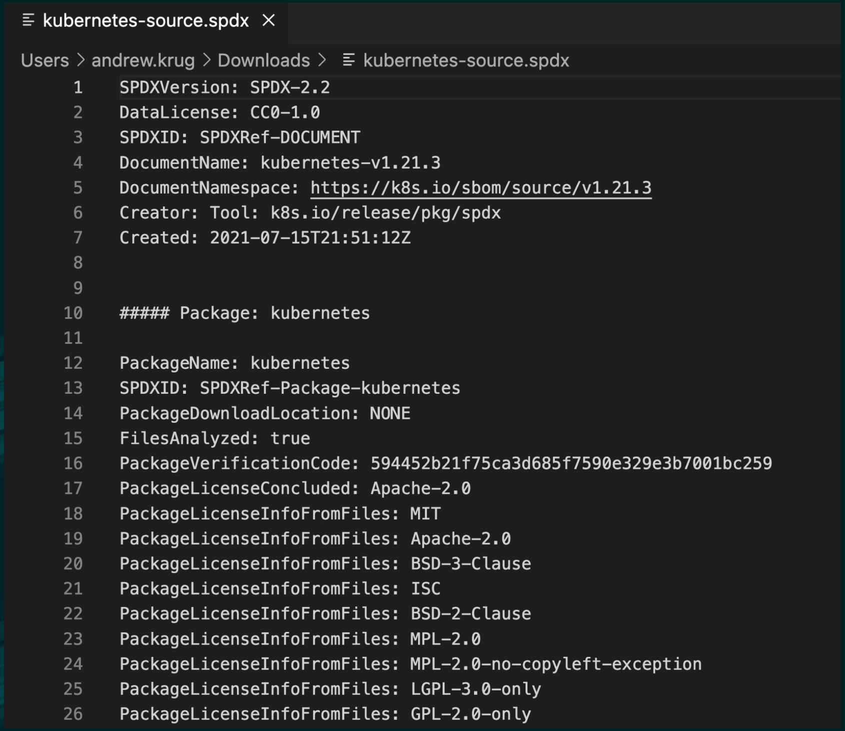
Generating SBOM is easy



```
~                                                                    10:11:22 AM
py382 ❯ syft clashapp/qa-page | head
```

https://github.com/anchore/syft

Knowing what to do with it is a challenge

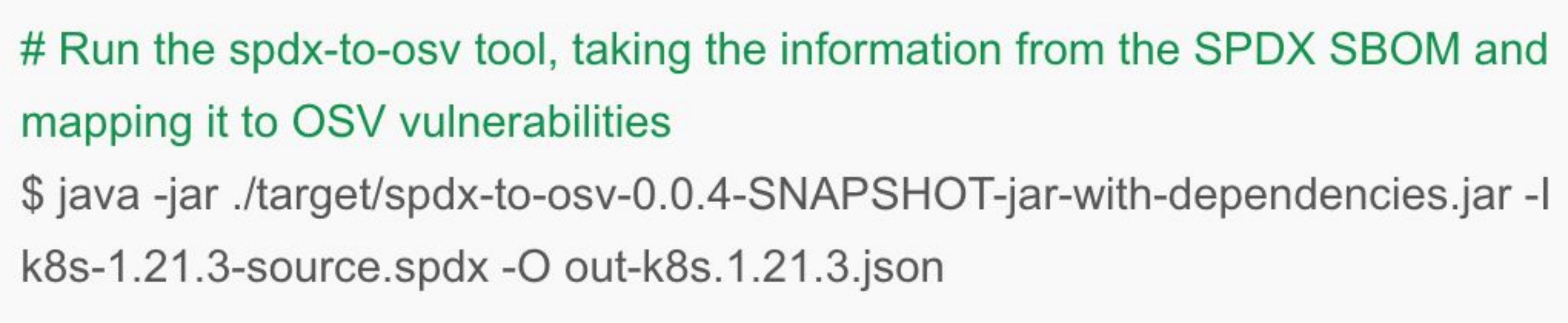


```
SPDXVersion: SPDX-2.2
DataLicense: CC0-1.0
SPDXID: SPDXRef-DOCUMENT
DocumentName: kubernetes-v1.21.3
DocumentNamespace: https://k8s.io/sbom/source/v1.21.3
Creator: Tool: k8s.io/release/pkg/spdx
Created: 2021-07-15T21:51:12Z


##### Package: kubernetes

PackageName: kubernetes
SPDXID: SPDXRef-Package-kubernetes
PackageDownloadLocation: NONE
FilesAnalyzed: true
PackageVerificationCode: 594452b21f75ca3d685f7590e329e3b7001bc259
PackageLicenseConcluded: Apache-2.0
PackageLicenseInfoFromFiles: MIT
PackageLicenseInfoFromFiles: Apache-2.0
PackageLicenseInfoFromFiles: BSD-3-Clause
PackageLicenseInfoFromFiles: ISC
PackageLicenseInfoFromFiles: BSD-2-Clause
PackageLicenseInfoFromFiles: MPL-2.0
PackageLicenseInfoFromFiles: MPL-2.0-no-copyleft-exception
PackageLicenseInfoFromFiles: LGPL-3.0-only
PackageLicenseInfoFromFiles: GPL-2.0-only
```

SPDX format

Some projects like Kubernetes make these available

SPDX => OSV to use free databases



```
# Run the spdx-to-osv tool, taking the information from the SPDX SBOM and
mapping it to OSV vulnerabilities
$ java -jar ./target/spdx-to-osv-0.0.4-SNAPSHOT-jar-with-dependencies.jar -I
k8s-1.21.3-source.spdx -O out-k8s.1.21.3.json
```

https://security.googleblog.com/2022/06/sbom-in-action-finding-vulnerabilities.html

https://ossf.github.io/osv-schema

https://github.com/spdx/spdx-to-osv

Issues with plain SBOM

Some can be quite large

- ○ How do you triage an SBOM for a container image bigger than 1GB

High rate of false positive / low fidelity alerts

Enter SCA
(Software Composition Analysis)

Vendors triage so you don’t have to

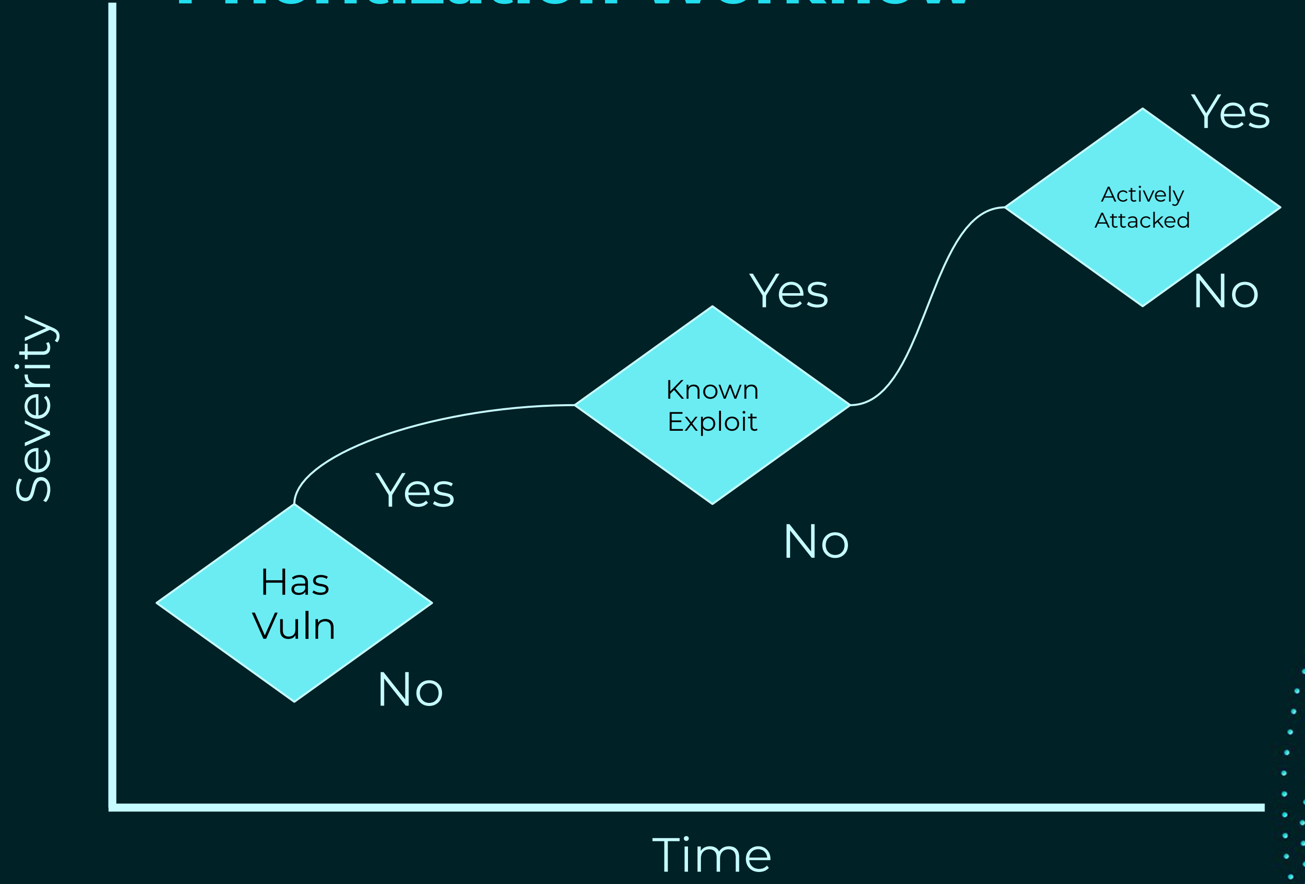
Attributes of great SCA

- SBOM Informed but not SBOM driven

- Built in workflow and prioritization

- Groups findings to resolve in batch

Prioritization Workflow



Goal is always to stop bugs pre-production …

Sometimes they make it there over time or bypass tooling

Continuous Scanning

Code and Cloud

SASST, DAST, and IAST

Oh my!!!

SAST - runs against code to detect known bad patterns in code using signatures

DAST - spin up the app and test it while it’s running

IAST - whitebox version of DAST with specific cases

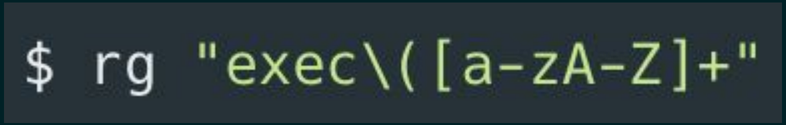
Less Complex

More Complex

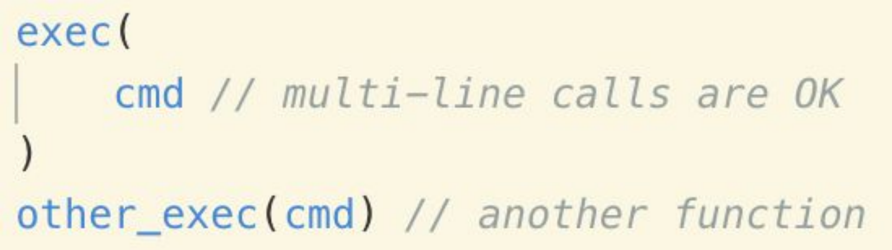
Please don’t DIY this stuff

Static Analysis - Rolling Your Own

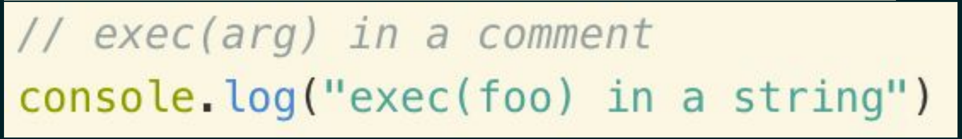
- Approach: Source code -> [Parser] -> AST
  - Lang-specific parser, ANTLR, (best) multi-lang parser semantic, bblfsh



```
$ rg "exec\([a-zA-Z]+"
```



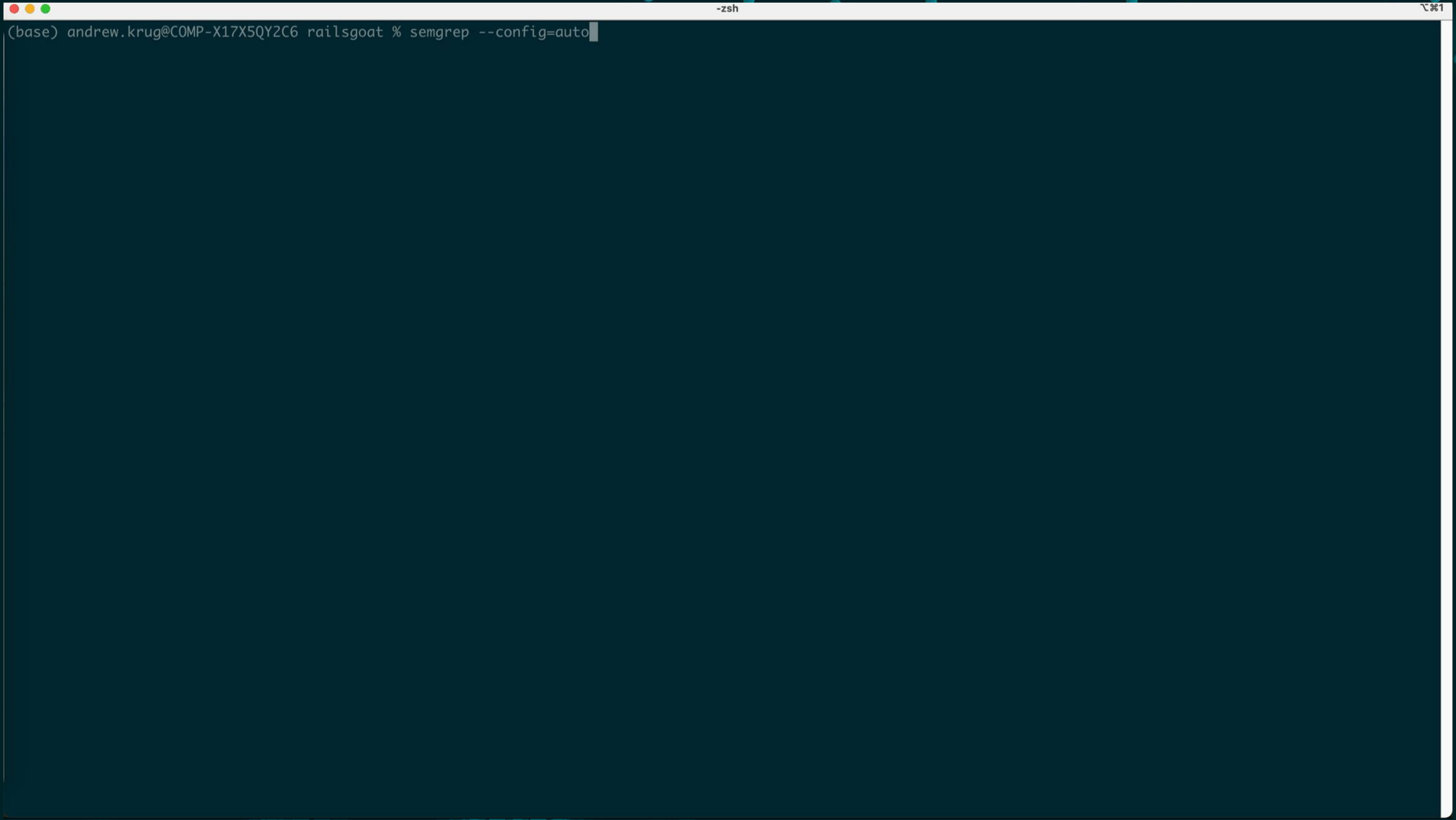
```
exec(
    cmd // multi-line calls are OK
)
other_exec(cmd) // another function
```



```
// exec(arg) in a comment
console.log("exec(foo) in a string")
```

Credit: https://bit.ly/4aEtjzT

OSS Solutions | SAST



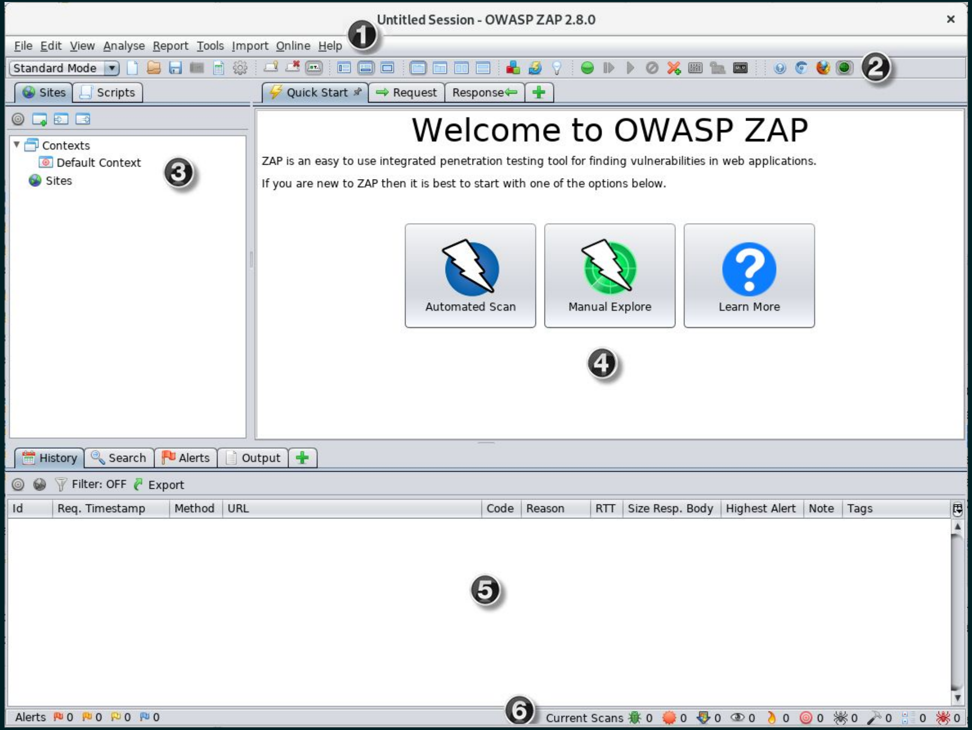
```
(base) andrew.krug@COMP-X17X5QY2C6 railsgoat % semgrep --config=auto
```

DAST Testing

DAST is a **“Black-Box” testing**, can find security vulnerabilities and weaknesses in a running application by injecting malicious payloads to identify potential flaws that allow for attacks like SQL injections or cross-site scripting (XSS), etc.

DAST Testing

https://www.zaproxy.org/

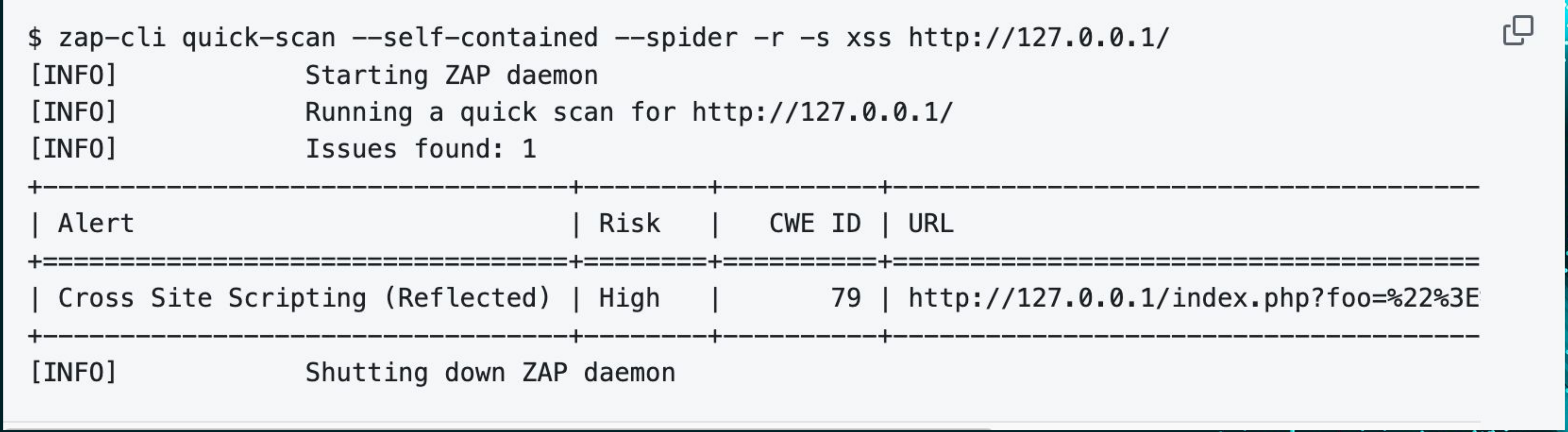


Welcome to OWASP ZAP

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

If you are new to ZAP then it is best to start with one of the options below.

DAST in a pipeline



```
$ zap-cli quick-scan --self-contained --spider -r -s xss http://127.0.0.1/
[INFO]            Starting ZAP daemon
[INFO]            Running a quick scan for http://127.0.0.1/
[INFO]            Issues found: 1
+---------------------------------+--------+----------+--------------------------------------
| Alert                           | Risk   |   CWE ID | URL
+=================================+========+==========+======================================
| Cross Site Scripting (Reflected) | High   |       79 | http://127.0.0.1/index.php?foo=%22%3E
+---------------------------------+--------+----------+--------------------------------------
[INFO]            Shutting down ZAP daemon
```

https://github.com/Grunny/zap-cli

IAST – Still emerging

IAST (interactive application security testing) is an application security testing method that tests the application while the app is run by an automated test, human tester, or any activity “interacting” with the application functionality.

IAST – OSS

CSPM
Cloud Security Posture Management

Hunt down misconfigurations before they are exploited

Public buckets, open security groups etc

Two types of CSPM

Point and shoot

- Prowler OSS

  https://github.com/prowler-cloud/prowler

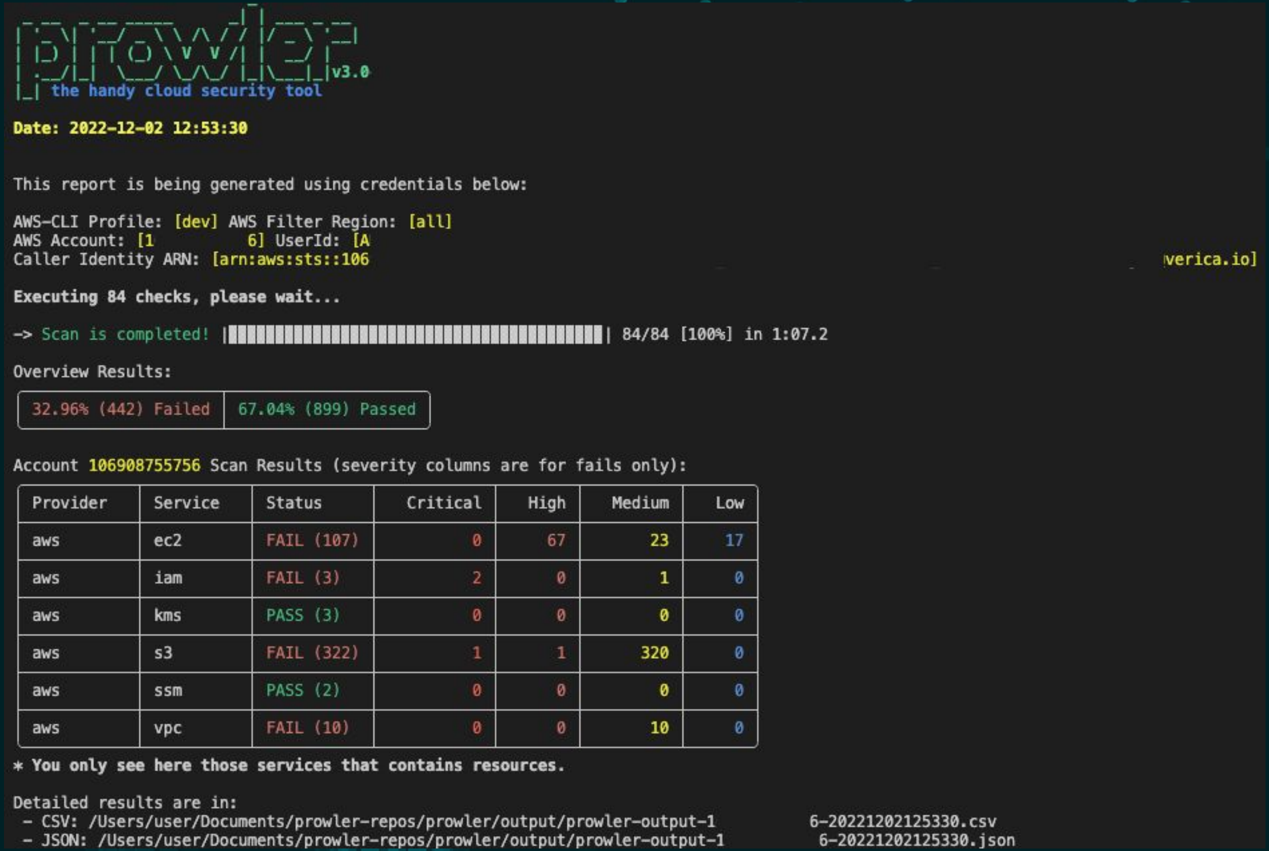
- ScoutSuite

  https://github.com/nccgroup/ScoutSuite

Continuous Scanning

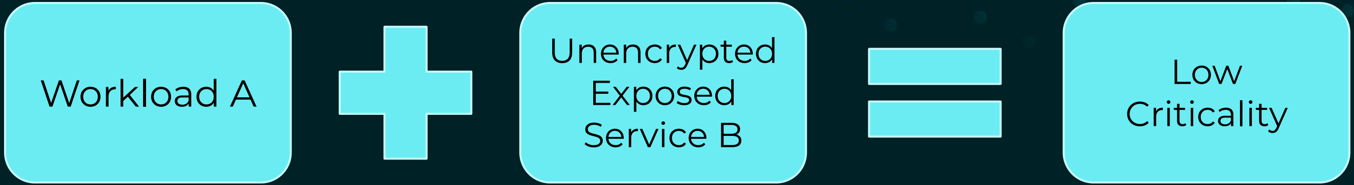
- AWS Config
- Point and shoots with cron
- Cloud Custodian
- Commercial offerings like Datadog CSM

Favorite OSS CSPM



```
                         _
 _ __  _ __ _____      _| | ___ _ __
| '_ \| '__/ _ \ \ /\ / / |/ _ \ '__|
| |_) | | | (_) \ V  V /| |  __/ |
| .__/|_|  \___/ \_/\_/ |_|\___|_|v3.0
|_| the handy cloud security tool

Date: 2022-12-02 12:53:30


This report is being generated using credentials below:

AWS-CLI Profile: [dev] AWS Filter Region: [all]
AWS Account: [1 [redacted] 6] UserId: [A [redacted]
Caller Identity ARN: [arn:aws:sts::106 [redacted] verica.io]

Executing 84 checks, please wait...

-> Scan is completed! |████████████████████████████████████████| 84/84 [100%] in 1:07.2

Overview Results:
╭────────────────────┬────────────────────╮
│ 32.96% (442) Failed │ 67.04% (899) Passed │
╰────────────────────┴────────────────────╯

Account 106908755756 Scan Results (severity columns are for fails only):
╭────────────┬───────────┬────────────┬────────────┬────────┬──────────┬───────╮
│ Provider   │ Service   │ Status     │   Critical │   High │   Medium │   Low │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ ec2       │ FAIL (107) │          0 │     67 │       23 │    17 │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ iam       │ FAIL (3)   │          2 │      0 │        1 │     0 │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ kms       │ PASS (3)   │          0 │      0 │        0 │     0 │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ s3        │ FAIL (322) │          1 │      1 │      320 │     0 │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ ssm       │ PASS (2)   │          0 │      0 │        0 │     0 │
├────────────┼───────────┼────────────┼────────────┼────────┼──────────┼───────┤
│ aws        │ vpc       │ FAIL (10)  │          0 │      0 │       10 │     0 │
╰────────────┴───────────┴────────────┴────────────┴────────┴──────────┴───────╯
* You only see here those services that contains resources.

Detailed results are in:
 - CSV: /Users/user/Documents/prowler-repos/prowler/output/prowler-output-1 [redacted] 6-20221202125330.csv
 - JSON: /Users/user/Documents/prowler-repos/prowler/output/prowler-output-1 [redacted] 6-20221202125330.json
```

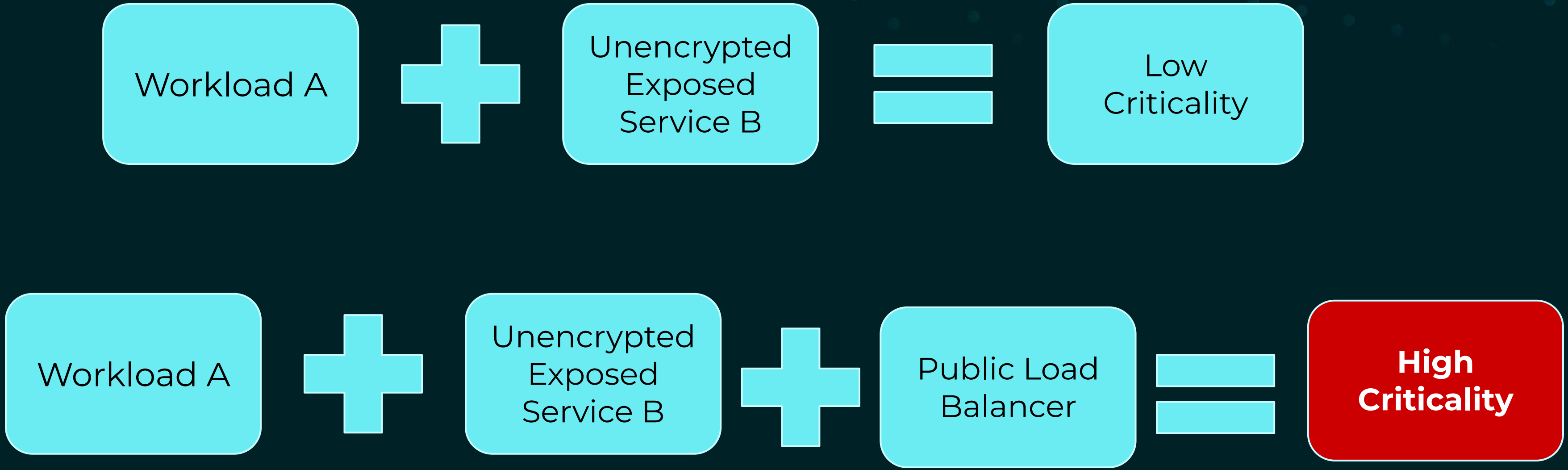
The next generation of CSPM



Workload A + Unencrypted Exposed Service B = Low Criticality

More context for dynamic criticality

The next generation of CSPM



Workload A + Unencrypted Exposed Service B = Low Criticality

Workload A + Unencrypted Exposed Service B + Public Load Balancer = **High Criticality**

More context for dynamic criticality

The trends

Tool Sprawl

More tools providing findings and signals than ever

Triage Pain

Engineers don’t know where to start

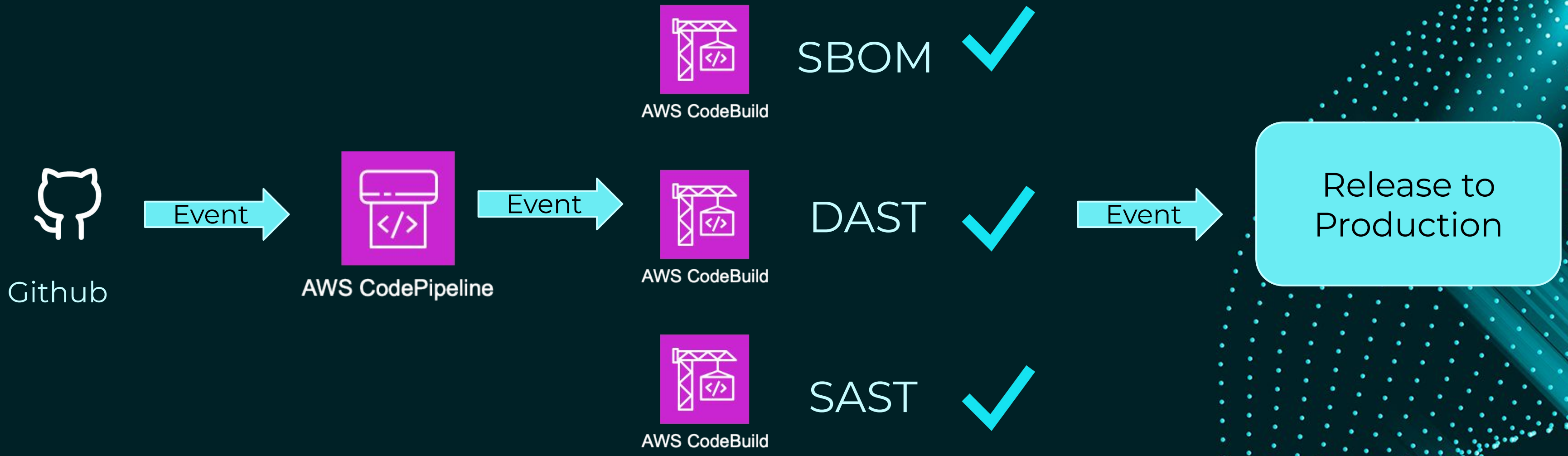
Blind Spots

Most engineers don’t get metrics from security tooling

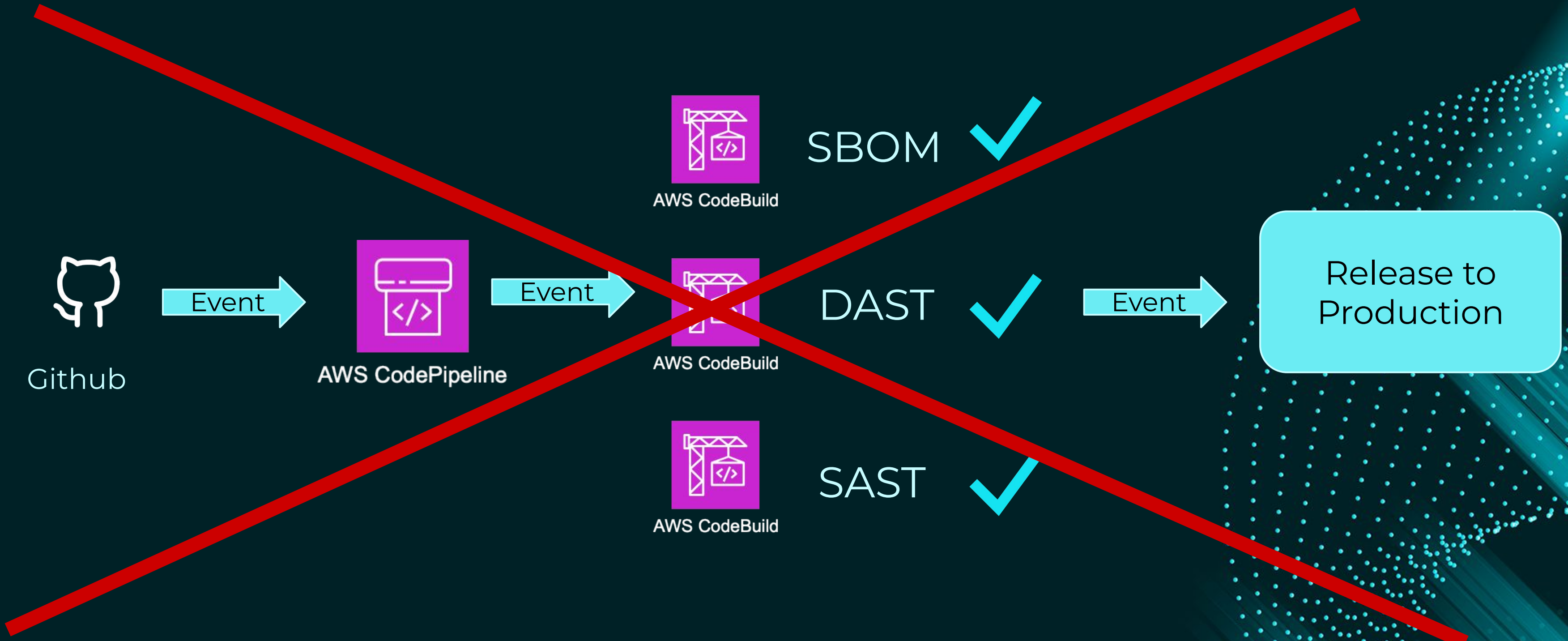
Silos

Communication and empathy still needs work

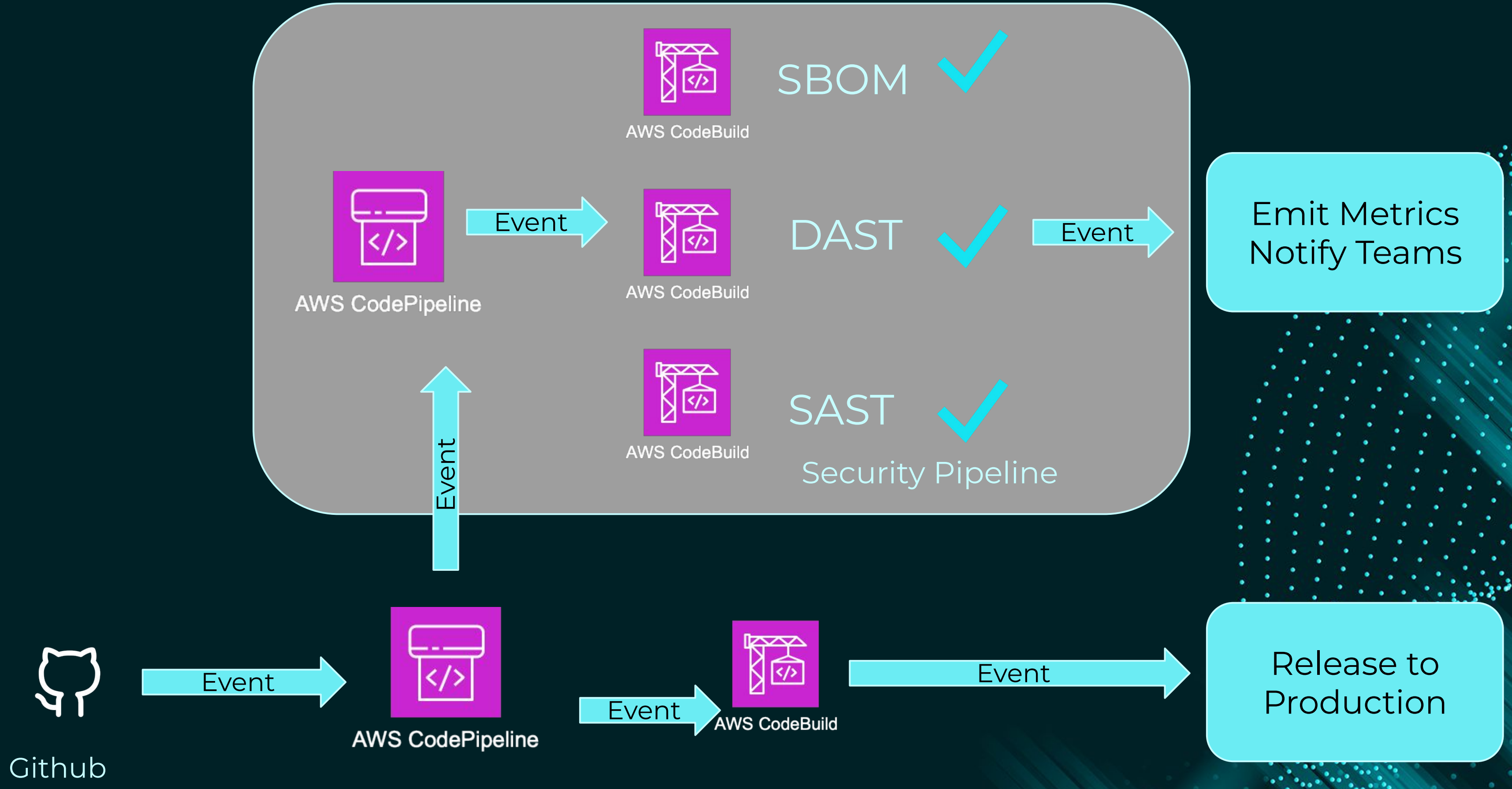
Putting it all together



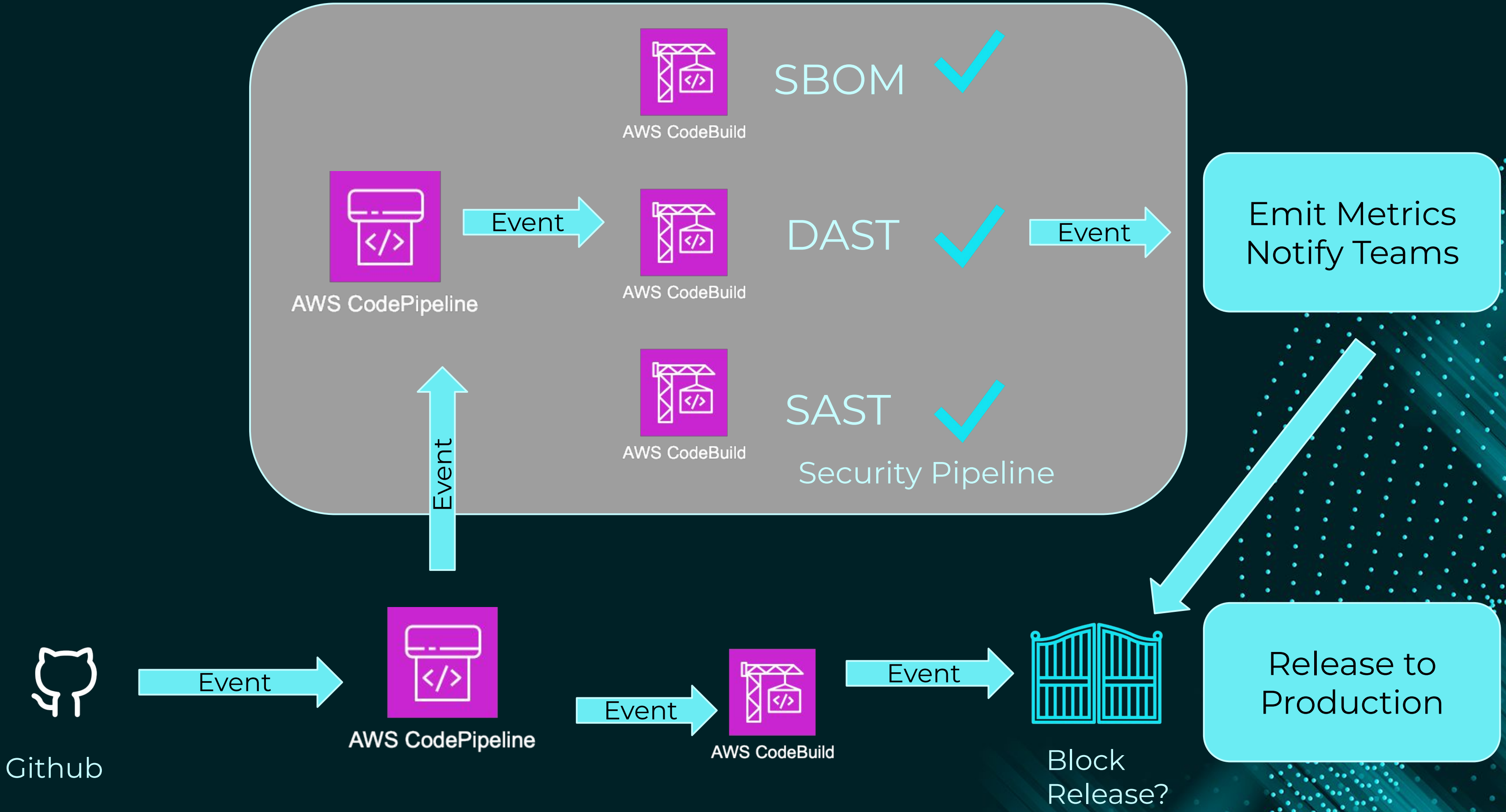
Putting it all together



Putting it all together



Putting it all together



Rules of the AppSec Pipeline

1. Tight feedback loops to teams
   a. Slack
   b. Pull request comments
   c. Commercial Product

2. Guidance not gates by default

3. Gates when failure is not an option as defined by risk assessment

4. Emit metrics as every stage to define maturity

Good and bad metrics

Good:

Number of vulns by criticality

Time to resolve vulns

Ignored vulns

Good and bad metrics

Less Good:

Average age of bug

Oldest vuln still in production

Total number of findings

Add more tools as desired

Terraform Linter - https://github.com/terraform-linters/tflint

IAMSpy - https://github.com/WithSecureLabs/IAMSpy

CFNLint - https://github.com/aws-cloudformation/cfn-lint

If you liked this or you didn’t

Brief Survey

https://forms.gle/mTtGgd2yaqu1XnKX7

Thank you