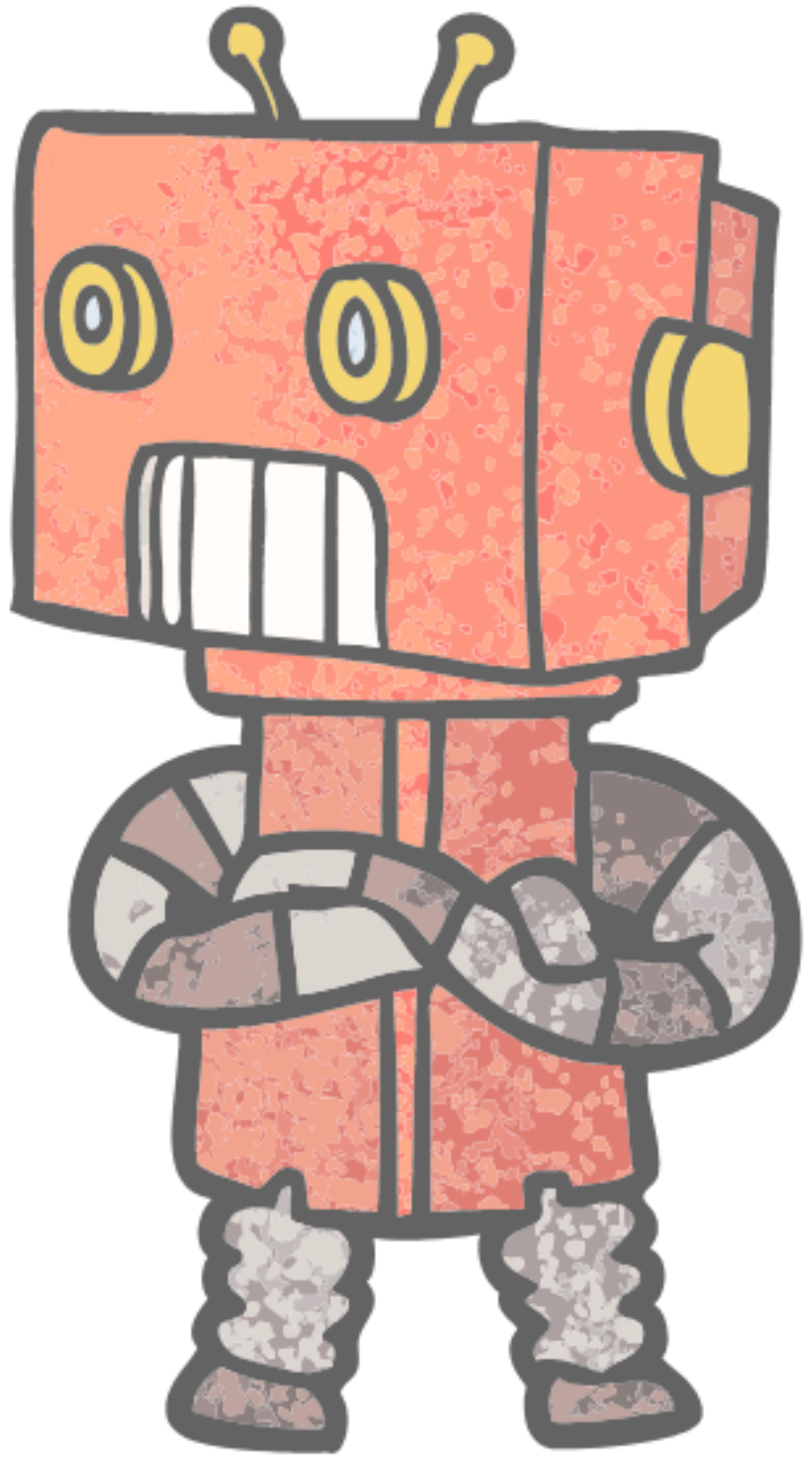
PRAGMATIC SECURITY FOR SRE

VERICA

@WICKETT

@WICKETT

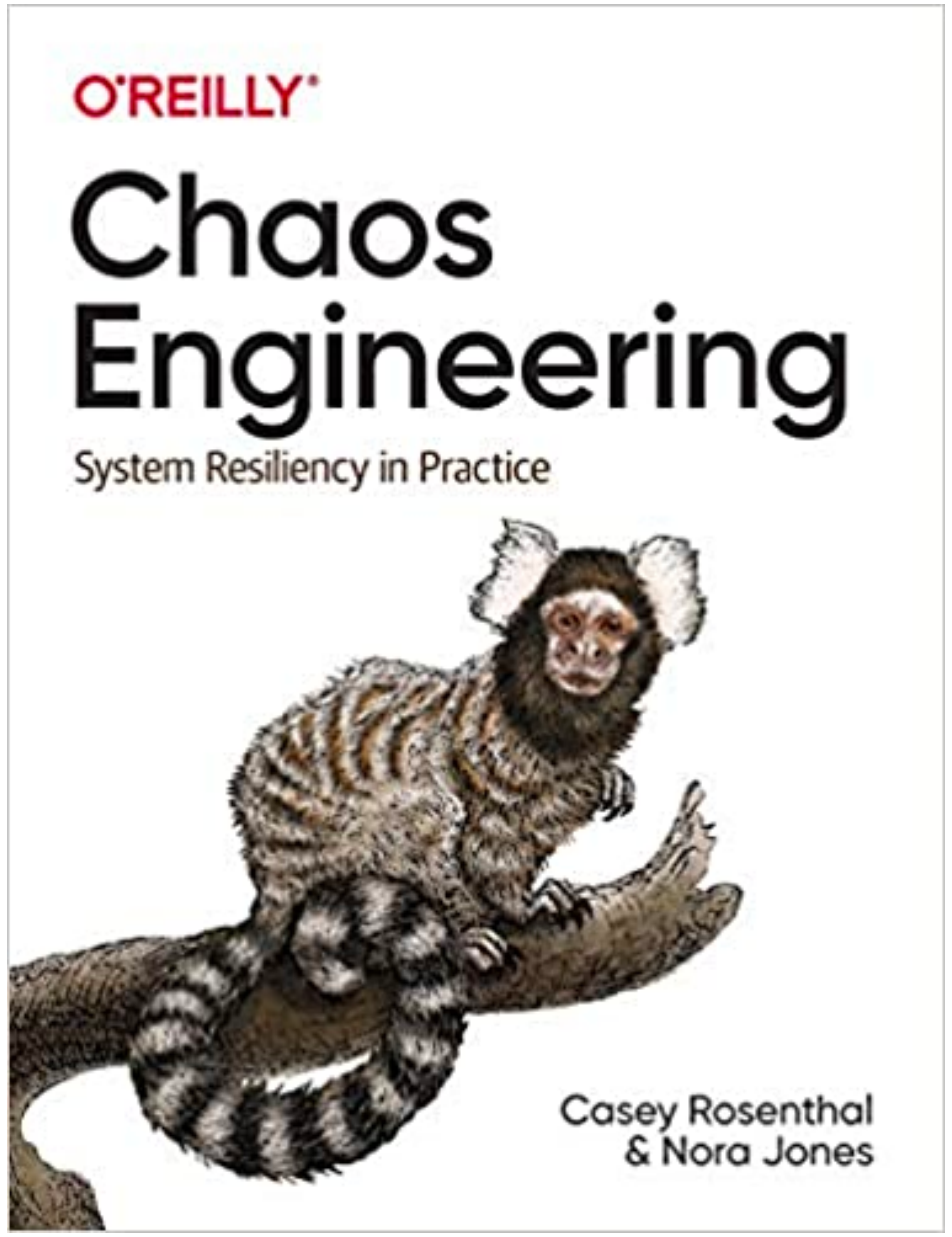
- Head of Research @ Verica
- Org of DevOpsDays Austin
- Org of DevSecOpsDays Austin
- LinkedIn Learning author on DevOps and Security Courses http://lnkd.in/JamesWickett
- Find me at wickett.me



VERICA

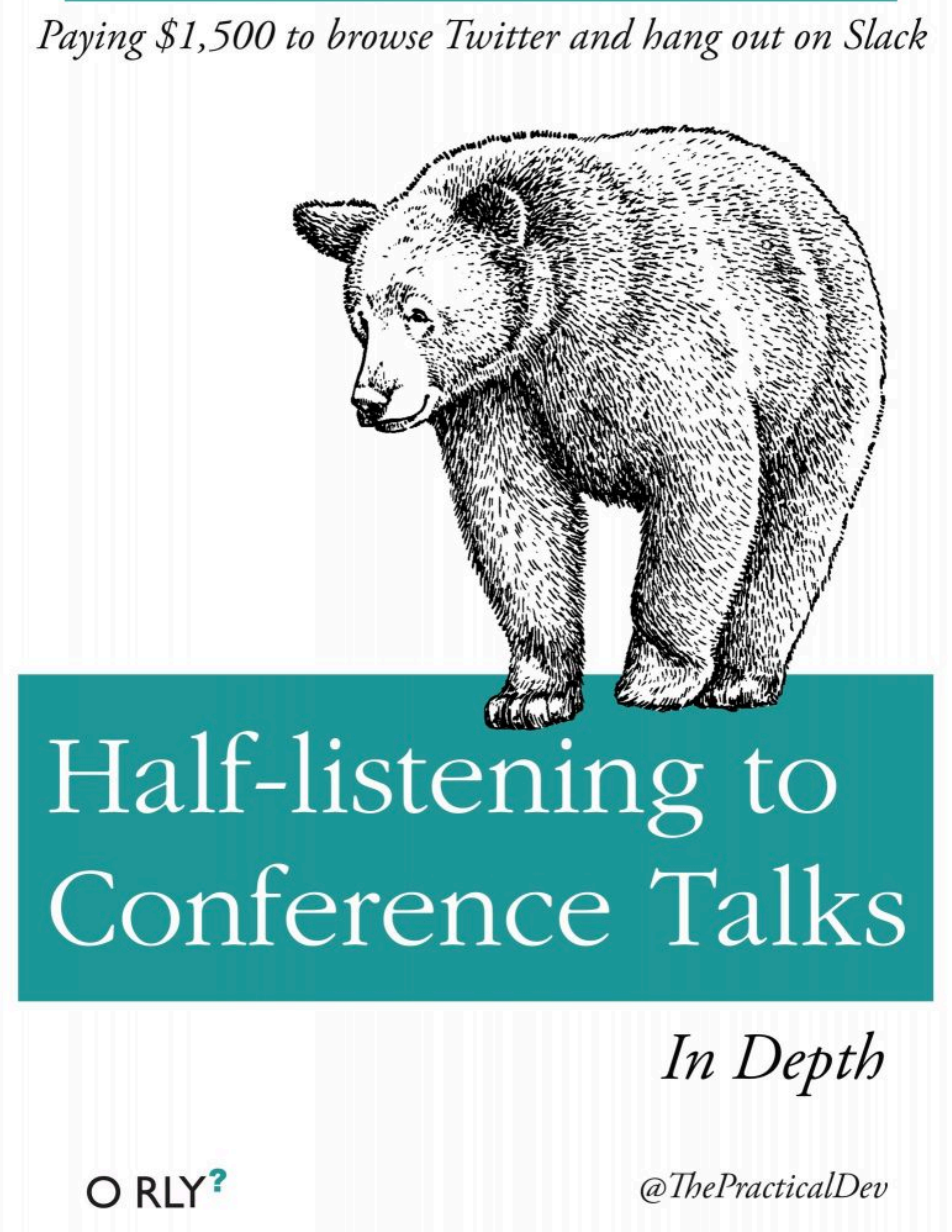
@WICKETT

Free book for attendees!
verica.io/book



VERICA

@WICKETT



Paying $1,500 to browse Twitter and hang out on Slack

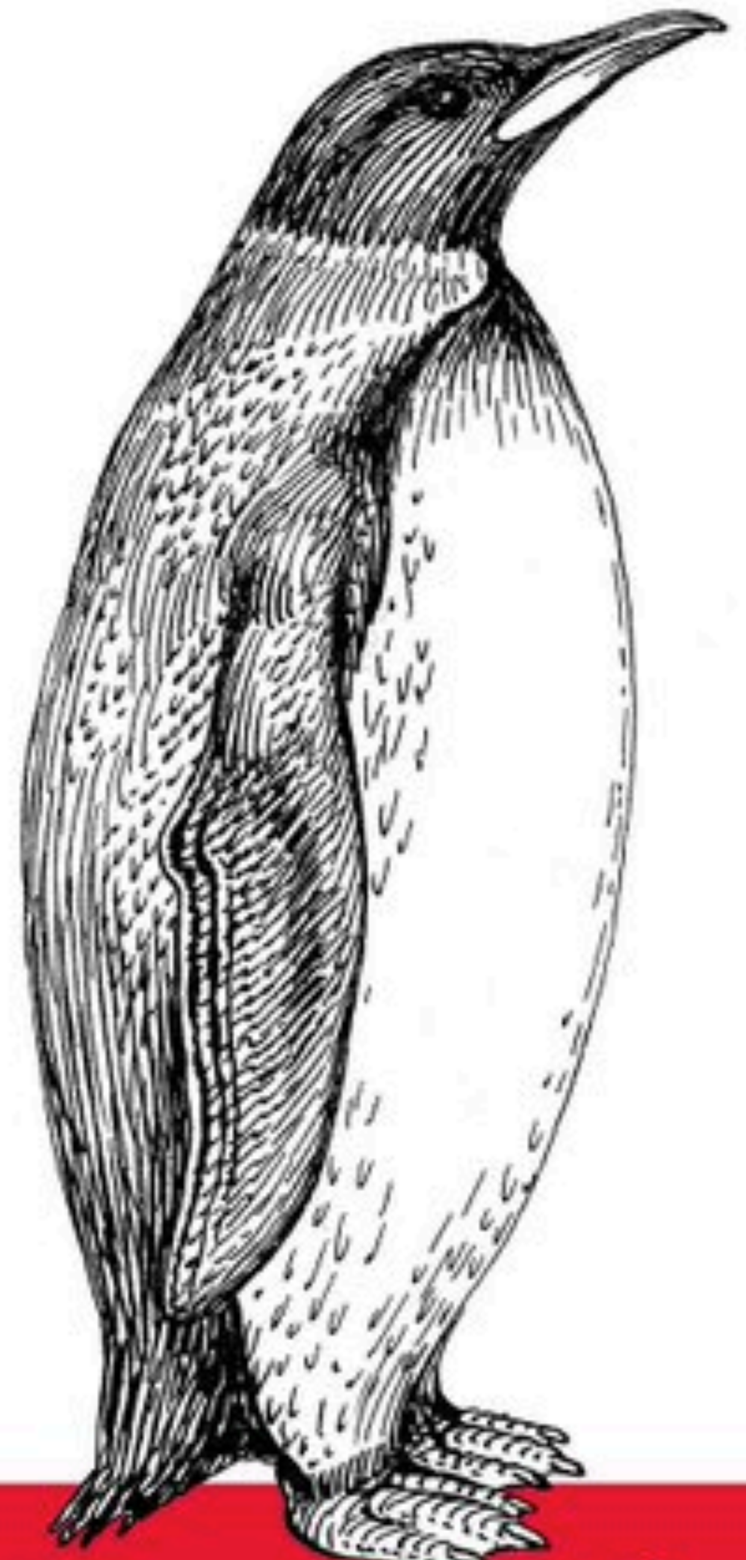
Half-listening to Conference Talks

In Depth

O RLY?

@ThePracticalDev

Letting your baby out of the nest — for better or worse



Good Enough to Ship

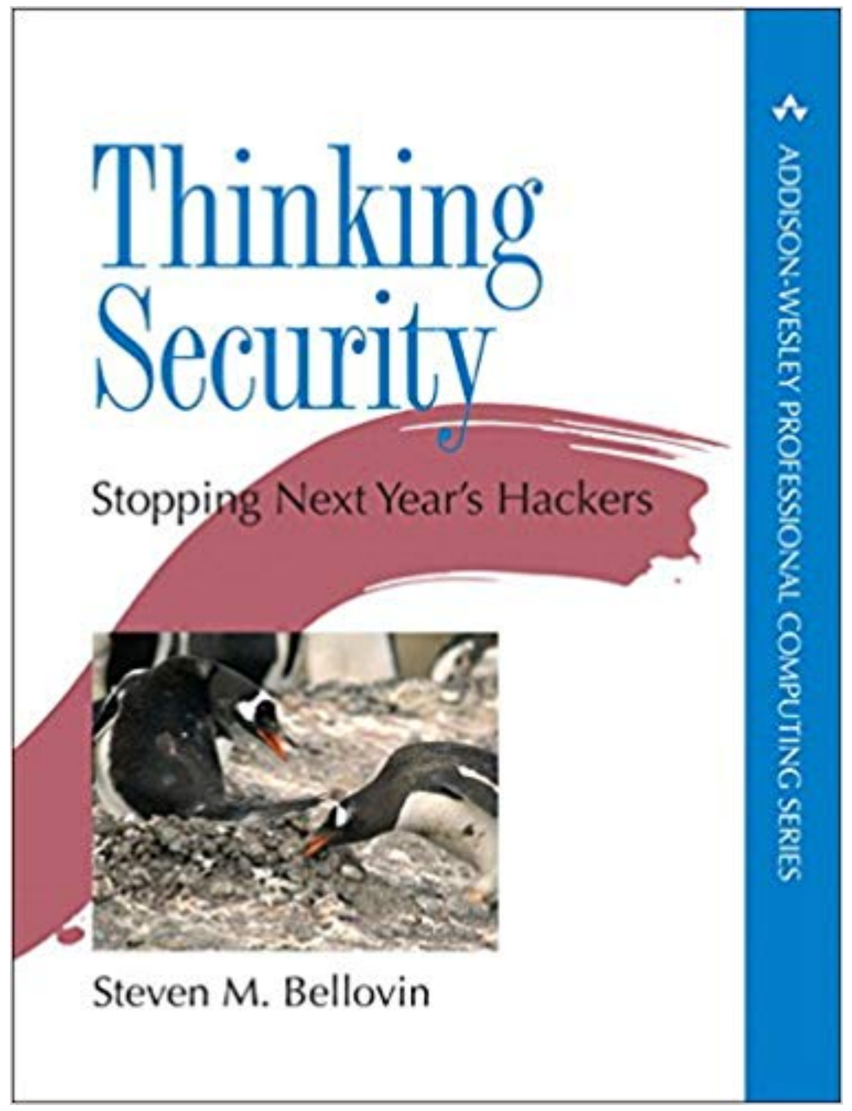
The Definitive Guide

O RLY?

@ThePracticalDev

Ah yes, security

Companies are spending a great deal on security, but we read of massive computer-related attacks. Clearly something is wrong. The root of the problem is twofold: we’re **protecting the wrong things**, and **we’re hurting productivity** in the process.

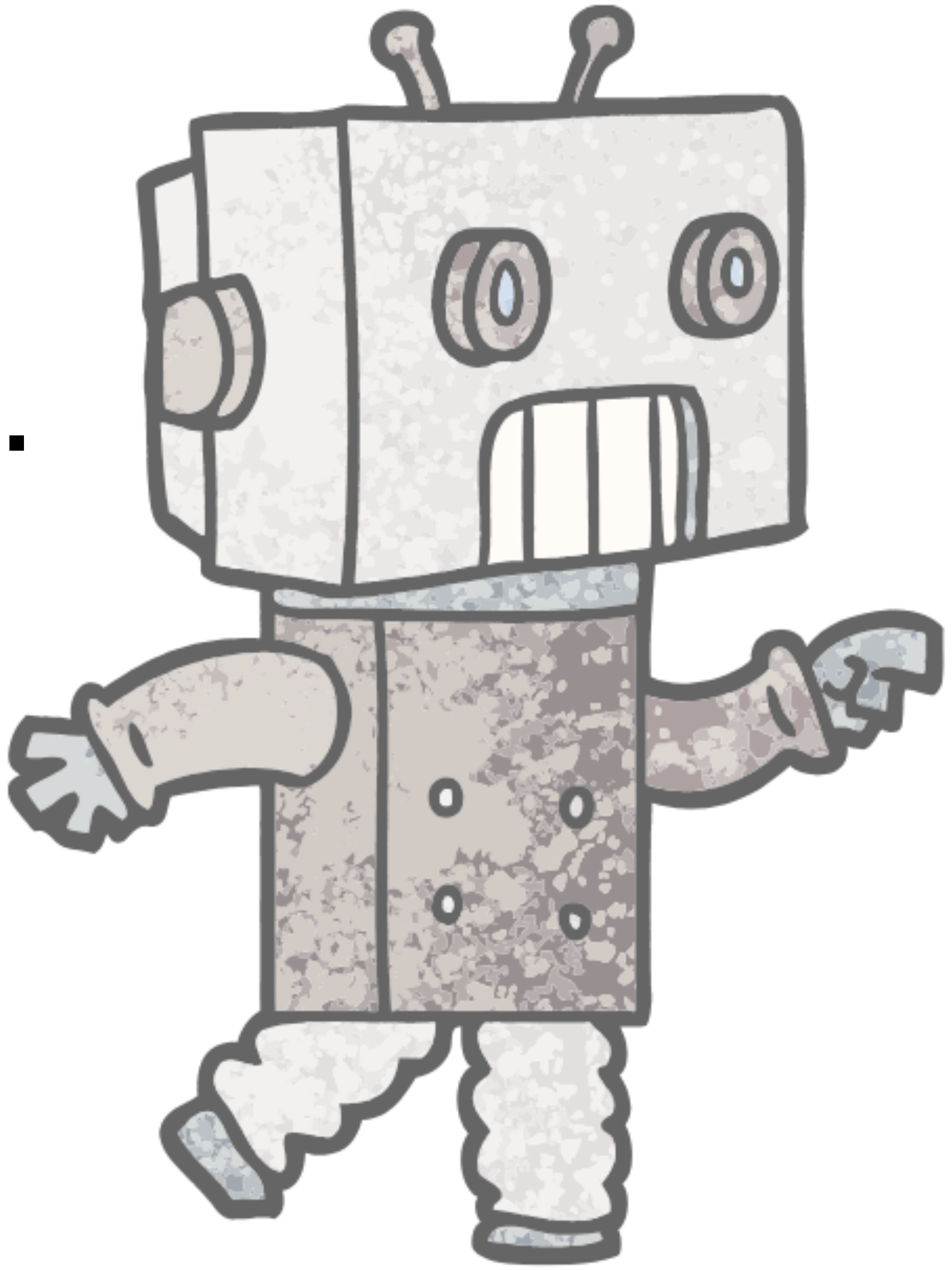


Thinking Security

Stopping Next Year's Hackers

Steven M. Bellovin

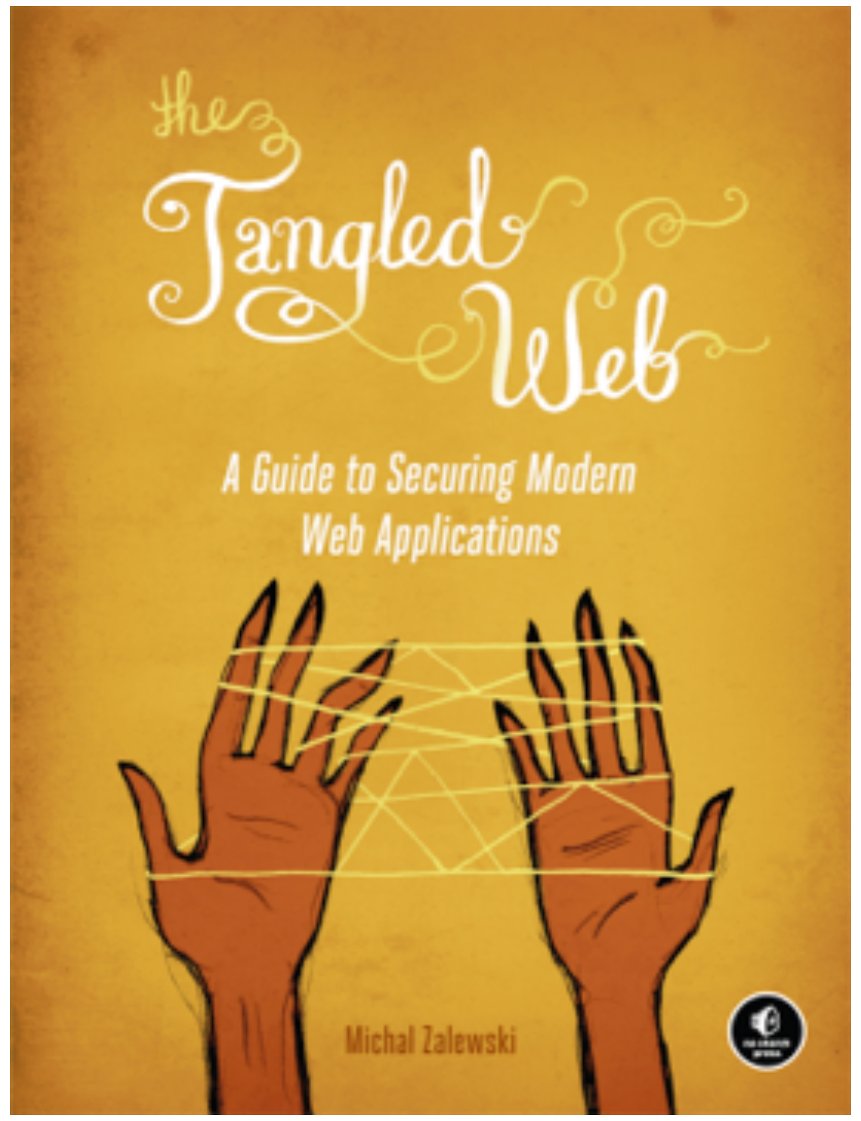
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES



VERICA

@WICKETT

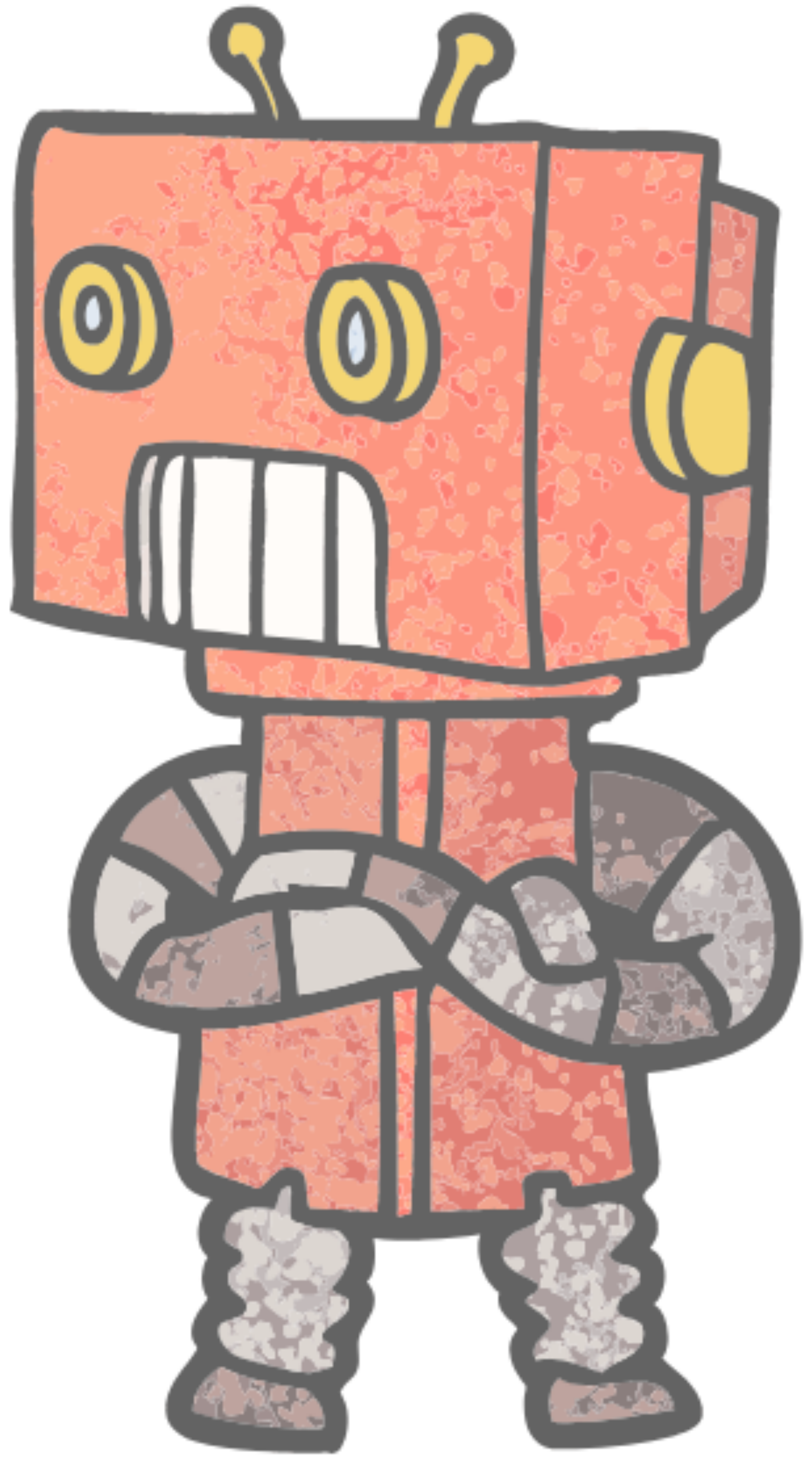
[Security by risk assessment] introduces a dangerous fallacy: that structured inadequacy is almost as good as adequacy and that underfunded security efforts plus risk management are about as good as properly funded security work



the Tangled Web

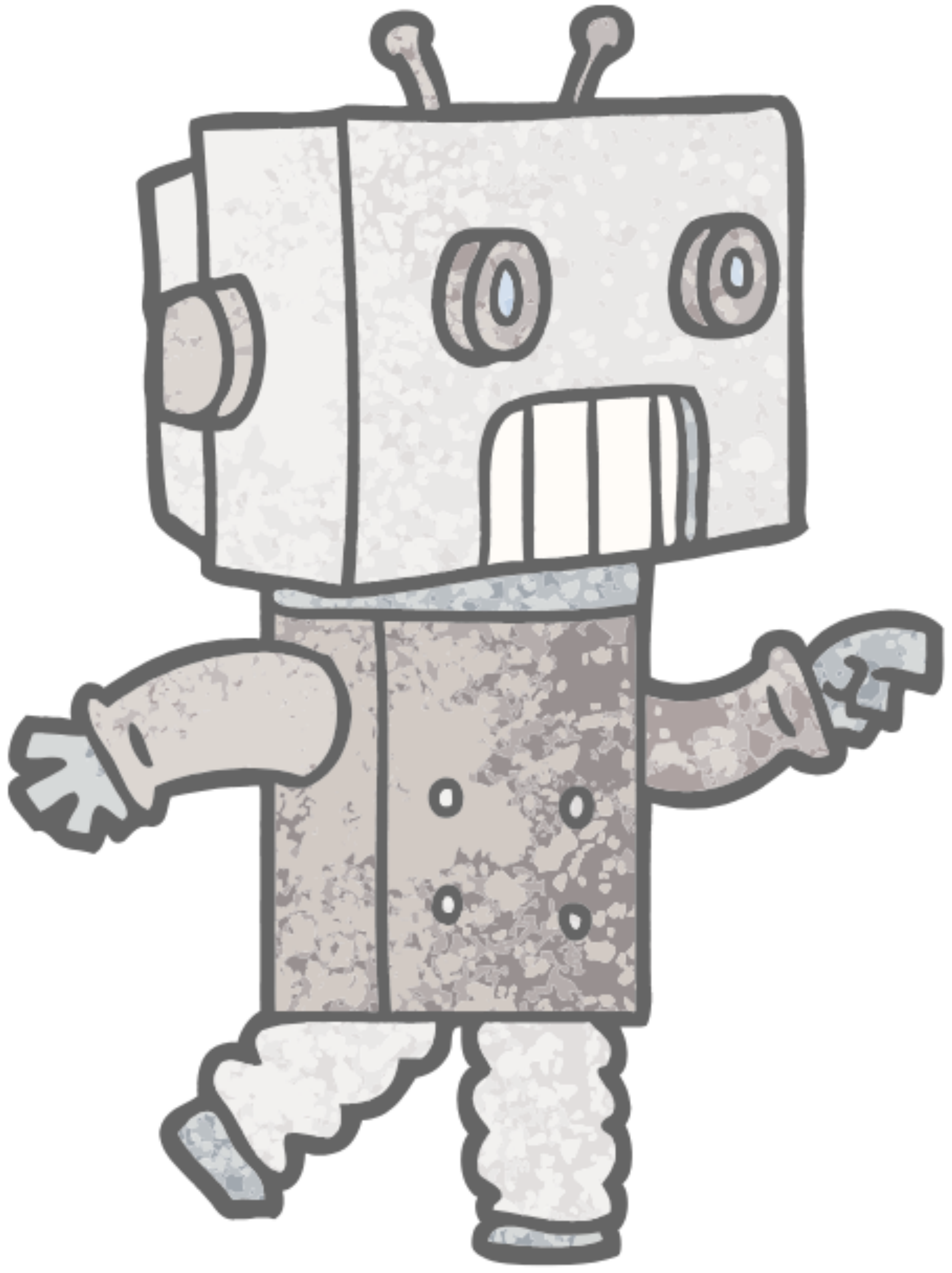
A Guide to Securing Modern Web Applications

Michal Zalewski



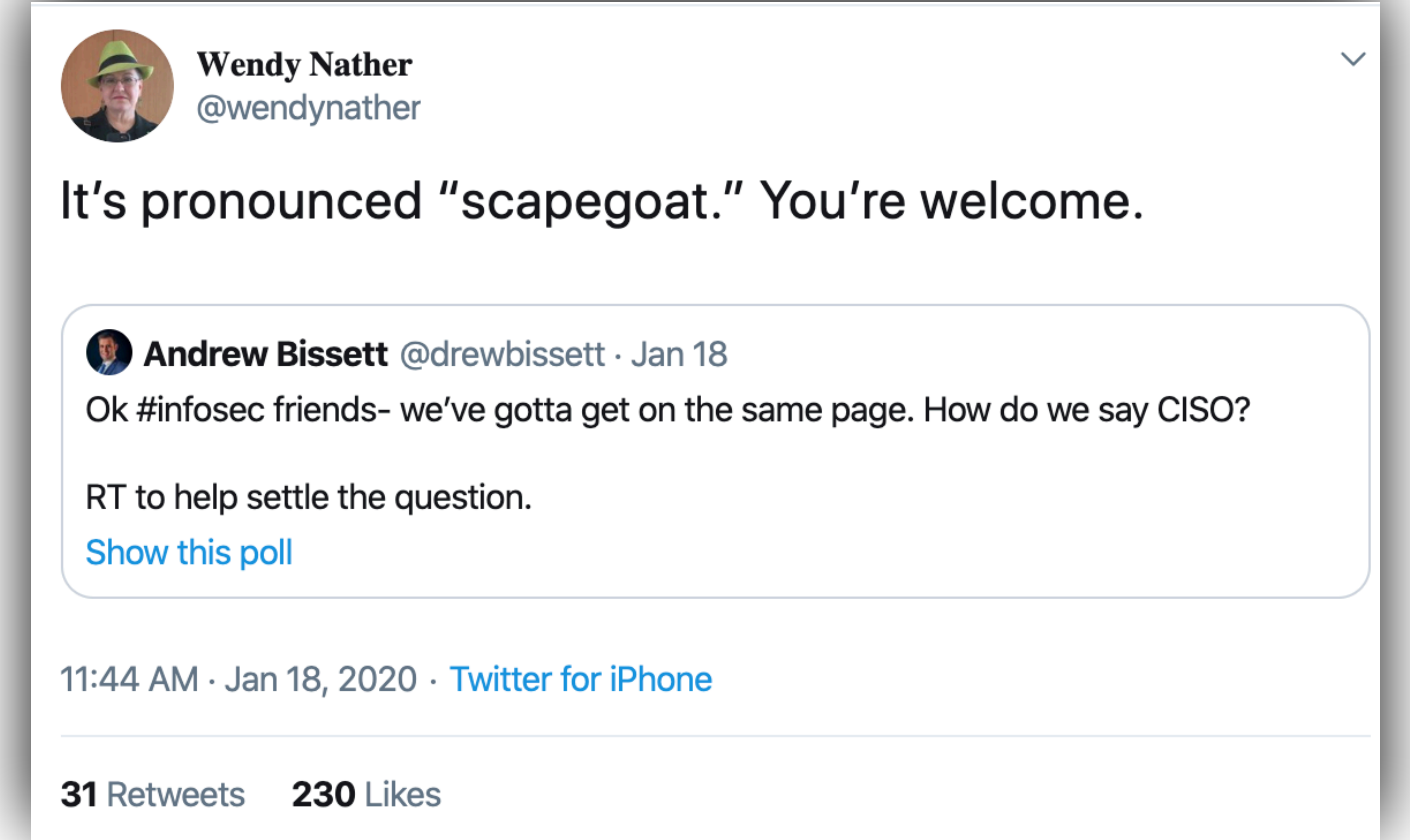
many security teams work with a worldview where their goal is to inhibit change as much as possible

O'REILLY
Agile Application Security
ENABLING SECURITY IN A CONTINUOUS DELIVERY PIPELINE
Laura Bell, Michael Brunton-Spall,
Rich Smith & Jim Bird



VERICA

@WICKETT



Wendy Nather
@wendynather

It’s pronounced “scapegoat.” You’re welcome.

Andrew Bissett @drewbissett · Jan 18

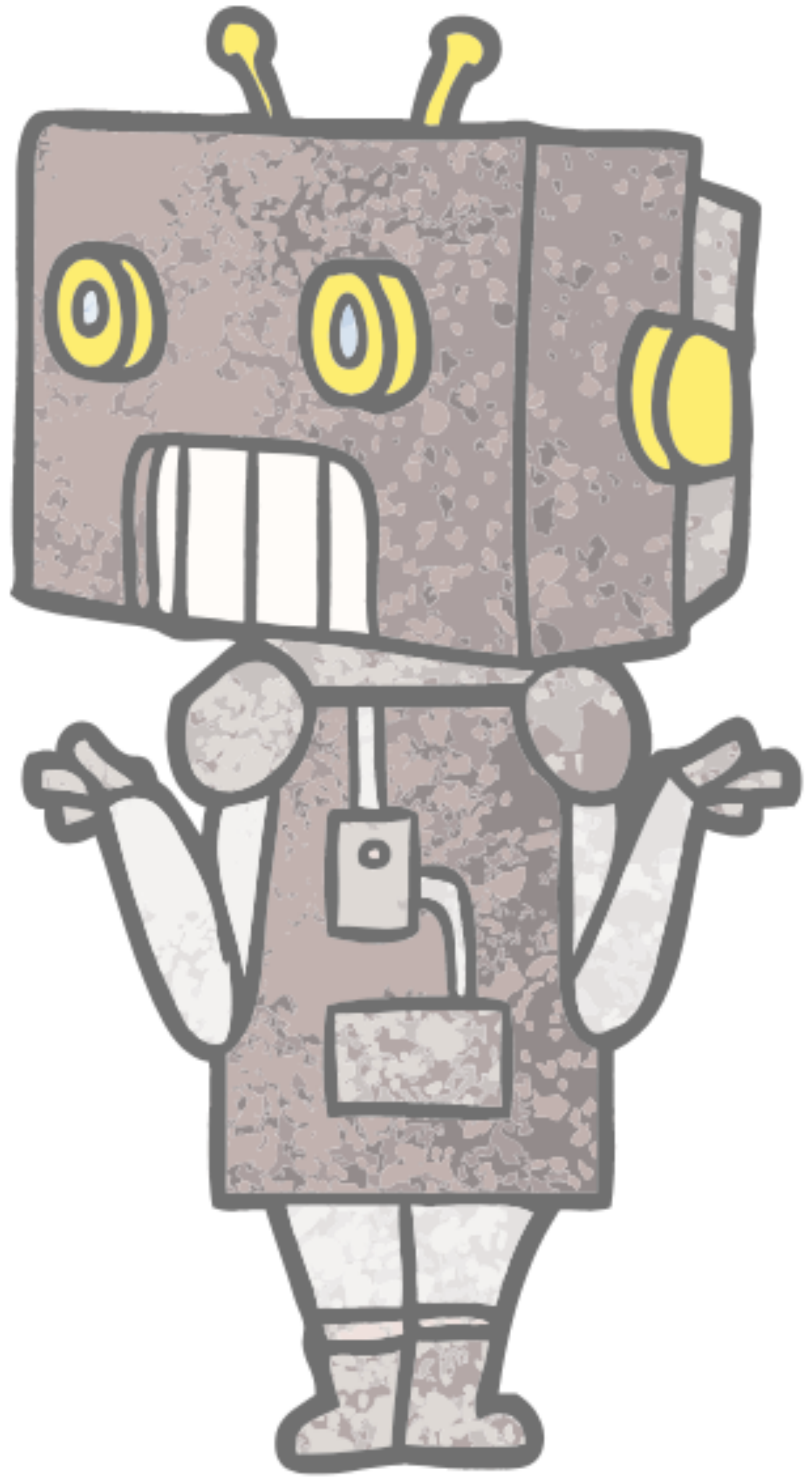
Ok #infosec friends- we’ve gotta get on the same page. How do we say CISO?

RT to help settle the question.

Show this poll

11:44 AM · Jan 18, 2020 · Twitter for iPhone

31 Retweets 230 Likes



VERICA

@WICKETT

Putting off critical tasks until everyone forgets about them

Getting Around to Security Next Month

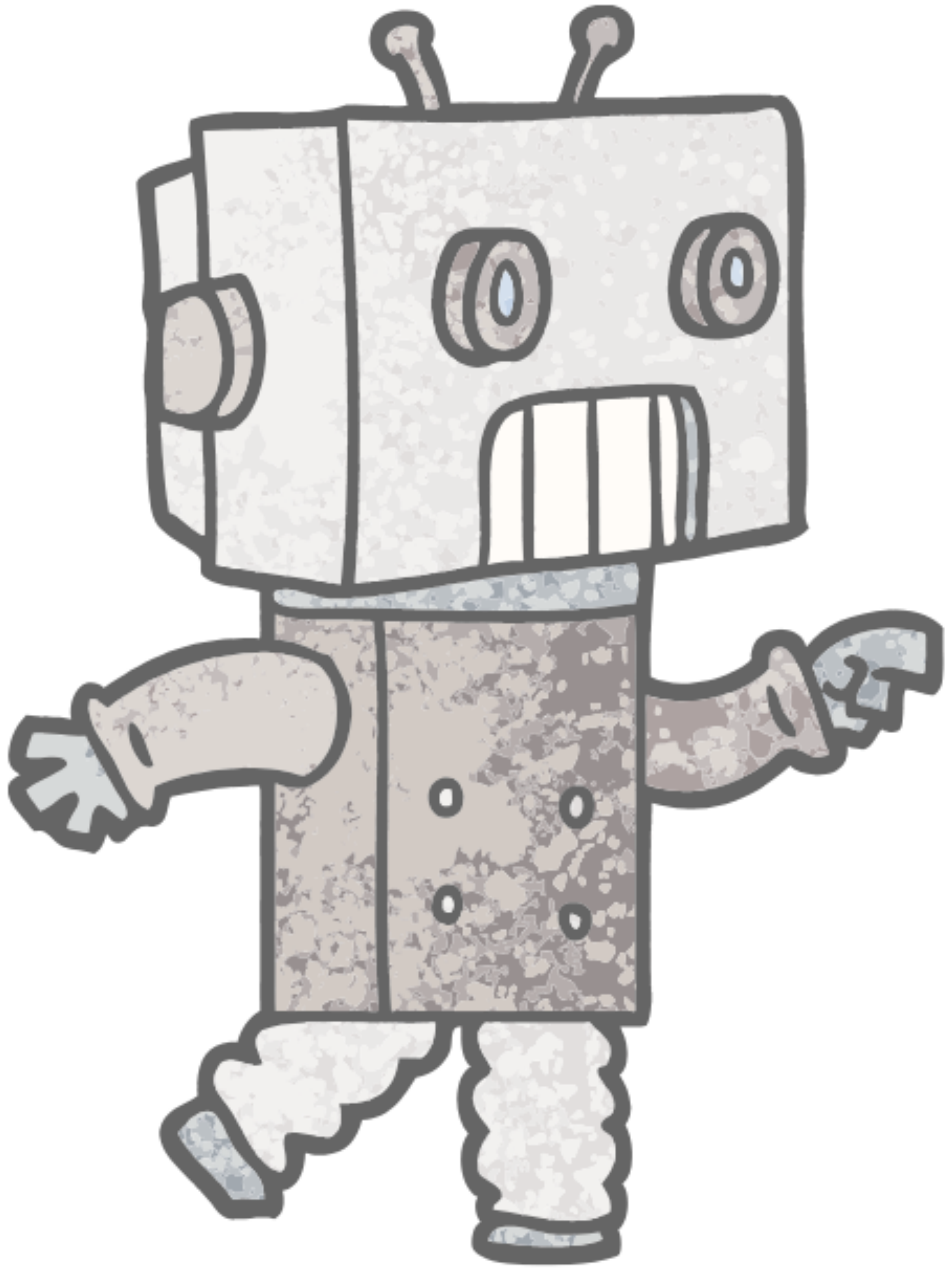
If there's time

O RLY?

@ThePracticalDev

CULTURE IS THE MOST IMPORTANT ASPECT TO DEVOPS SUCCEEDING IN THE ENTERPRISE

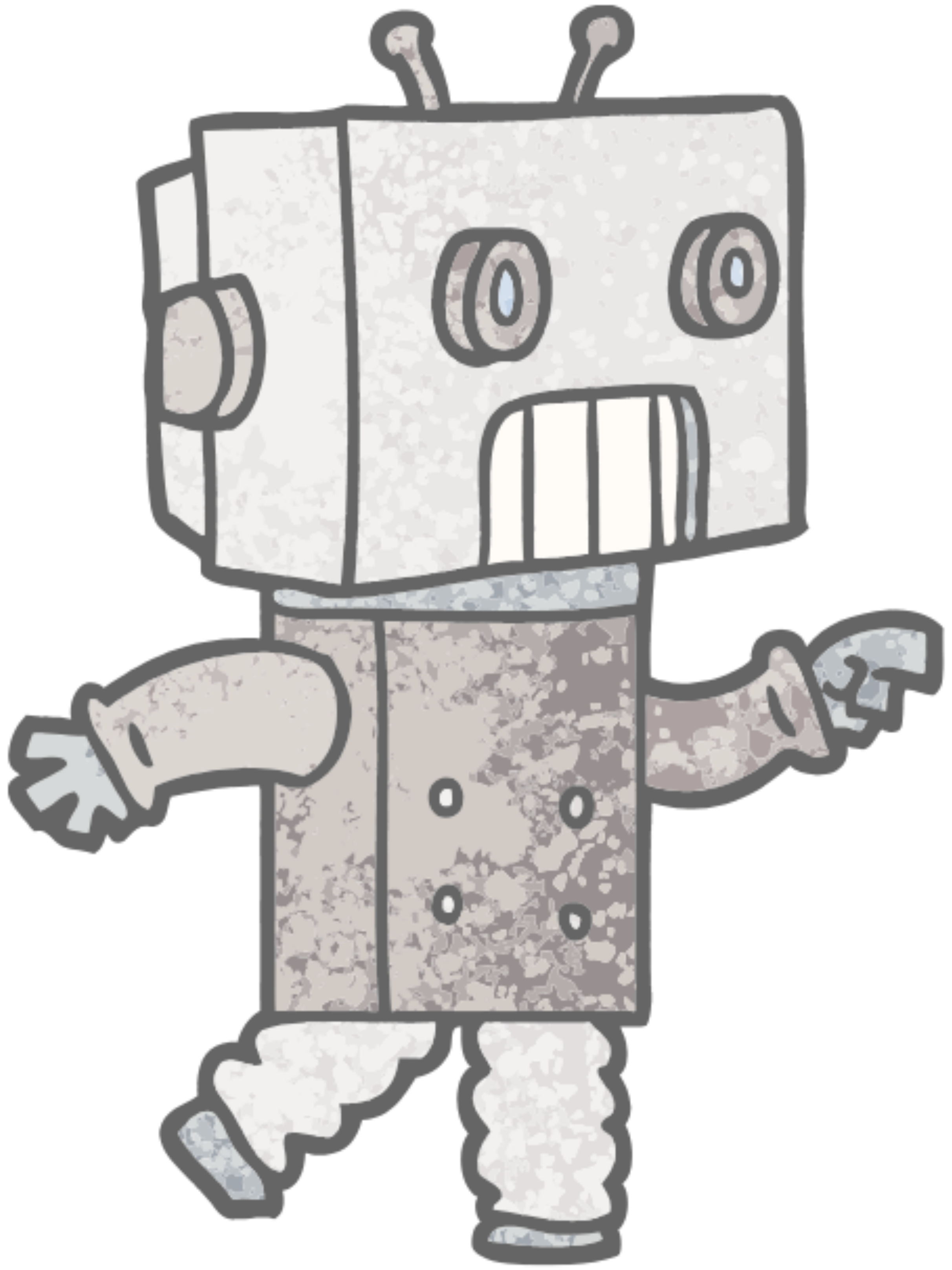
- PATRICK DEBOIS



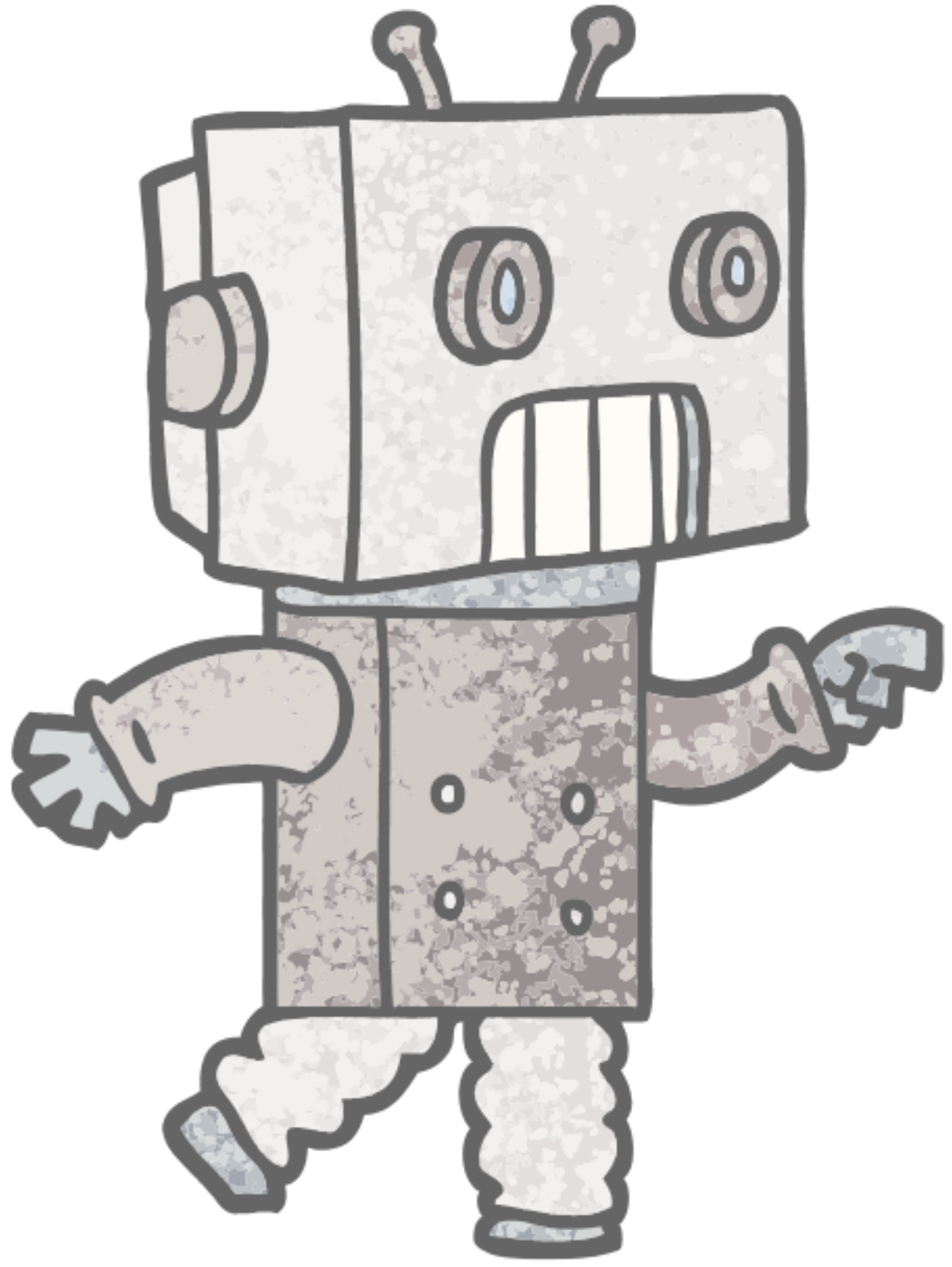
VERICA

@WICKETT

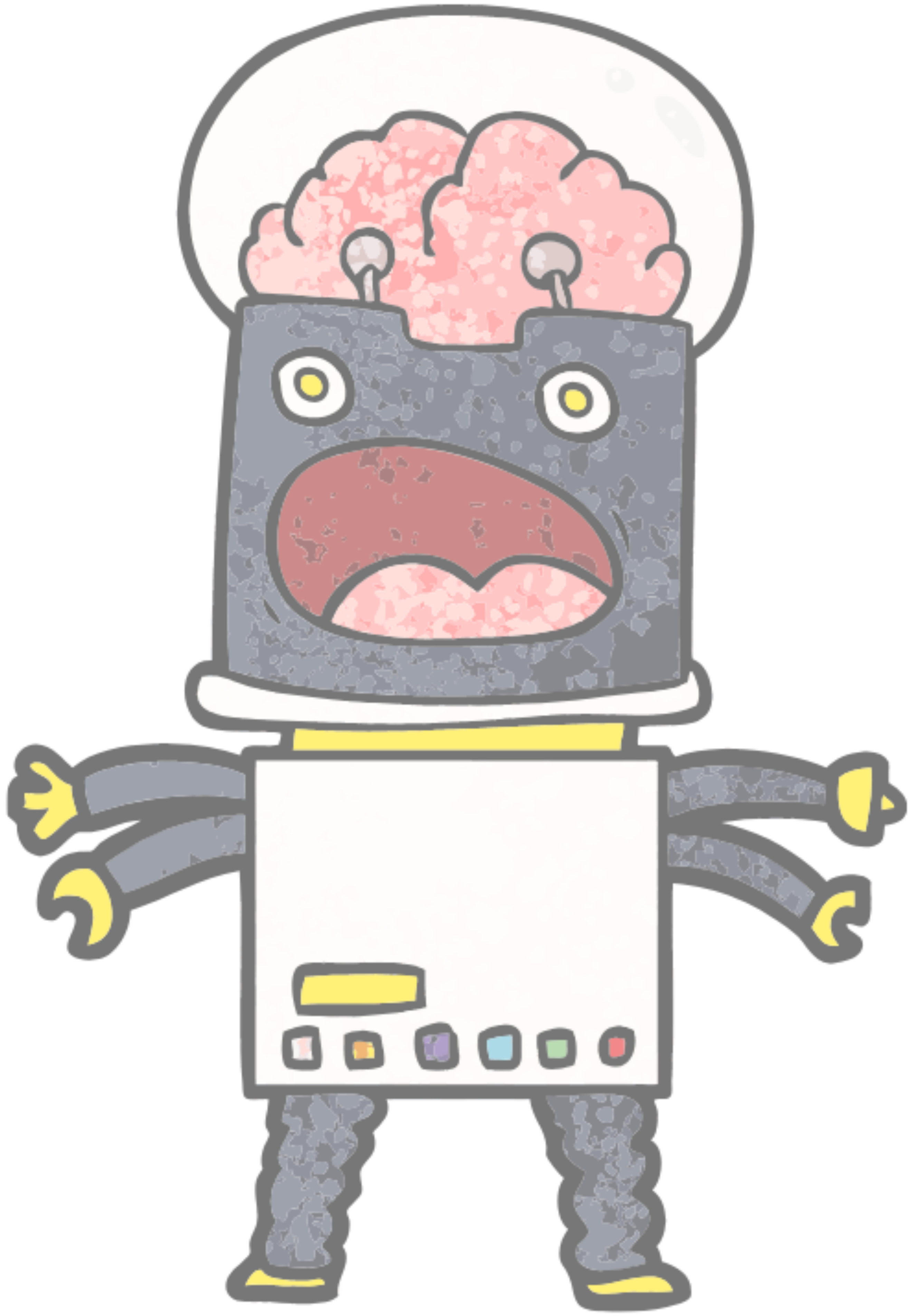
DevSecOps is a cultural movement that furthers the movements of Agile and DevOps into Security



SRE, also known as the people who actually get things done.



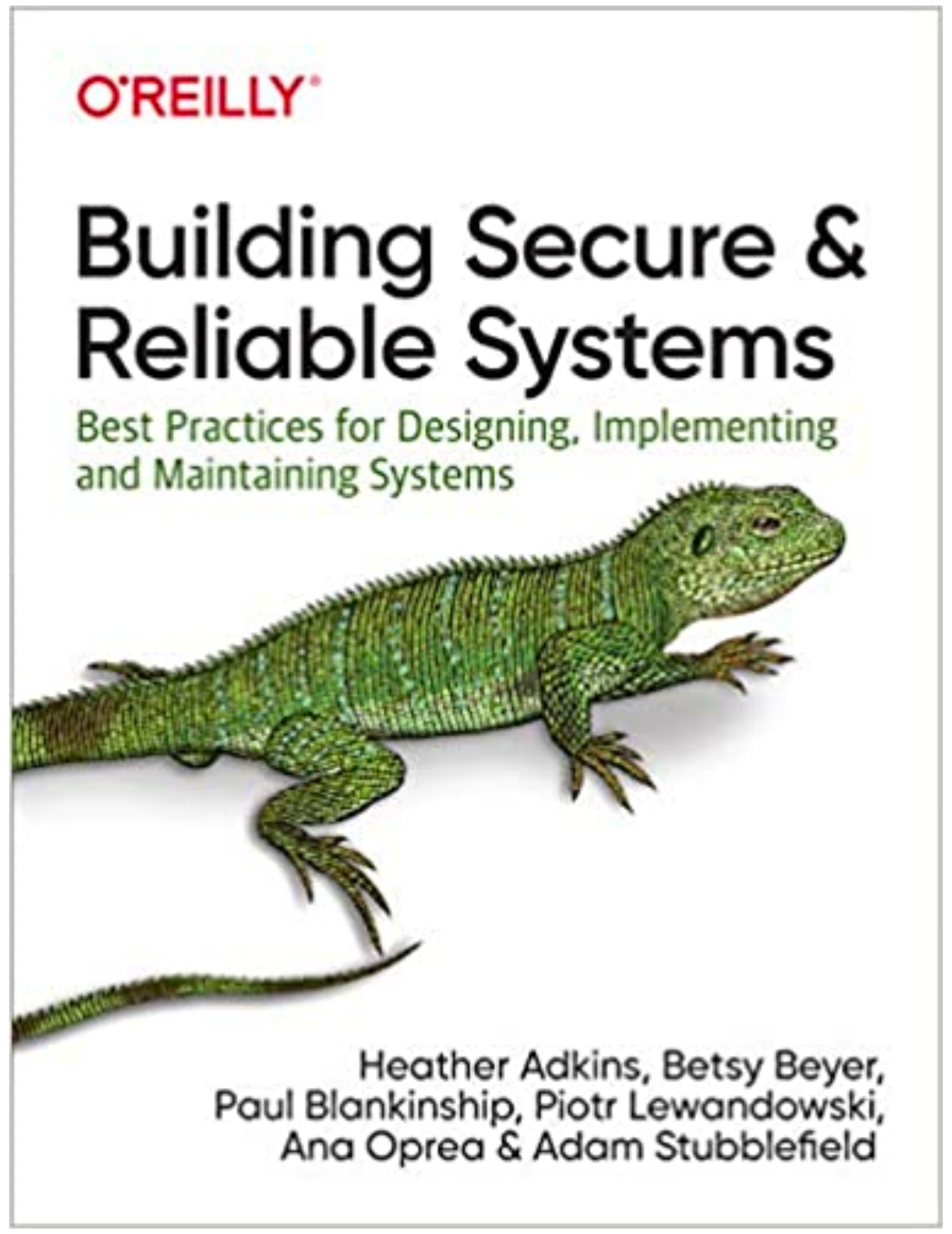
SRE TO THE RESCUE



VERICA

@WICKETT

Reliability and Security Tradeoffs

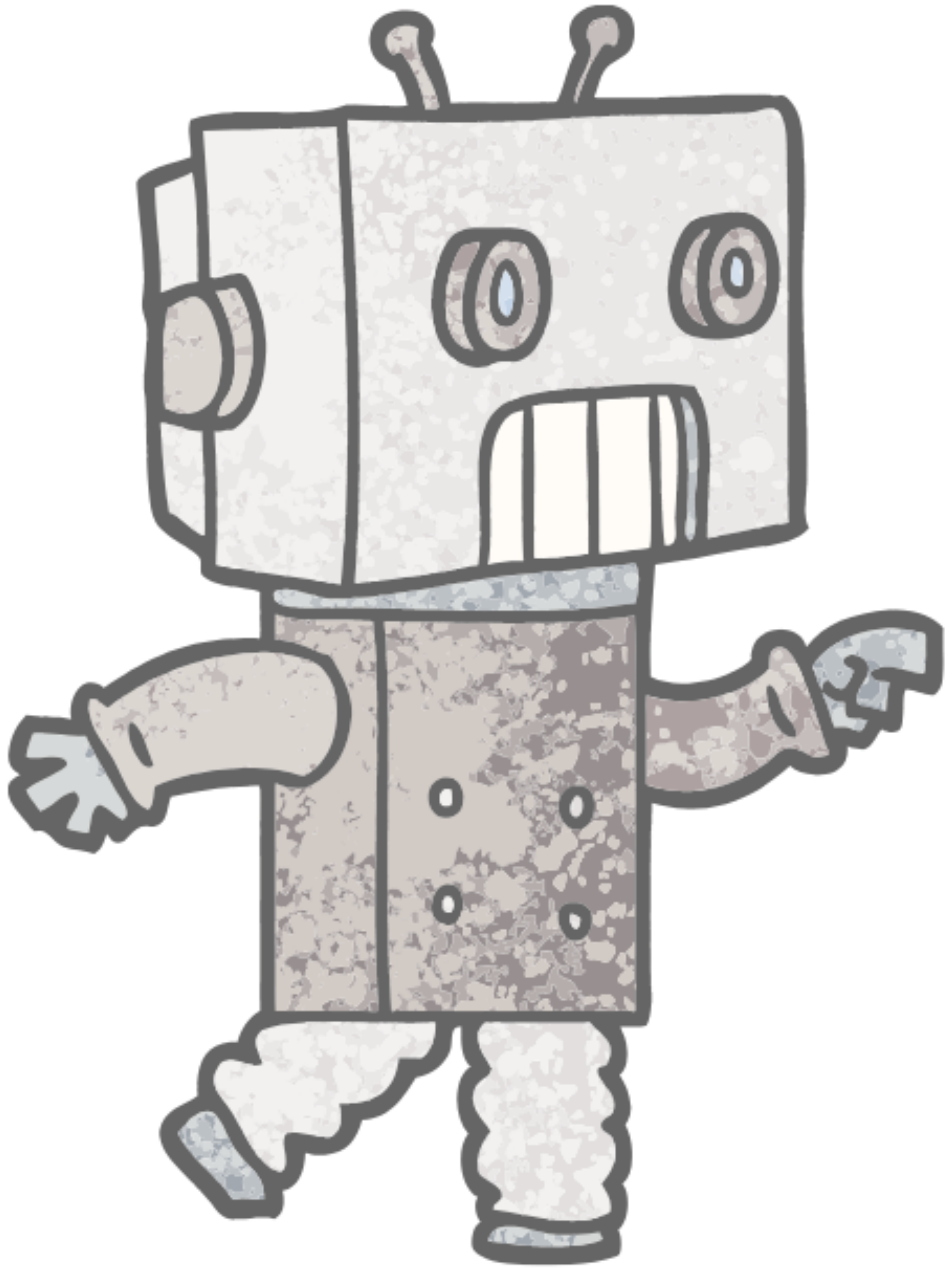


O'REILLY

Building Secure & Reliable Systems

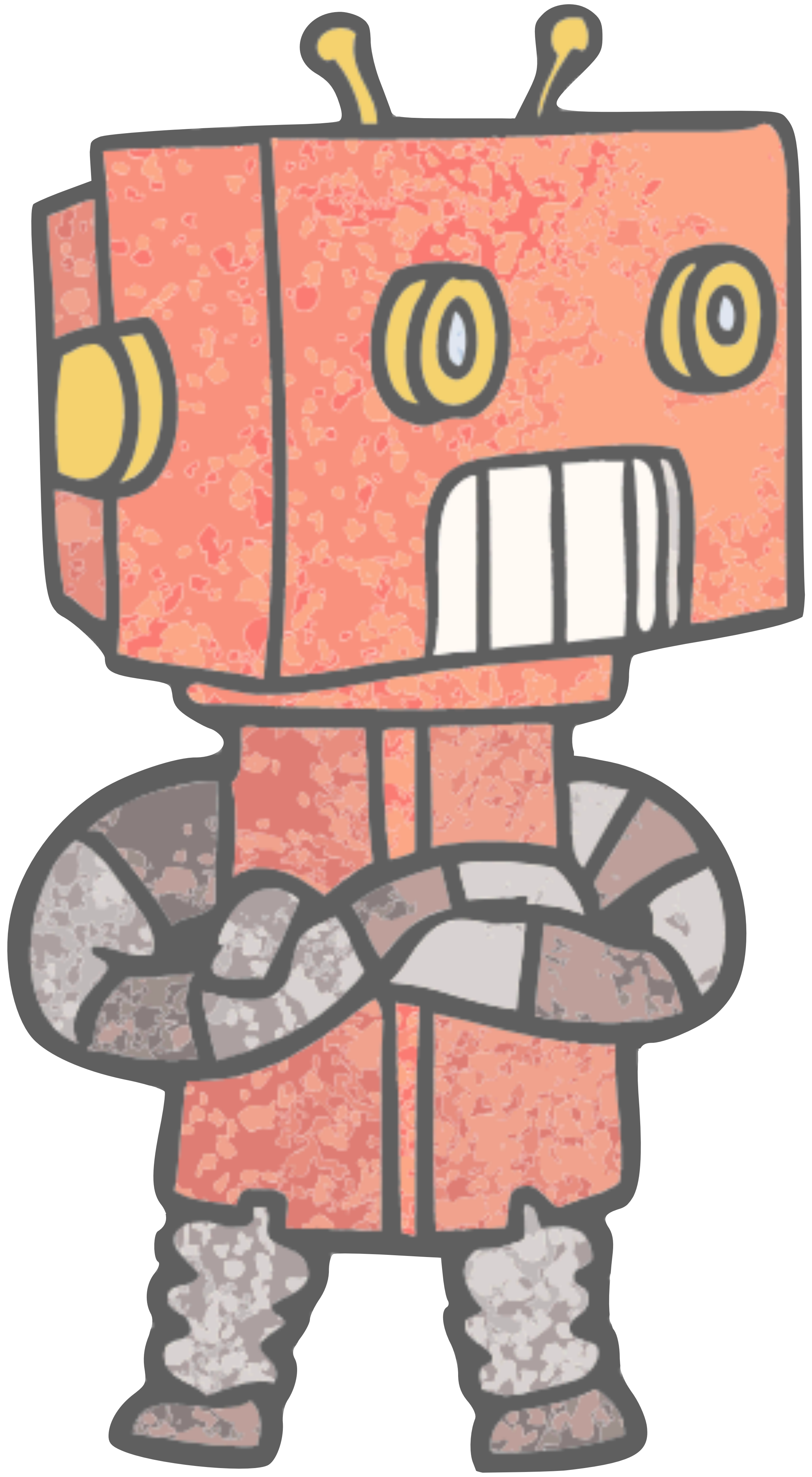
Best Practices for Designing, Implementing and Maintaining Systems

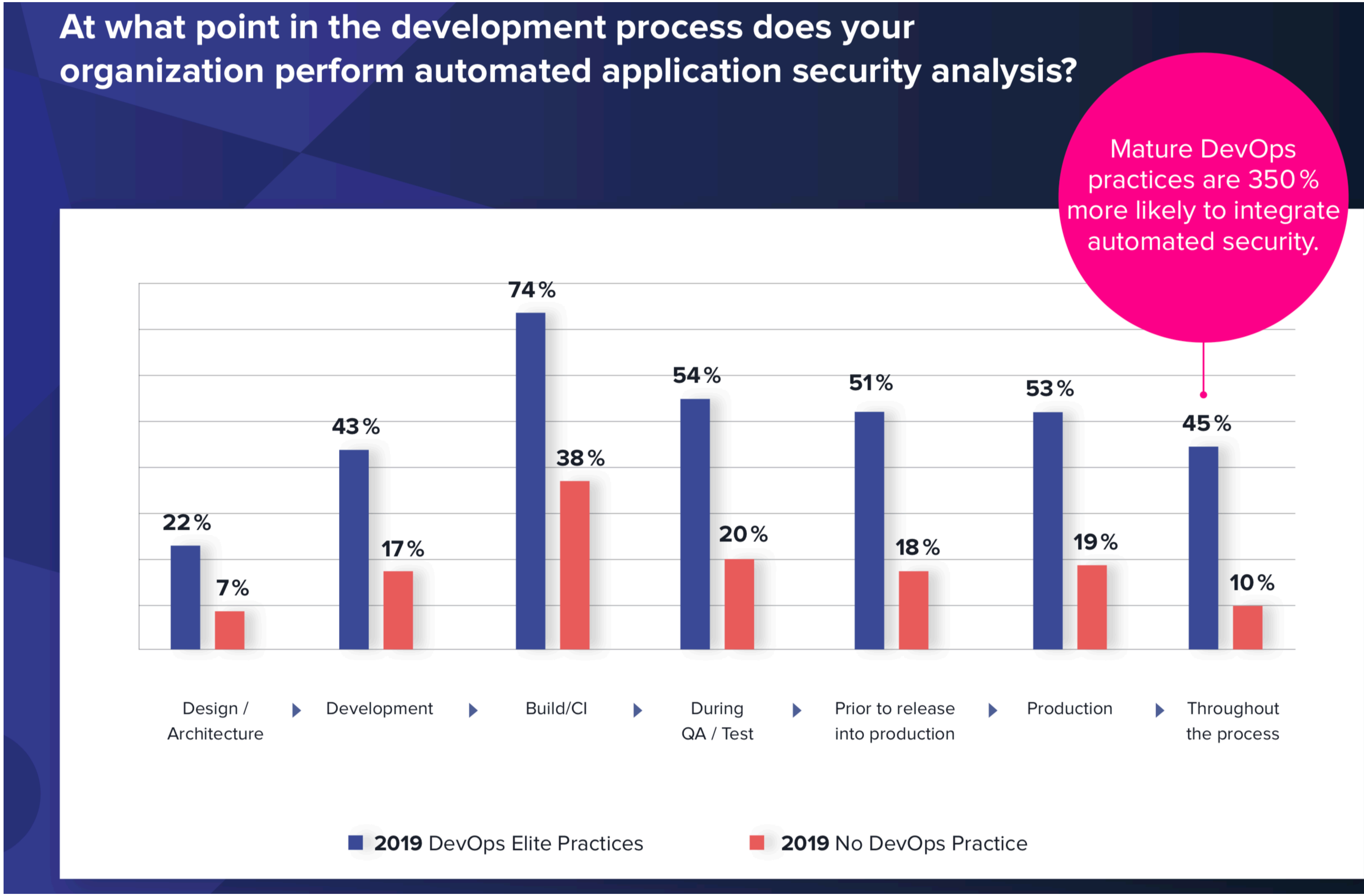
Heather Adkins, Betsy Beyer, Paul Blankinship, Piotr Lewandowski, Ana Oprea & Adam Stubblefield



VERICA

@WICKETT

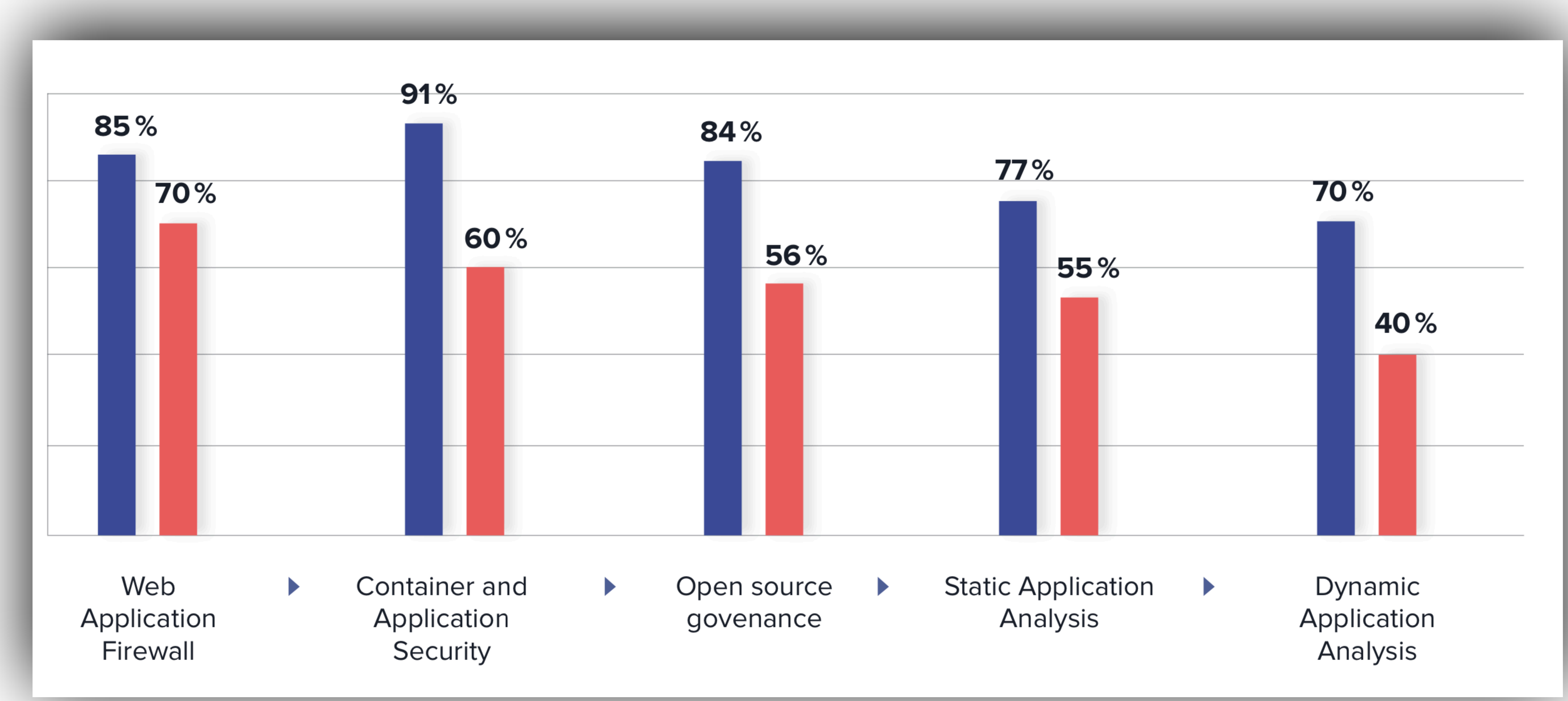




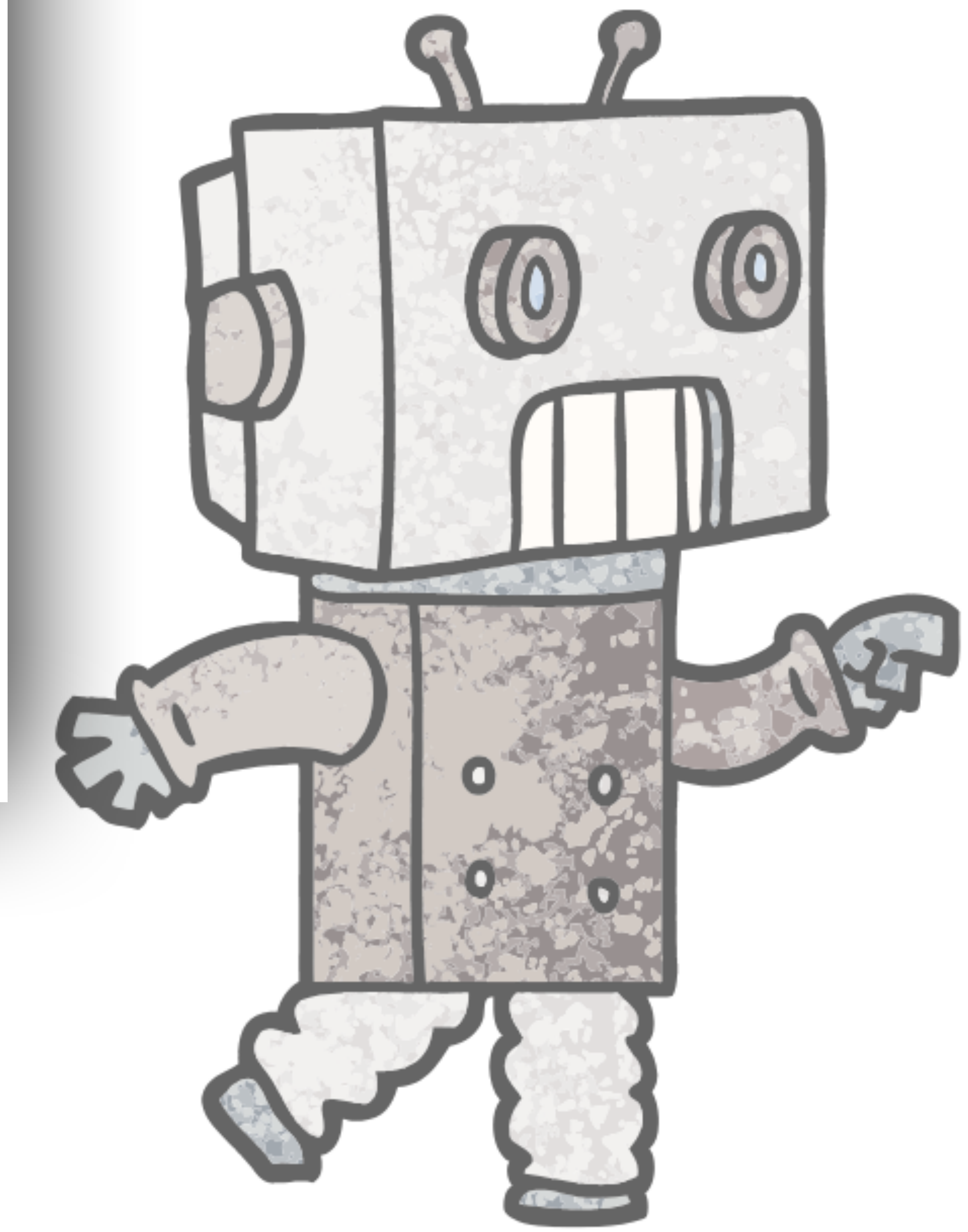
At what point in the development process does your organization perform automated application security analysis?

Mature DevOps practices are 350 % more likely to integrate automated security.

Source: 209
DevSecOps Community S

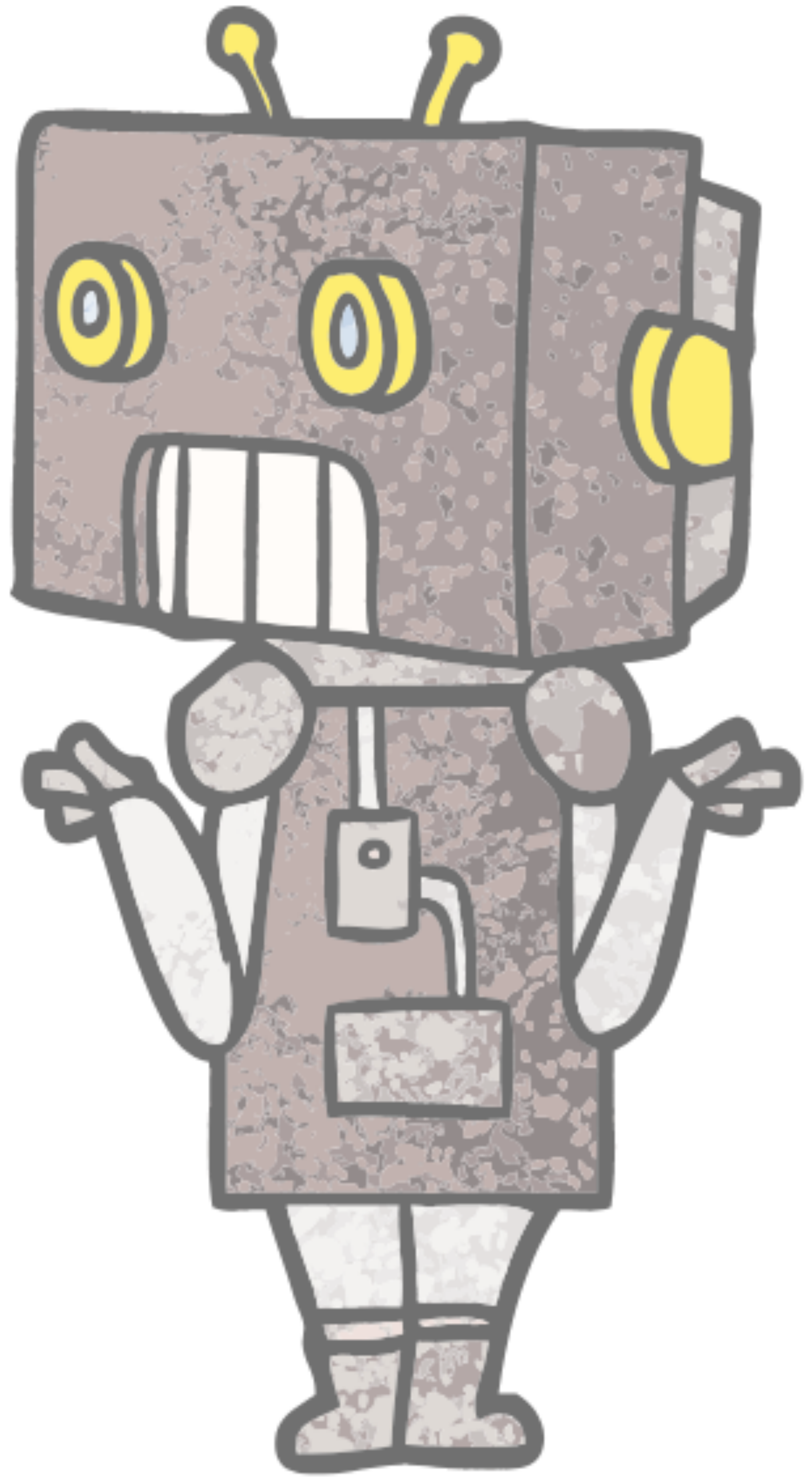


Source: 2019 DevSecOps Community Survey



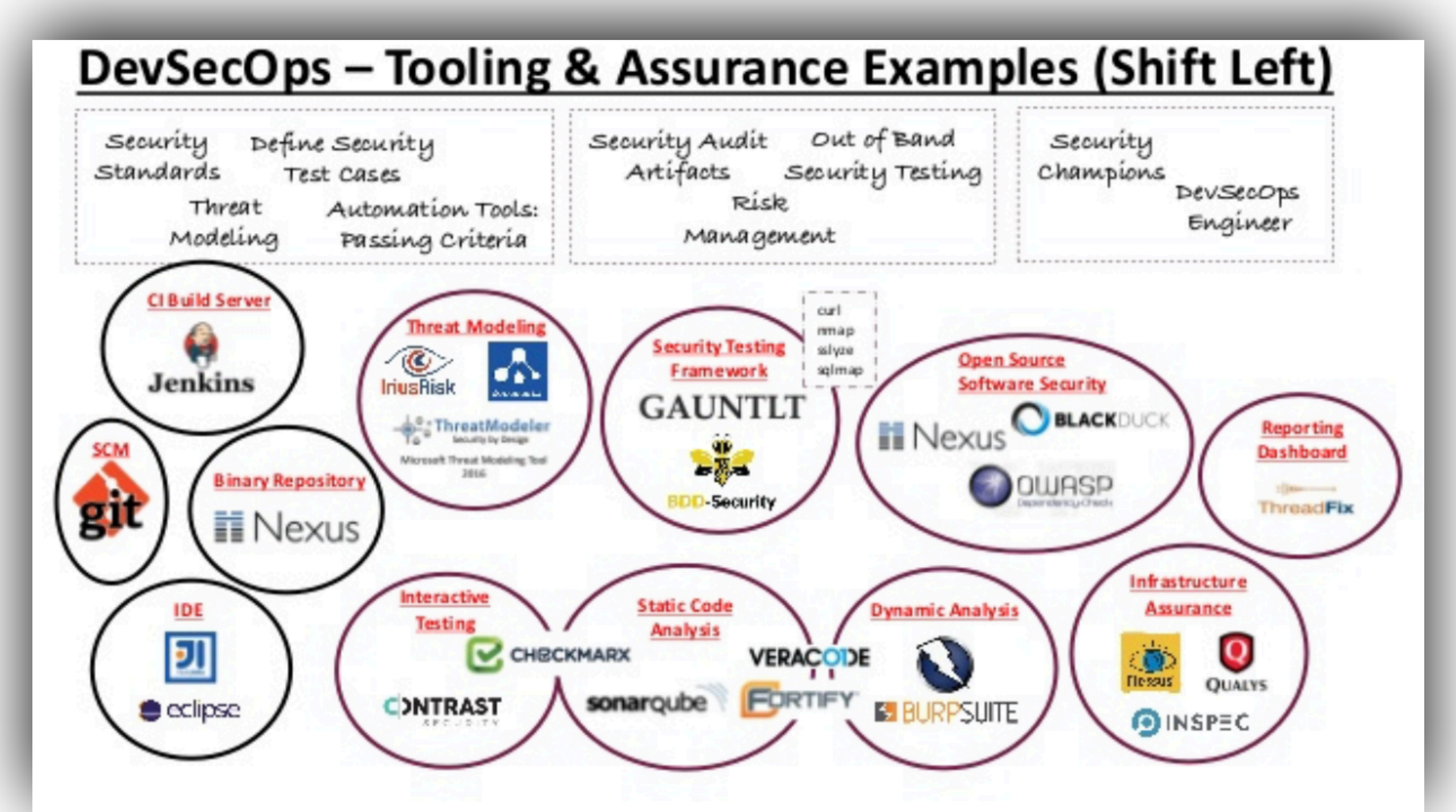
@WICKETT

SECURITY TOOLCHAIN FOR CI/CD



VERICA

@WICKETT



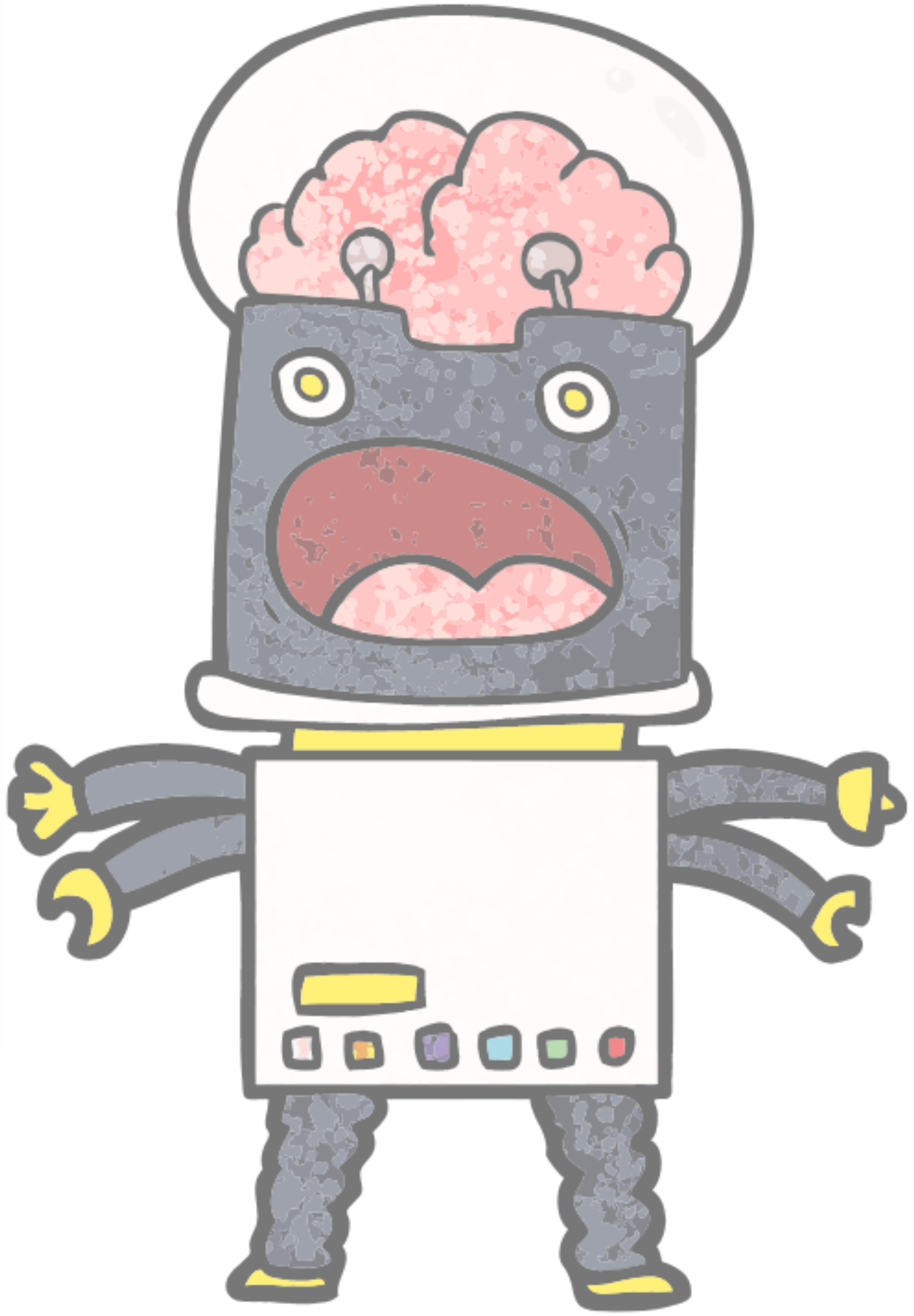
DevSecOps – Tooling & Assurance Examples (Shift Left)

Security Standards
Define Security Test Cases
Threat Modeling
Automation Tools: Passing Criteria

Security Audit Artifacts
Out of Band Security Testing
Risk Management

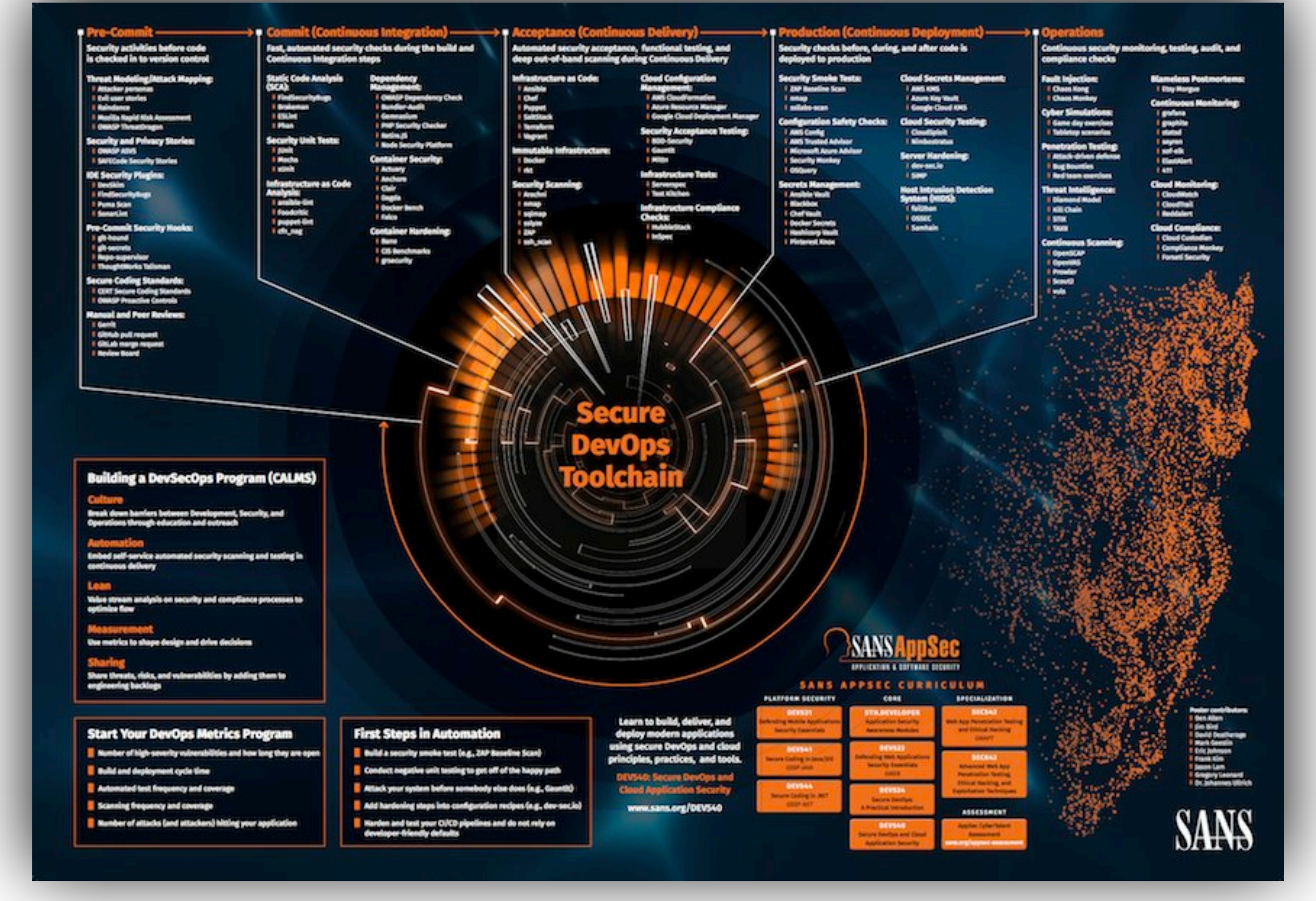
Security Champions
DevSecOps Engineer

https://www.slideshare.net/MichaelMan11/devsecops-pipeline-example-not-just-tools



VERICA

@WICKETT



Pre-Commit → **Commit (Continuous Integration)** → **Acceptance (Continuous Delivery)** → **Production (Continuous Deployment)** → **Operations**

Pre-Commit

Security activities before code is checked in to version control

Threat Modeling/Attack Mapping:
- Attacker personas
- Evil user stories
- Raindance
- Mozilla Rapid Risk Assessment
- OWASP ThreatDragon

Security and Privacy Stories:
- OWASP ASVS
- SAFECode Security Stories

IDE Security Plugins:
- DevSkim
- FindSecurityBugs
- Puma Scan
- SonarLint

Pre-Commit Security Hooks:
- git-hound
- git-secrets
- Repo-supervisor
- ThoughtWorks Talisman

Secure Coding Standards:
- CERT Secure Coding Standards
- OWASP Proactive Controls

Manual and Peer Reviews:
- Gerrit
- GitHub pull request
- GitLab merge request
- Review Board

Commit (Continuous Integration)

Fast, automated security checks during the build and Continuous Integration steps

Static Code Analysis (SCA):
- FindSecurityBugs
- Brakeman
- ESLint
- Phan

Security Unit Tests:
- JUnit
- Mocha
- xUnit

Infrastructure as Code Analysis:
- ansible-lint
- Foodcritic
- puppet-lint
- cfn_nag

Dependency Management:
- OWASP Dependency Check
- Bundler-Audit
- Gemnasium
- PHP Security Checker
- Retire.JS
- Node Security Platform

Container Security:
- Actuary
- Anchore
- Clair
- Dagda
- Docker Bench
- Falco

Container Hardening:
- Bane
- CIS Benchmarks
- grsecurity

Acceptance (Continuous Delivery)

Automated security acceptance, functional testing, and deep out-of-band scanning during Continuous Delivery

Infrastructure as Code:
- Ansible
- Chef
- Puppet
- SaltStack
- Terraform
- Vagrant

Immutable Infrastructure:
- Docker
- rkt

Security Scanning:
- Arachni
- nmap
- sqlmap
- sslyze
- ZAP
- ssh_scan

Cloud Configuration Management:
- AWS CloudFormation
- Azure Resource Manager
- Google Cloud Deployment Manager

Security Acceptance Testing:
- BDD-Security
- Gauntlt
- Mittn

Infrastructure Tests:
- Serverspec
- Test Kitchen

Infrastructure Compliance Checks:
- HubbleStack
- InSpec

Production (Continuous Deployment)

Security checks before, during, and after code is deployed to production

Security Smoke Tests:
- ZAP Baseline Scan
- nmap
- ssllabs-scan

Configuration Safety Checks:
- AWS Config
- AWS Trusted Advisor
- Microsoft Azure Advisor
- Security Monkey
- OSQuery

Secrets Management:
- Ansible Vault
- Blackbox
- Chef Vault
- Docker Secrets
- Hashicorp Vault
- Pinterest Knox

Cloud Secrets Management:
- AWS KMS
- Azure Key Vault
- Google Cloud KMS

Cloud Security Testing:
- CloudSploit
- Nimbostratus

Server Hardening:
- dev-sec.io
- SIMP

Host Intrusion Detection System (HIDS):
- fail2ban
- OSSEC
- Samhain

Operations

Continuous security monitoring, testing, audit, and compliance checks

Fault Injection:
- Chaos Kong
- Chaos Monkey

Cyber Simulations:
- Game day exercises
- Tabletop scenarios

Penetration Testing:
- Attack-driven defense
- Bug Bounties
- Red team exercises

Threat Intelligence:
- Diamond Model
- Kill Chain
- STIX
- TAXII

Continuous Scanning:
- OpenSCAP
- OpenVAS
- Prowler
- Scout2
- vuls

Blameless Postmortems:
- Etsy Morgue

Continuous Monitoring:
- grafana
- graphite
- statsd
- seyren
- sof-elk
- ElastAlert
- 411

Cloud Monitoring:
- CloudWatch
- CloudTrail
- Reddalert

Cloud Compliance:
- Cloud Custodian
- Compliance Monkey
- Forseti Security

Secure DevOps Toolchain

Building a DevSecOps Program (CALMS)

Culture
Break down barriers between Development, Security, and Operations through education and outreach

Automation
Embed self-service automated security scanning and testing in continuous delivery

Lean
Value stream analysis on security and compliance processes to optimize flow

Measurement
Use metrics to shape design and drive decisions

Sharing
Share threats, risks, and vulnerabilities by adding them to engineering backlogs

Start Your DevOps Metrics Program

- Number of high-severity vulnerabilities and how long they are open
- Build and deployment cycle time
- Automated test frequency and coverage
- Scanning frequency and coverage
- Number of attacks (and attackers) hitting your application

First Steps in Automation

- Build a security smoke test (e.g., ZAP Baseline Scan)
- Conduct negative unit testing to get off of the happy path
- Attack your system before somebody else does (e.g., Gauntlt)
- Add hardening steps into configuration recipes (e.g., dev-sec.io)
- Harden and test your CI/CD pipelines and do not rely on developer-friendly defaults

Learn to build, deliver, and deploy modern applications using secure DevOps and cloud principles, practices, and tools.

DEV540: Secure DevOps and Cloud Application Security

www.sans.org/DEV540

SANS AppSec
APPLICATION & SOFTWARE SECURITY

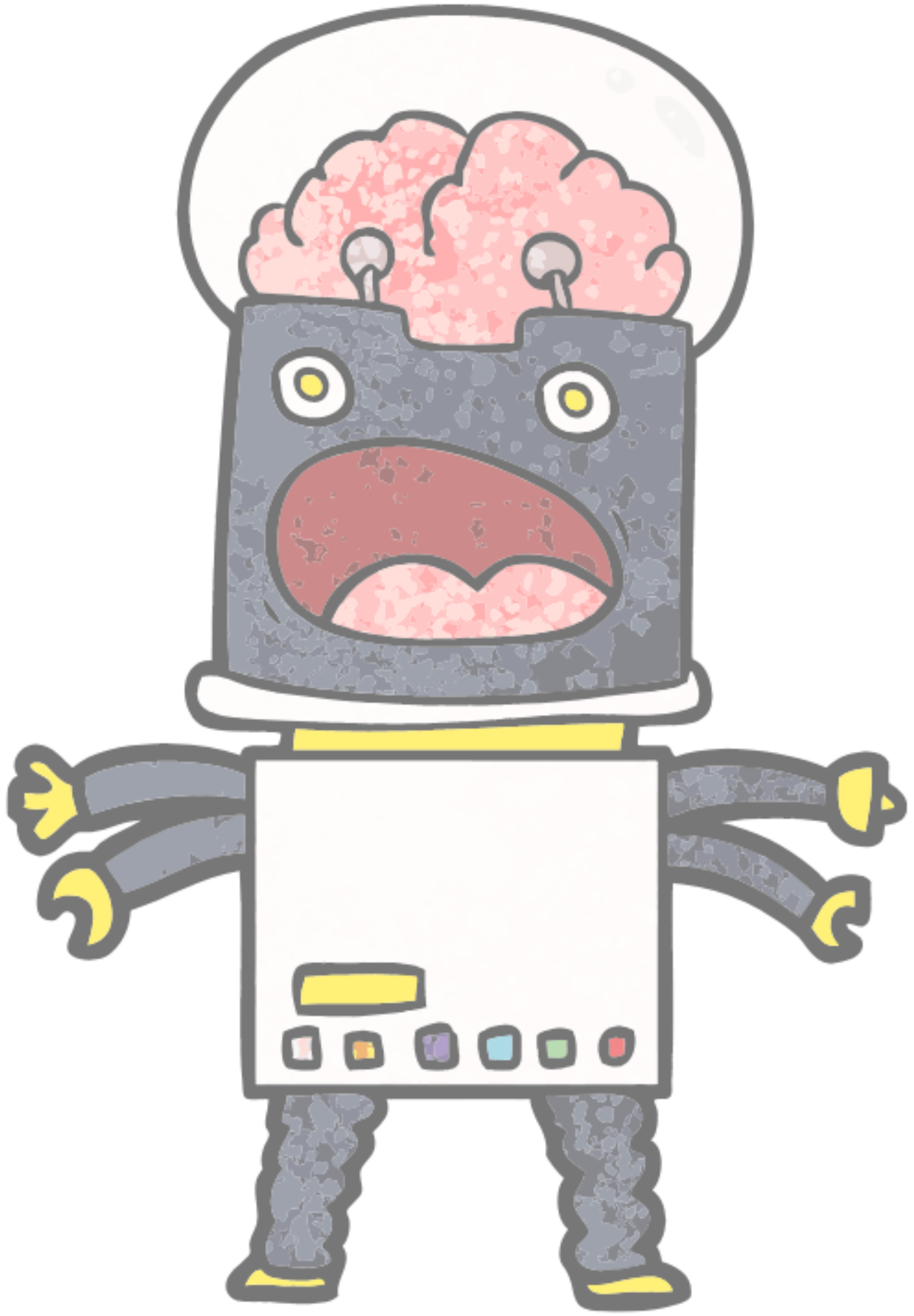
SANS APPSEC CURRICULUM

PLATFORM SECURITY	CORE	SPECIALIZATION
DEV531 Defending Mobile Applications Security Essentials	STH.DEVELOPER Application Security Awareness Modules	SEC542 Web App Penetration Testing and Ethical Hacking GWAPT
DEV541 Secure Coding in Java/JEE GSSP-JAVA	DEV522 Defending Web Applications Security Essentials GWEB	SEC642 Advanced Web App Penetration Testing, Ethical Hacking, and Exploitation Techniques
DEV544 Secure Coding in .NET GSSP-.NET	DEV534 Secure DevOps: A Practical Introduction	ASSESSMENT
	DEV540 Secure DevOps and Cloud Application Security	AppSec CyberTalent Assessment sans.org/appsec-assessment

Poster contributors:
- Ben Allen
- Jim Bird
- David Deatherage
- Mark Geeslin
- Eric Johnson
- Frank Kim
- Jason Lam
- Gregory Leonard
- Dr. Johannes Ullrich

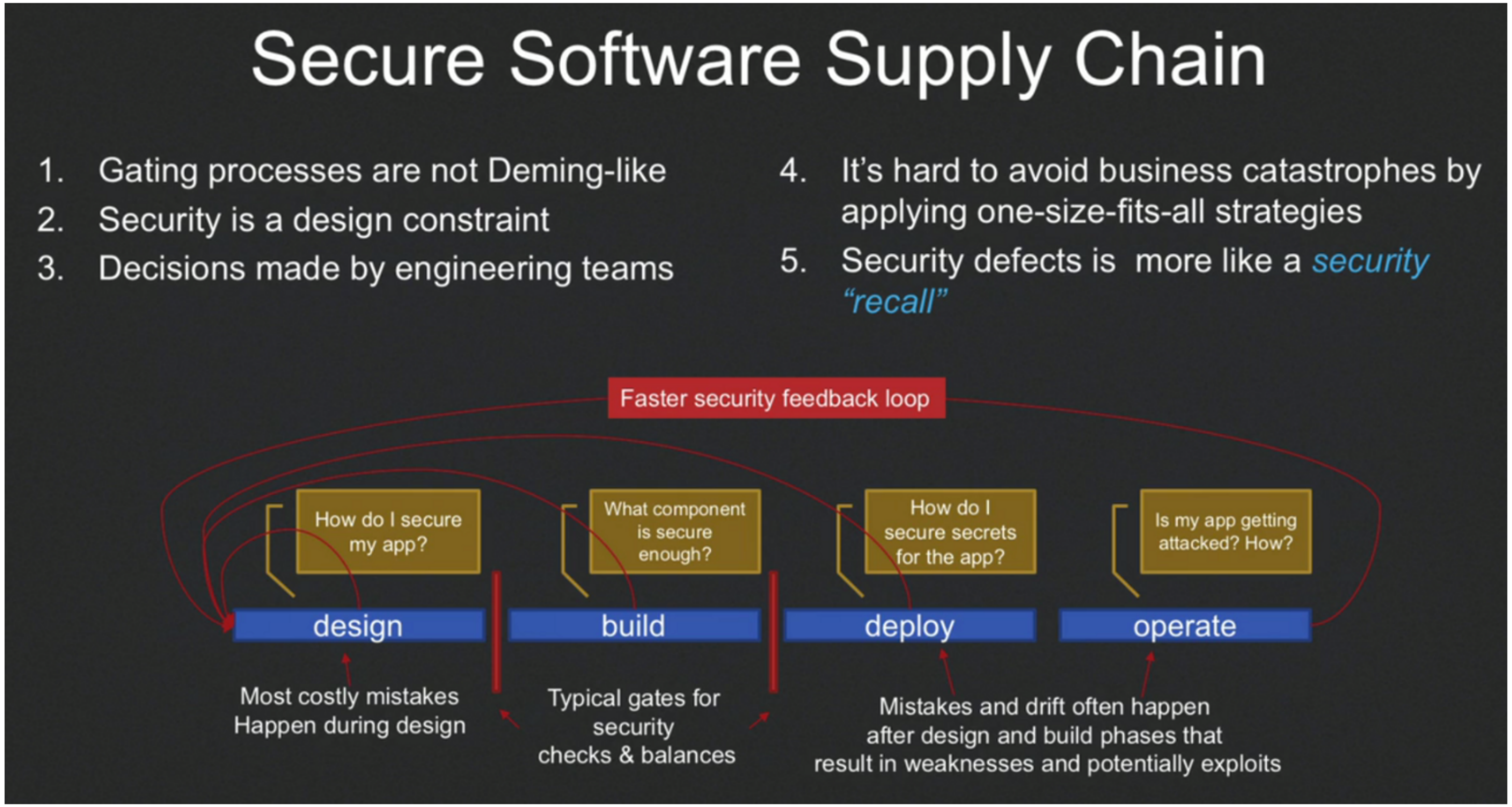
SANS

https://www.sans.org/security-resources/posters/appsec/secure-devops-toolchain-swat-checklist-60



VERICA

@WICKETT



Secure Software Supply Chain

1. Gating processes are not Deming-like
2. Security is a design constraint
3. Decisions made by engineering teams
4. It's hard to avoid business catastrophes by applying one-size-fits-all strategies
5. Security defects is more like a *security "recall"*

Faster security feedback loop

How do I secure my app?

What component is secure enough?

How do I secure secrets for the app?

Is my app getting attacked? How?

design

build

deploy

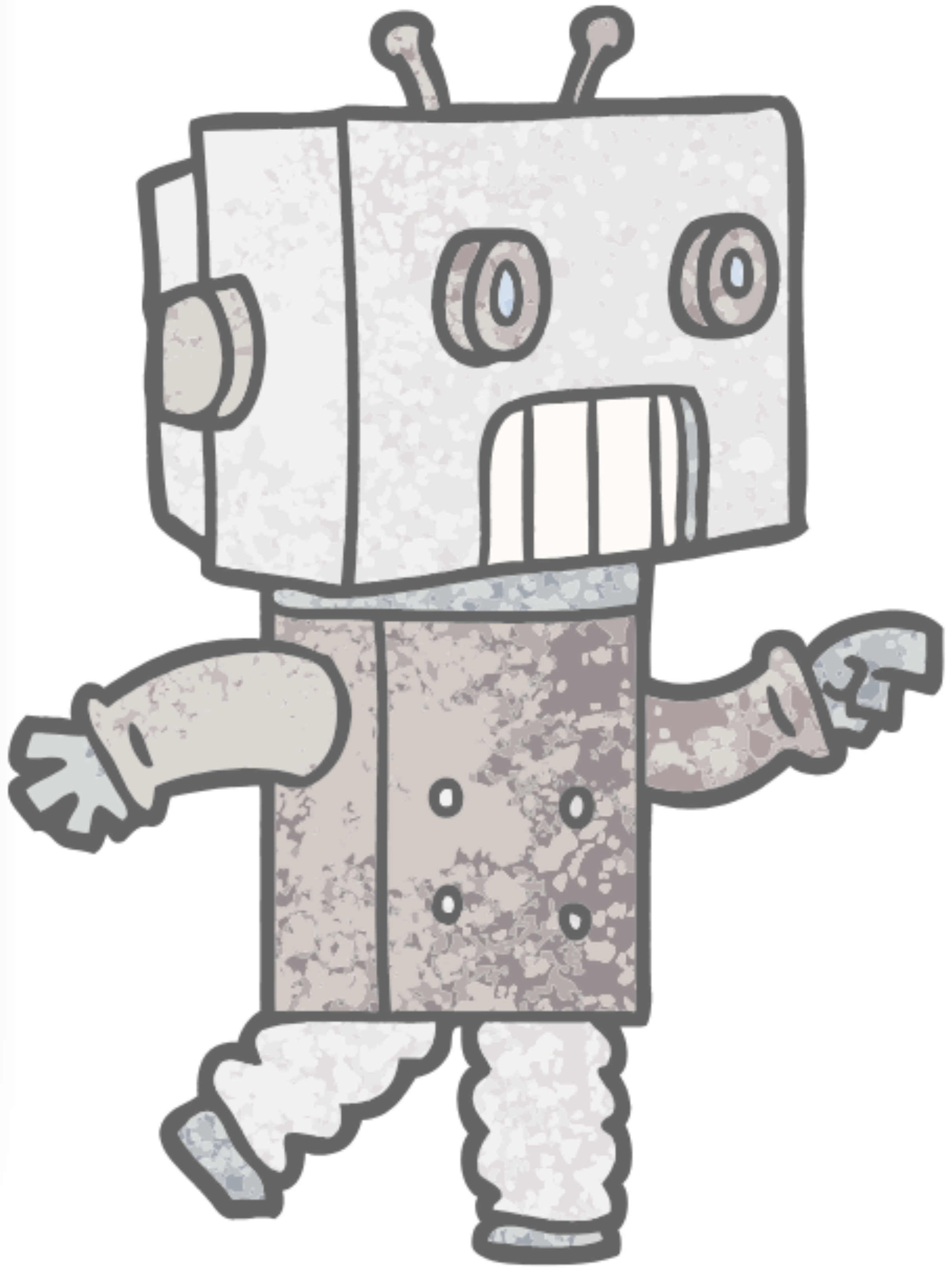
operate

Most costly mistakes Happen during design

Typical gates for security checks & balances

Mistakes and drift often happen after design and build phases that result in weaknesses and potentially exploits

Secure Software Supply Chain presented by Shannon Leitz at DevOps Days Austin 2016.



@WICKETT

Develop

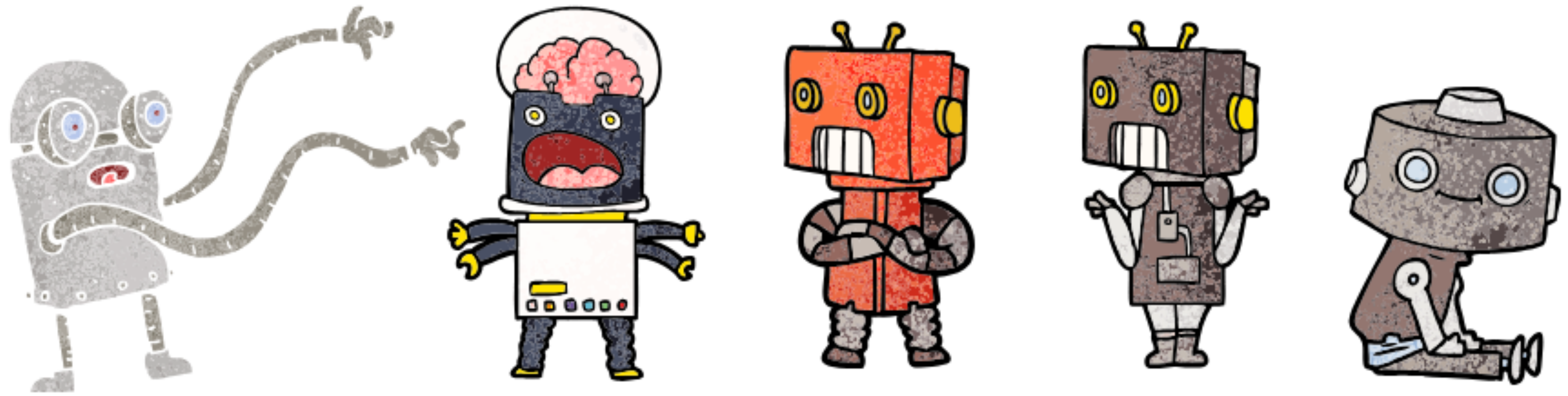
Inherit

Build

Deploy

Operate

VERICA



@WICKETT

Develop | Inherit | Build | Deploy | Operate

The design and development of an application and its features. Including all the development practices like version control, sprint planning, unit-testing.

Develop | Inherit | Build | Deploy | Operate

Security Activities and Considerations

- **Threat Modeling**
- **Security Stories**
- **Authentication to Push**
- **Development Standards**
- **Peer Review**
- **Static Code Analysis**
- **Unit Tests for Security**

Develop | Inherit | Build | Deploy | Operate

Threat Modeling and Security Stories

- The Threat Modeling Book
- OWASP App Threat Modeling Cheat Sheet
- Evil User Stories (link)
- OWASP Application Security Verification Standard
- OWASP threatdragon.org
- Mozilla Rapid Risk Assessment (link)

Develop | Inherit | Build | Deploy | Operate

Development Standards

- Pre-commit Hooks for Security
- Coding Standards (Security and otherwise)

- Peer Review
- Single Mainline Branch
- Linting and Code Hygiene

Code Standards and Team Tooling

- **gometalinter** if you use golang or find one for whatever your language of choice
- **gofmt** formats the code automatically and makes everything look the same, easier for everyone to grok (again, this is specific to golang)

Keeping Secrets Out of Codebase

- **git-secrets** Prevents you from committing passwords and other sensitive information to a git repository. From awslabs. (**link**)

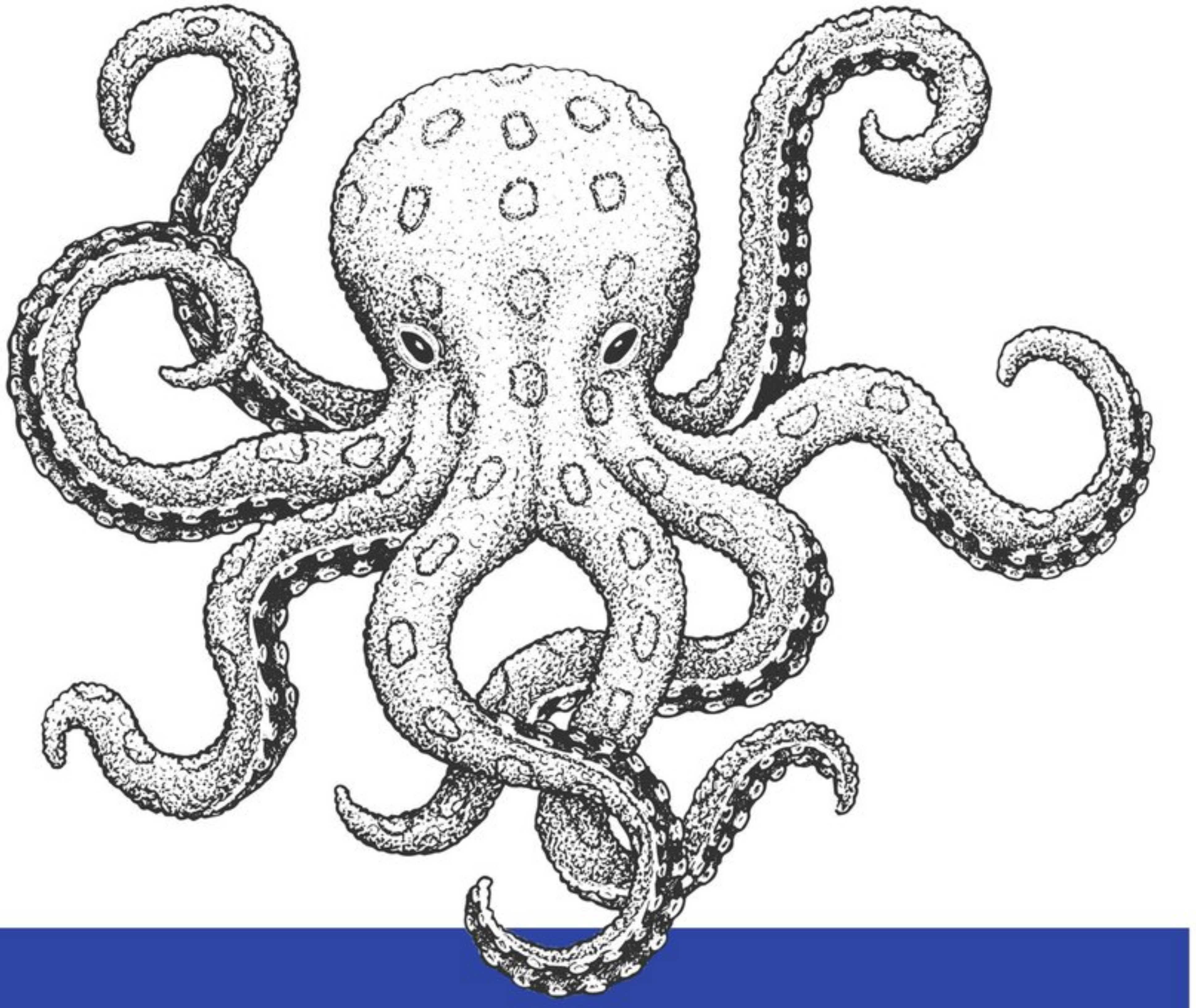
- **git-hound** Hound is a Git plugin that helps prevent sensitive data from being committed into a repository by sniffing potential commits against PCRE regular expressions. (**link**)

- **Other Reources:**
    - Talisman link
    - Repo Supervisor link

A Bug is a Bug is a Bug Philosophy

Security testing where other error testing lives. In the IDE, in local build env, and in CI system.

Just memorize these fourteen contextually dependant instructions



Exiting Vim

Eventually

O RLY?

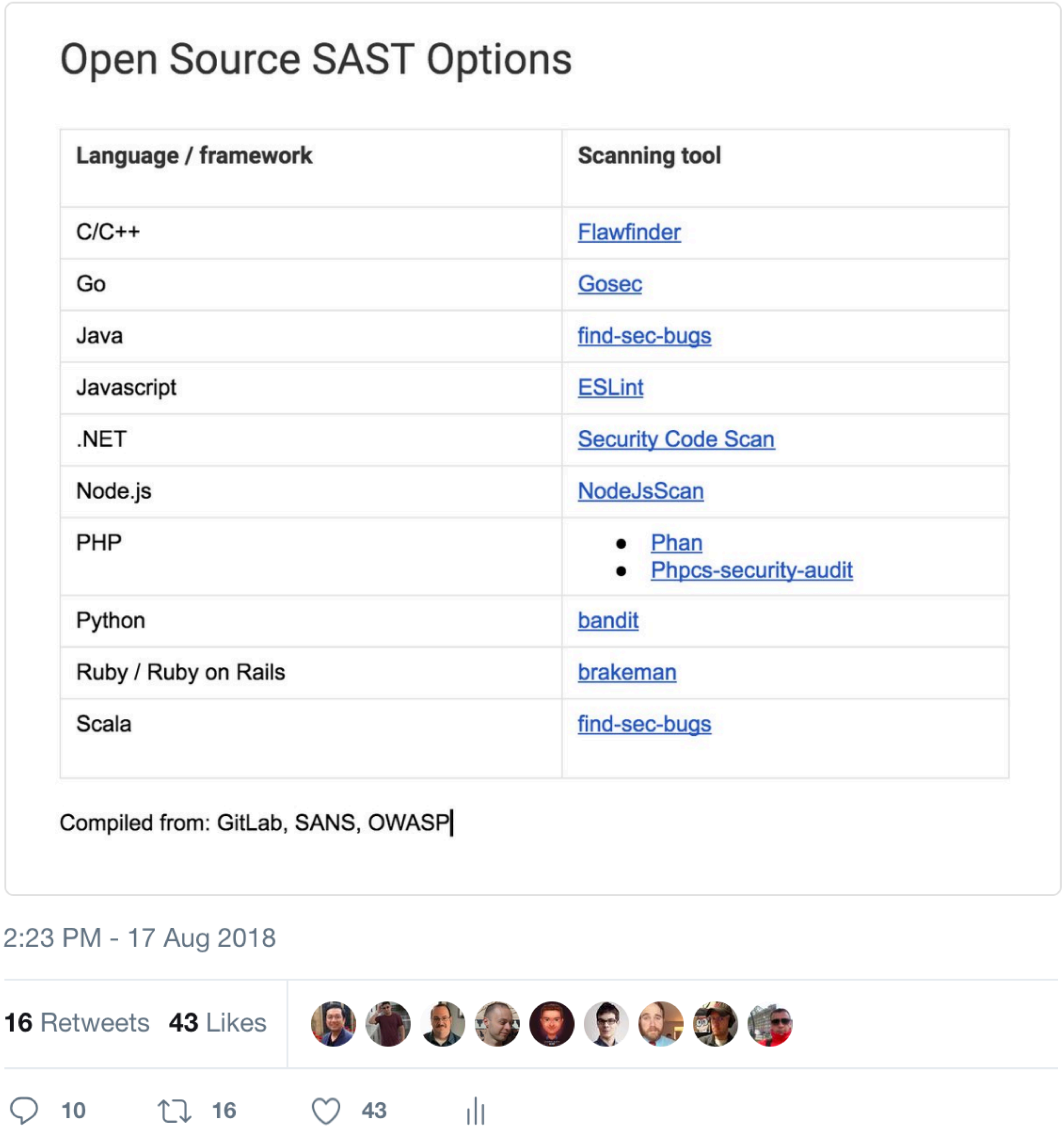
@ThePracticalDev

Develop | Inherit | Build | Deploy | Operate

Static Code Analysis

- **Not unfamiliar territory for security pros**
- **Static Application Security Testing (SAST)**
- **IDE Plugin if possible**

- **Open Source:** Brakeman (Ruby), FindSecurityBugs (Java), Phan (PHP), gosec (golang), Puma (C#)
- **Paid:** Brakeman Pro, Veracode, Fortify, …



Open Source SAST Options

Language / framework	**Scanning tool**
C/C++	Flawfinder
Go	Gosec
Java	find-sec-bugs
Javascript	ESLint
.NET	Security Code Scan
Node.js	NodeJsScan
PHP	• Phan • Phpcs-security-audit
Python	bandit
Ruby / Ruby on Rails	brakeman
Scala	find-sec-bugs

Compiled from: GitLab, SANS, OWASP

2:23 PM - 17 Aug 2018

16 Retweets 43 Likes

Unit Testing for Security

- Unit Testing is the currency of Developers

- JUnit, Rspec, Testing (golang), ….

- Goal is to have security tests being written with other unit tests or whatever testing patterns you use: TDD, BDD, ATDD, …

Develop Inherit Build Deploy Operate

Questions to Ask

Are the developers testing for security locally before it gets to the CI system?

Do we practice good hygiene and coding practices?

Are we preventing secrets from leaking into version control?

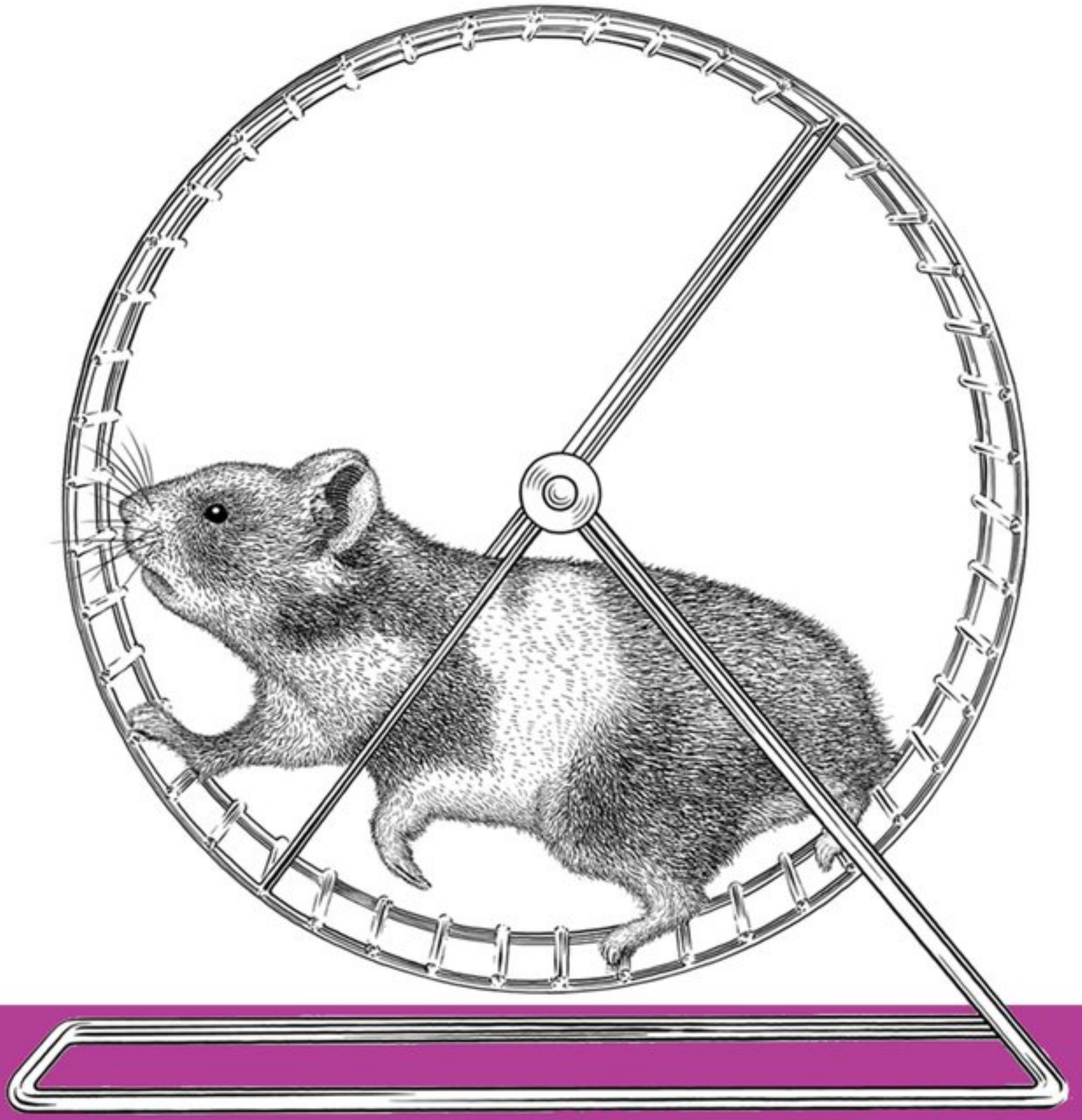
VERICA

@WICKETT

Develop | **Inherit** | Build | Deploy | Operate

This is an overlooked phase because it is the most invisible as software dependencies get bundled in and inherited in our own code and upstream.

“What did I do to deserve this?”



Resolving Broken Dependencies

This is Your Life Now

O RLY?

@ThePracticalDev

Develop | **Inherit** | Build | Deploy | Operate

Security Considerations

- This is your real LOC count!
- The Software Delivery Supply Chain
- Publish a Bill of Materials and trace back
- This is not just application dependencies and libraries, but also OS-level (remember shellshock, heartbleed, ..)

Language Tooling

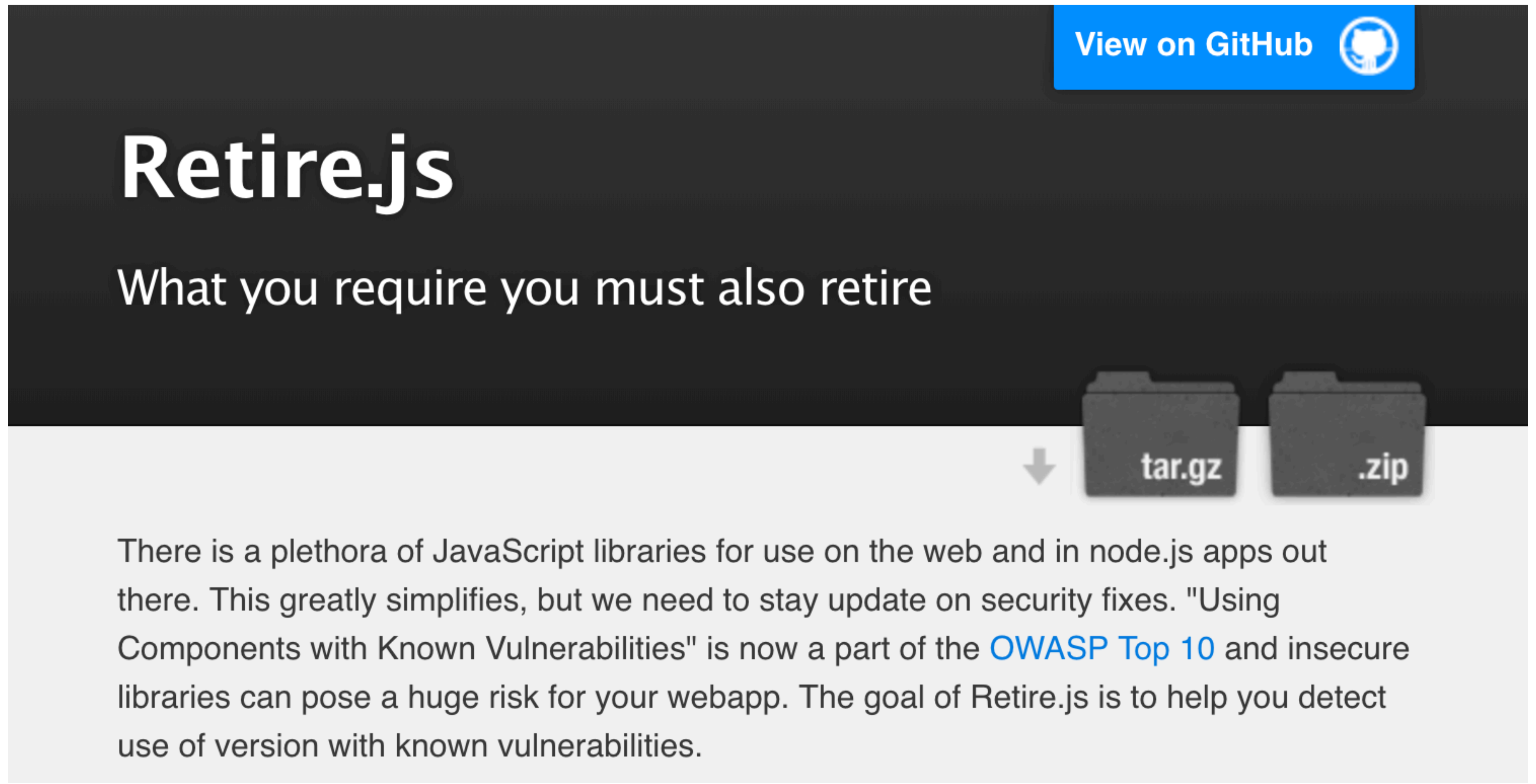
- **bundler-audit** - checks for vulnerable versions of gems in your ruby code (link)

- **OWASP Dep Check** - mostly Java

- **nsp** - node security platform (link)

- **Paid options:** Sonatype, Snyk, BlackDuck, JFrog

- **Retire.js** - known vuln JS libs (link)



Retire.js

What you require you must also retire

There is a plethora of JavaScript libraries for use on the web and in node.js apps out there. This greatly simplifies, but we need to stay update on security fixes. "Using Components with Known Vulnerabilities" is now a part of the OWASP Top 10 and insecure libraries can pose a huge risk for your webapp. The goal of Retire.js is to help you detect use of version with known vulnerabilities.

Develop | **Inherit** | Build | Deploy | Operate

- **Over 30% of containers in Docker Hub have high sev vulns (source)**

- **Open Source:** Docker Bench for Security, Clair, falco, anchore, …

- **Paid Options:** aqua, twistlock

Speaker Deck   Published on Feb 6, 2017

What's Inside That Container?

Containers and config management in the real world

Gareth Rushgrove
Puppet

puppet

share

Gareth Rush...
61 Presentations

★ Star this Talk   9 Stars
Published in   Technology
Stats   2,203 Views

Share

Twitter, Facebook
Embed
Direct Link
Download PDF

What's Inside That Container? by Gareth Rushgrove
Published February 6, 2017 in Technology

VERICA

@WICKETT

Develop | **Inherit** | Build | Deploy | Operate

Questions to Ask

What have I bundled into my app that is making vulnerable?

Am I publishing a Bill of Materials with my application?

Develop | Inherit | **Build** | Deploy | Operate

This phase is where the CI build system runs all the build steps and does acceptance testing. Previous testing and tooling gets verified here.

Develop | Inherit | **Build** | Deploy | Operate

Security Considerations

- **Outside-In Security Testing**
- **Infra as Code (Testing)**
- **Dynamic Application Security Testing (DAST)**

- **Compliance on every build!**
- **Cloud provider config as code**
- **Using containers**

Develop | Inherit | **Build** | Deploy | Operate

Dynamic Application Security Scanners

- These all require tuning and can be difficult to integrate into build pipelines.

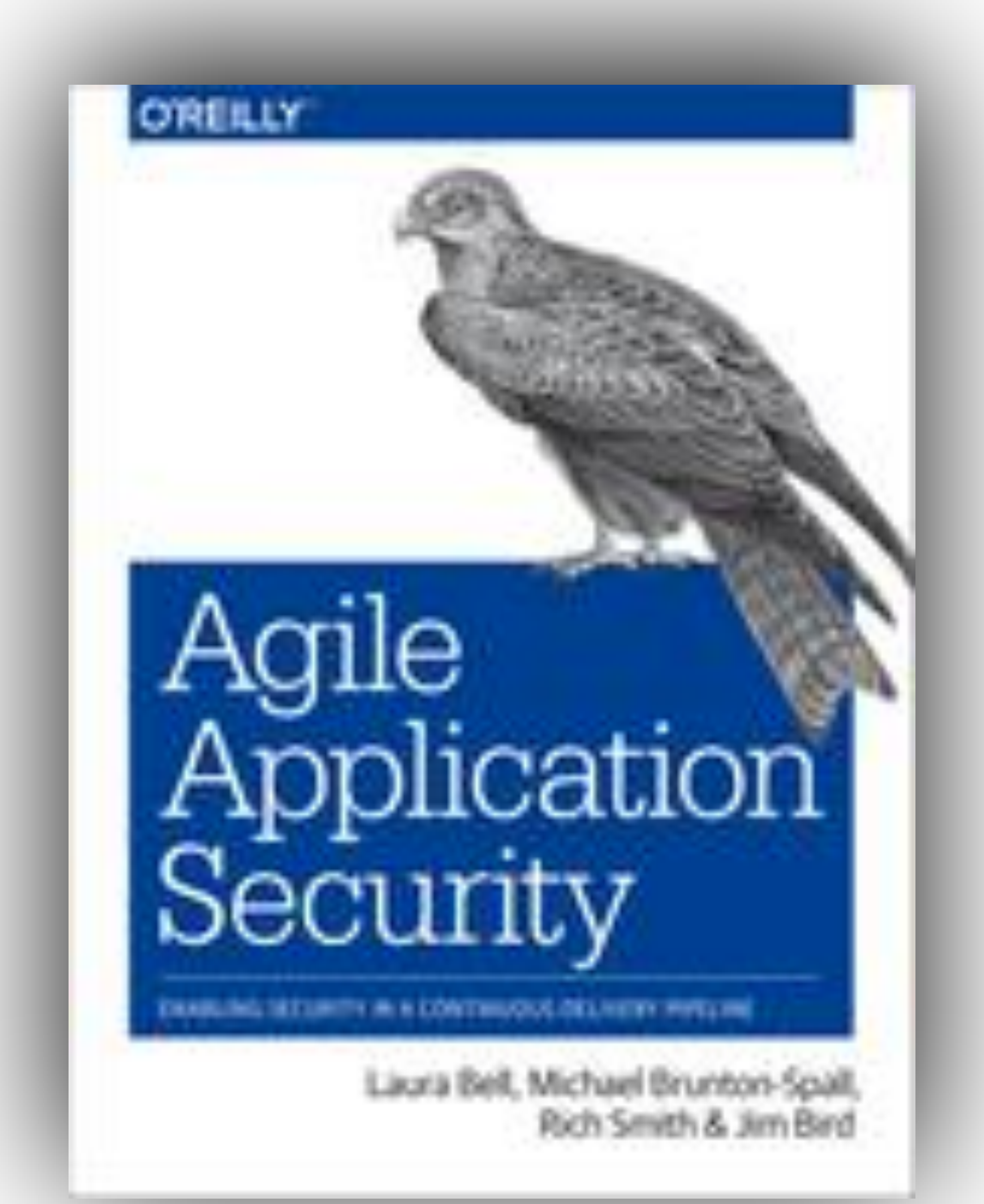
- Application Security scanners: Nikto, Arachni, ZAP, sqlmap, xsser, …

- Other - SSLyze, nmap, ssh_scan

- See Kali Linux

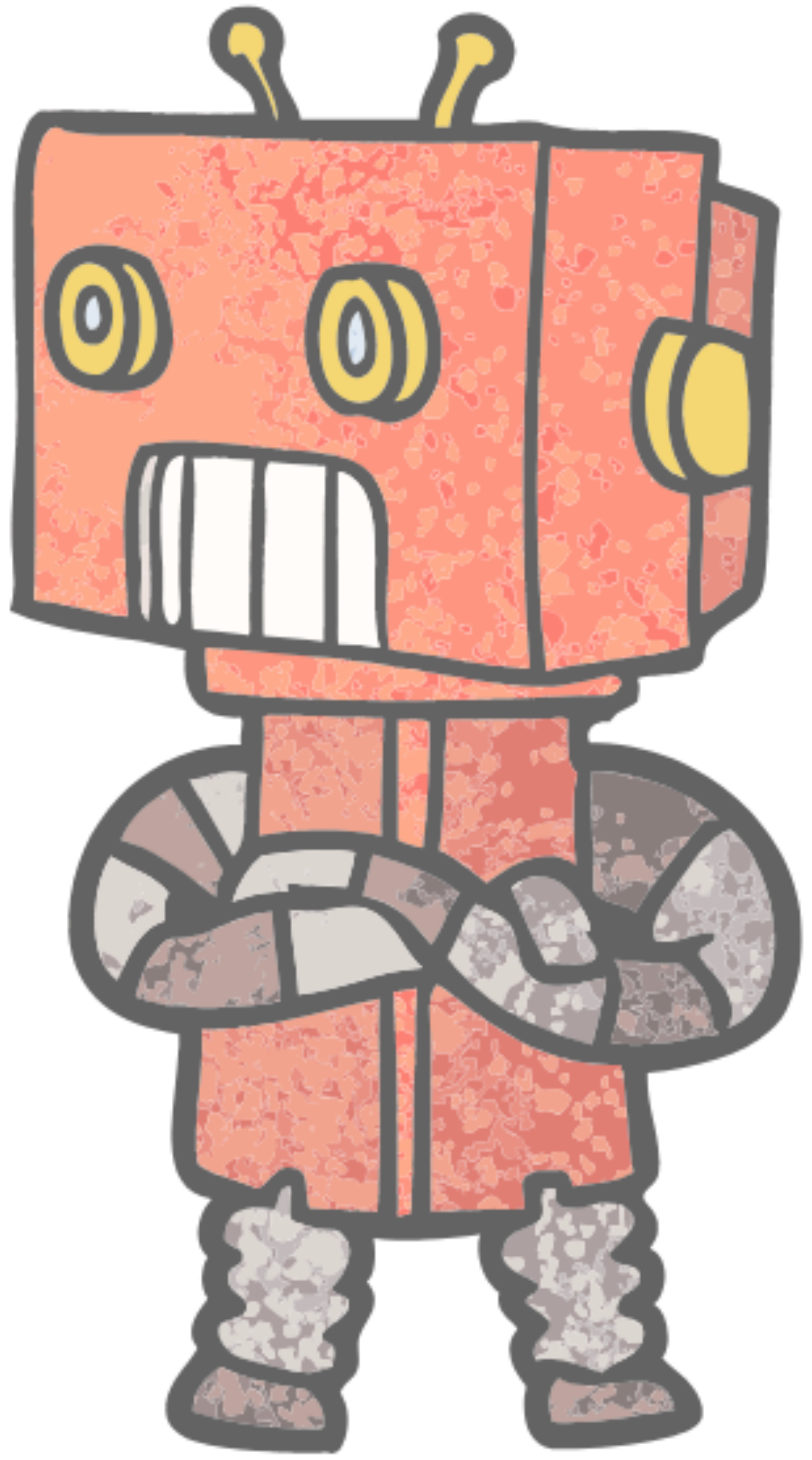
- Paid: Qualys, AppScan, BurpSuite, …

The goal should be to come up with a set of automated tests that probe and check security configurations and runtime system behavior for security features that will execute every time the system is built and every time it is deployed.


O'REILLY
Agile Application Security
ENABLING SECURITY IN A CONTINUOUS DELIVERY PIPELINE
Laura Bell, Michael Brunton-Spall, Rich Smith & Jim Bird

GAUNTLT

BE MEAN TO YOUR CODE AND LIKE IT



Framework with Security testing written in a natural language that developers, security and operations can understand.

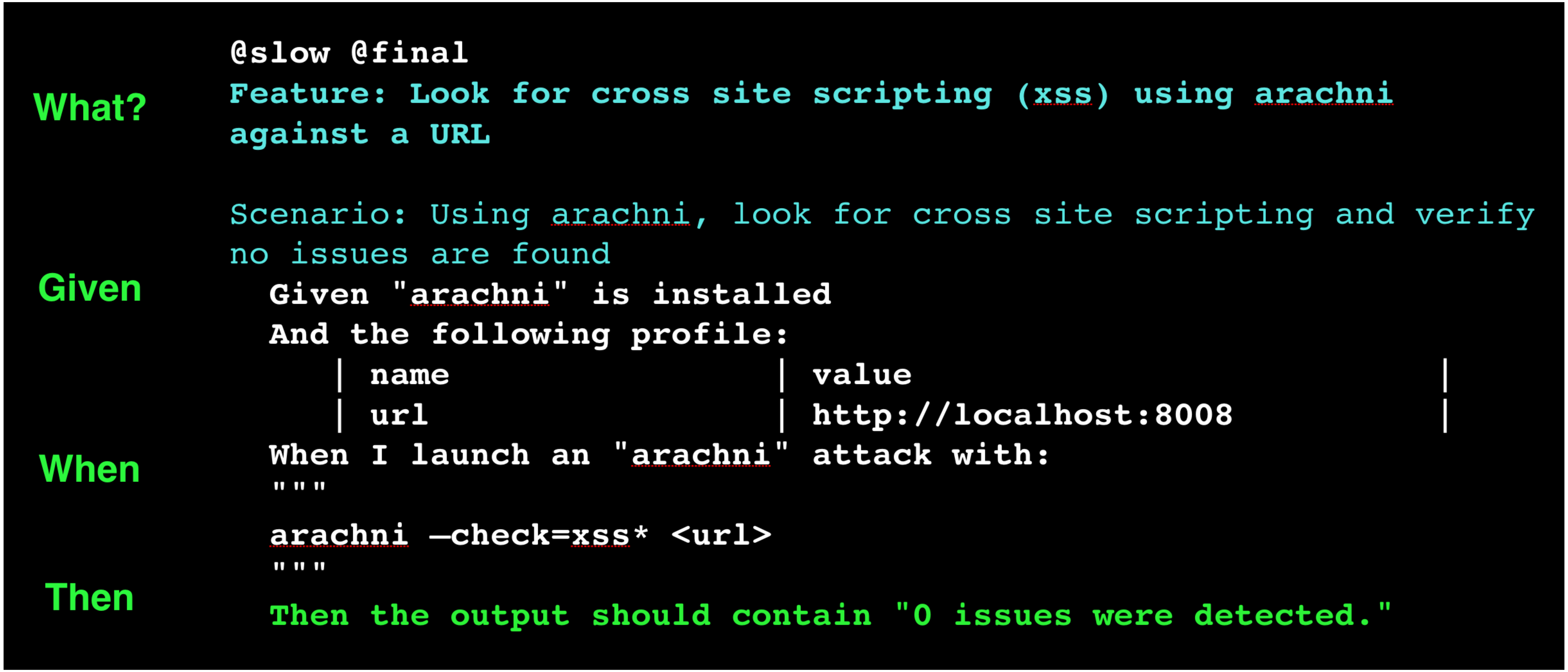
Gauntlt wraps security testing tools but does not install tools

Gauntlt was built to be part of the CI/CD pipeline

Open source, MIT License,

gauntlt.org

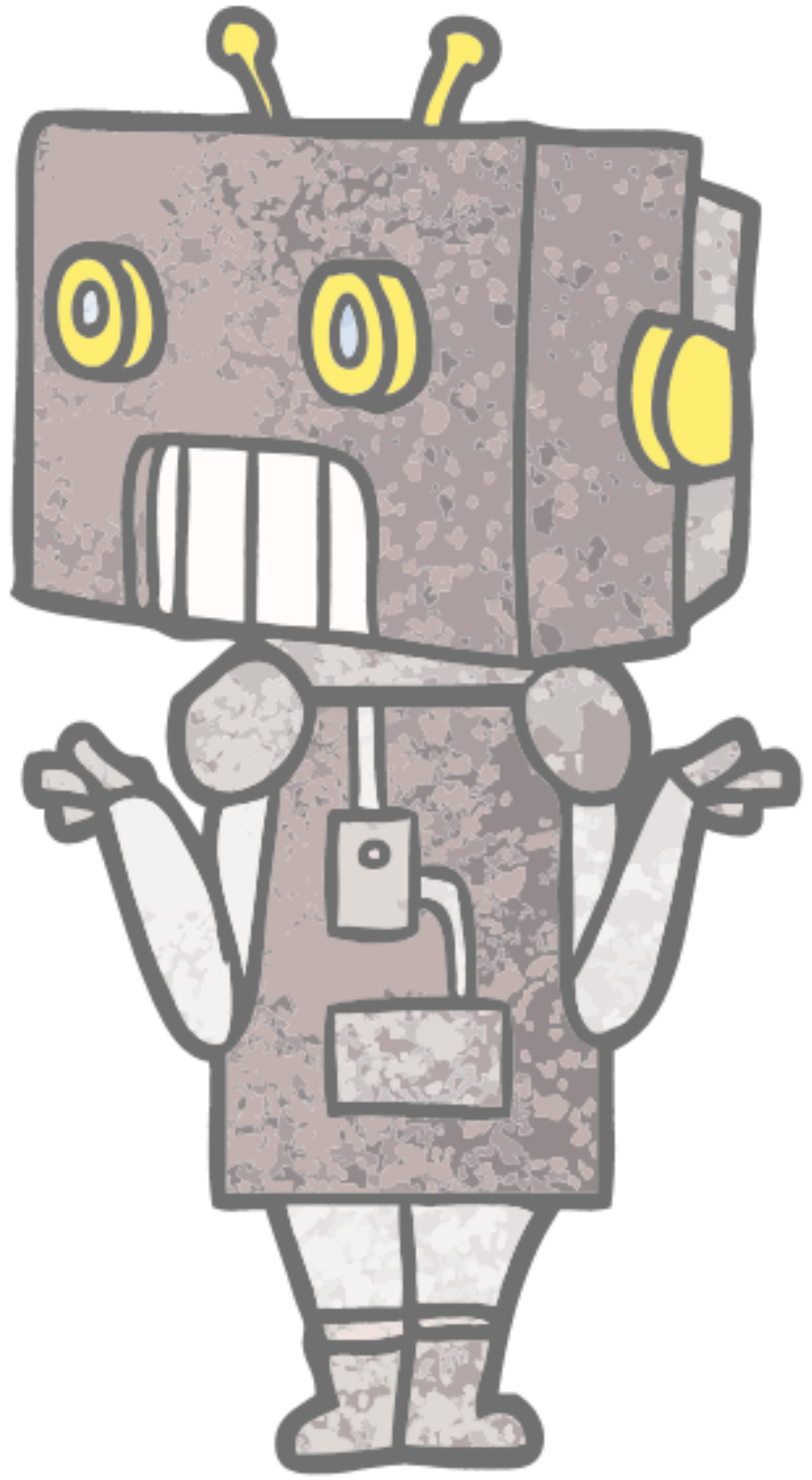
Gauntlt Example



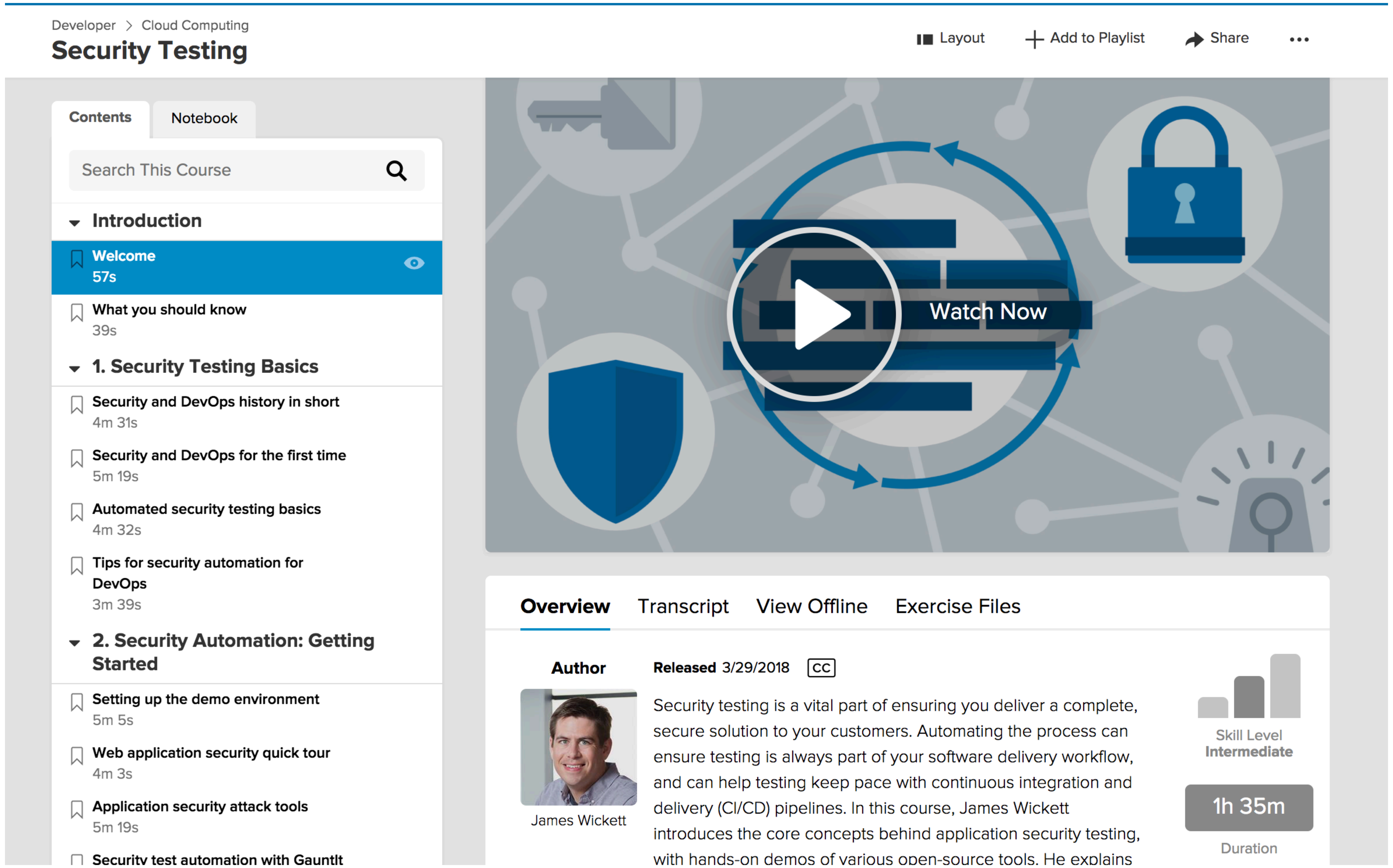
```
@slow @final
Feature: Look for cross site scripting (xss) using arachni
against a URL

Scenario: Using arachni, look for cross site scripting and verify
no issues are found
  Given "arachni" is installed
  And the following profile:
     | name                 | value                            |
     | url                  | http://localhost:8008            |
  When I launch an "arachni" attack with:
  """

  arachni —check=xss* <url>
  """
  Then the output should contain "0 issues were detected."
```

“We have saved millions of dollars using GauntIt for the largest healthcare industry project.”

- Aaron Rinehart, UnitedHealthCare



A Whole Course on Security Testing with Gauntlt



Security Testing

Introduction

- **Welcome** 57s
- **What you should know** 39s

1. Security Testing Basics

- **Security and DevOps history in short** 4m 31s
- **Security and DevOps for the first time** 5m 19s
- **Automated security testing basics** 4m 32s
- **Tips for security automation for DevOps** 3m 39s

2. Security Automation: Getting Started

- **Setting up the demo environment** 5m 5s
- **Web application security quick tour** 4m 3s
- **Application security attack tools** 5m 19s
- **Security test automation with Gauntlt**

Author

James Wickett

Released 3/29/2018

Security testing is a vital part of ensuring you deliver a complete, secure solution to your customers. Automating the process can ensure testing is always part of your software delivery workflow, and can help testing keep pace with continuous integration and delivery (CI/CD) pipelines. In this course, James Wickett introduces the core concepts behind application security testing, with hands-on demos of various open-source tools. He explains

Skill Level Intermediate

1h 35m

Duration

https://www.linkedin.com/learning/devsecops-automated-security-testing/welcome

VERICA

@WICKETT

Infrastructure and Compliance

- **Test Kitchen** - https://kitchen.ci/
- **Serverspec** - http://serverspec.org/
- **InSpec** - Continuous Compliance Testing **link**

- **Cloud Provider is Infrastructure too**
- **Version and test Cloud Config**

Questions to Ask

Am I testing for security low hanging fruit?

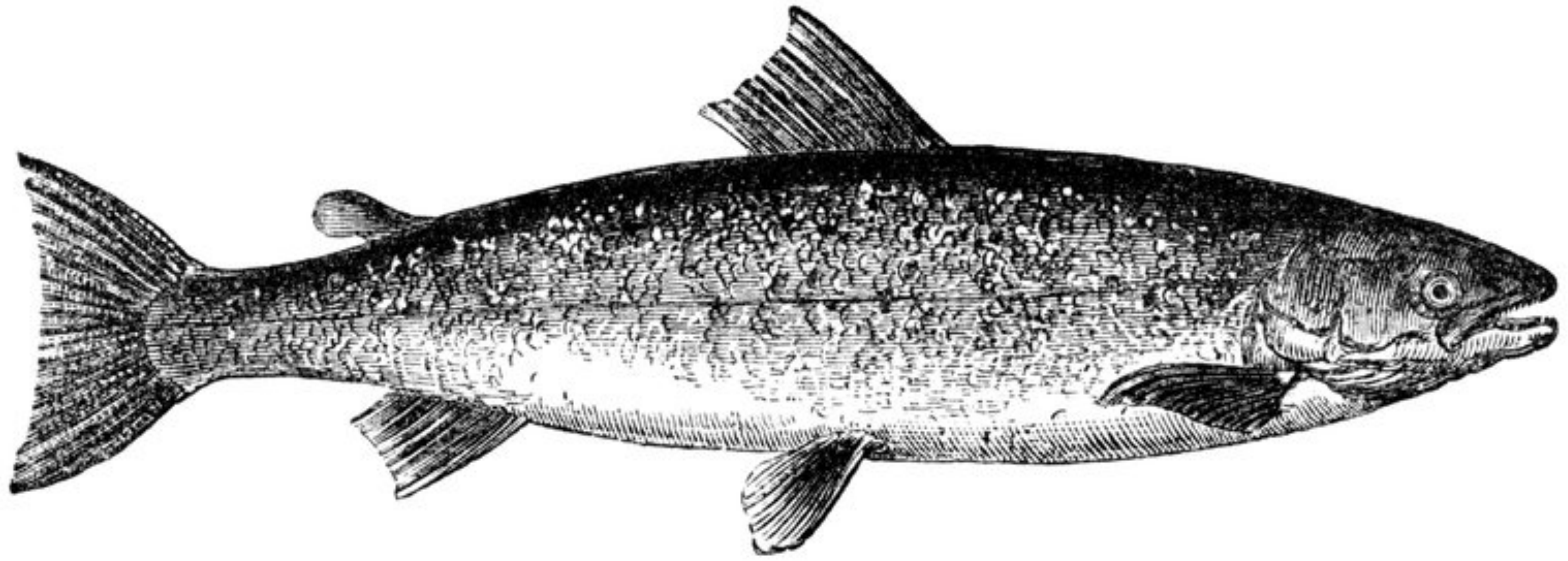
Am I arming my pipeline with attack tools to exercise my application?

Have I validated the previous two phases of testing in secure build environment?

Develop | Inherit | Build | **Deploy** | Operate

The phase where software moves from our testing to where customers are able to operate it for the first time.

Security by optimism and prayer



Expert

Hoping Nobody Hacks You

O RLY?

@ThePracticalDev

Develop | Inherit | Build | **Deploy** | Operate

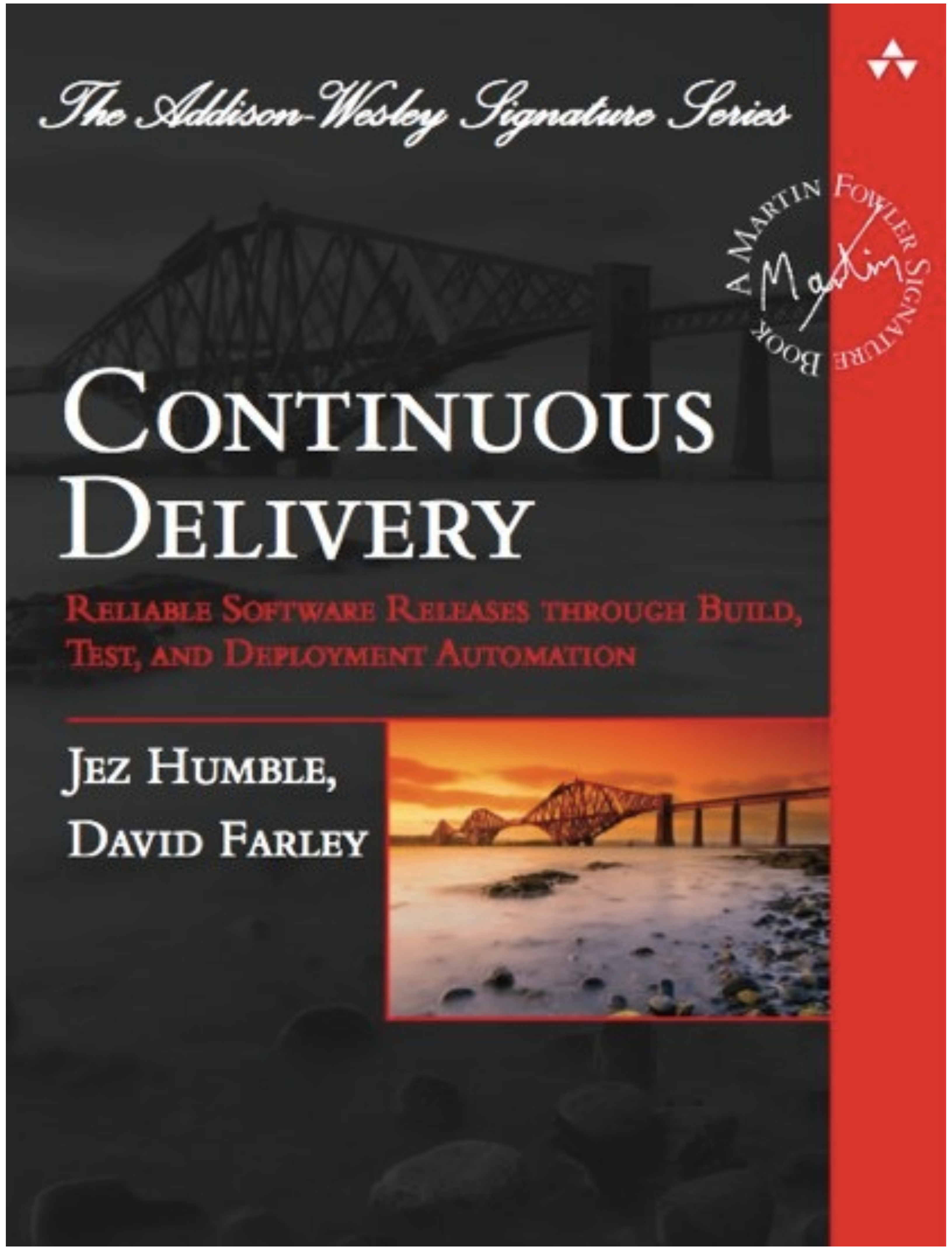
Security Considerations

- Watch out for Compliance
- Secrets Management
- Deploy Accountability
- Authorization and Logging

- Monitoring Deploys
- Infra as Code (Execution)
- Repeatable Execution

VERICA

@WICKETT



The Addison-Wesley Signature Series

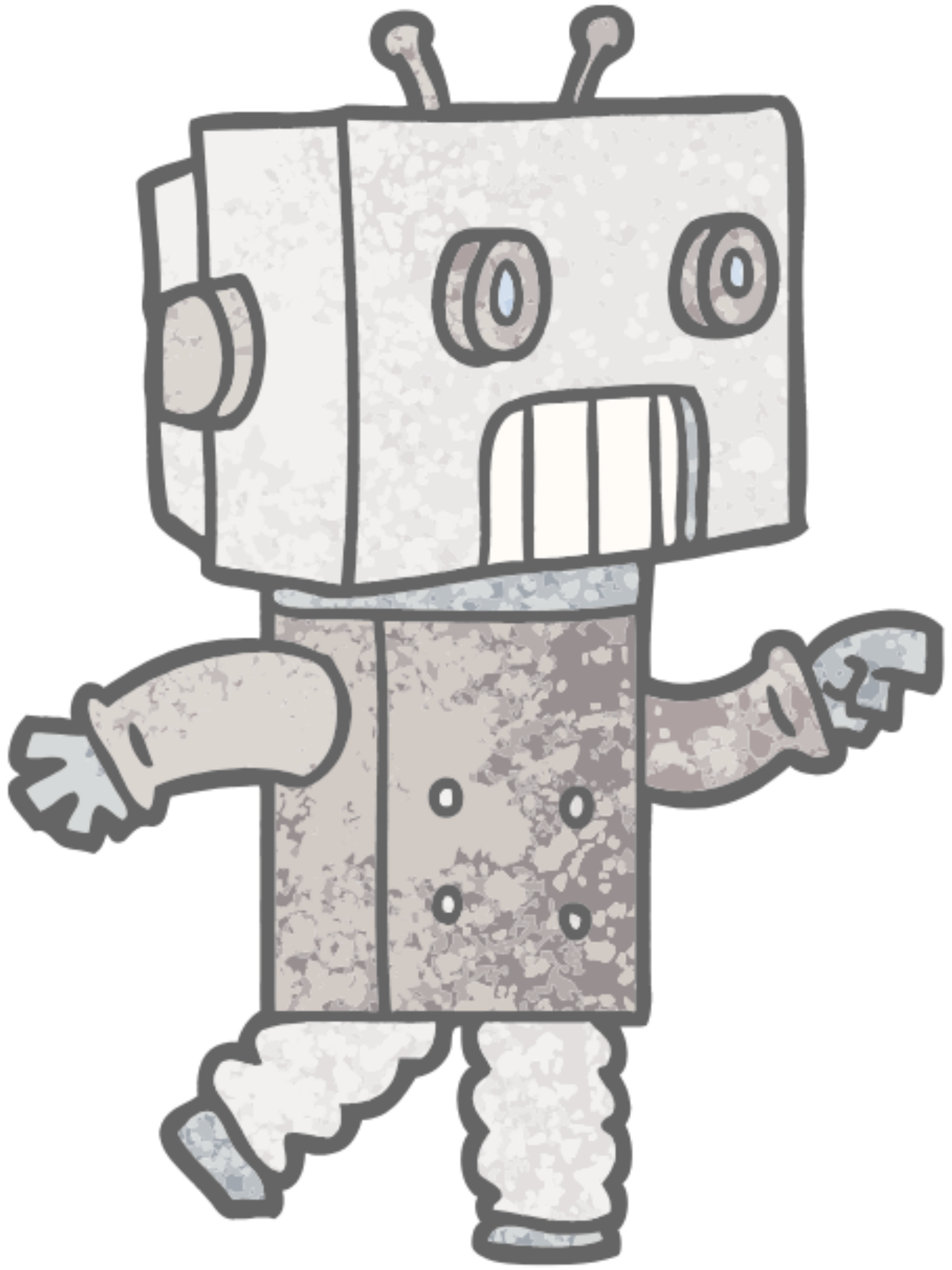
A Martin Fowler Signature Book

Martin

Continuous Delivery

Reliable Software Releases through Build, Test, and Deployment Automation

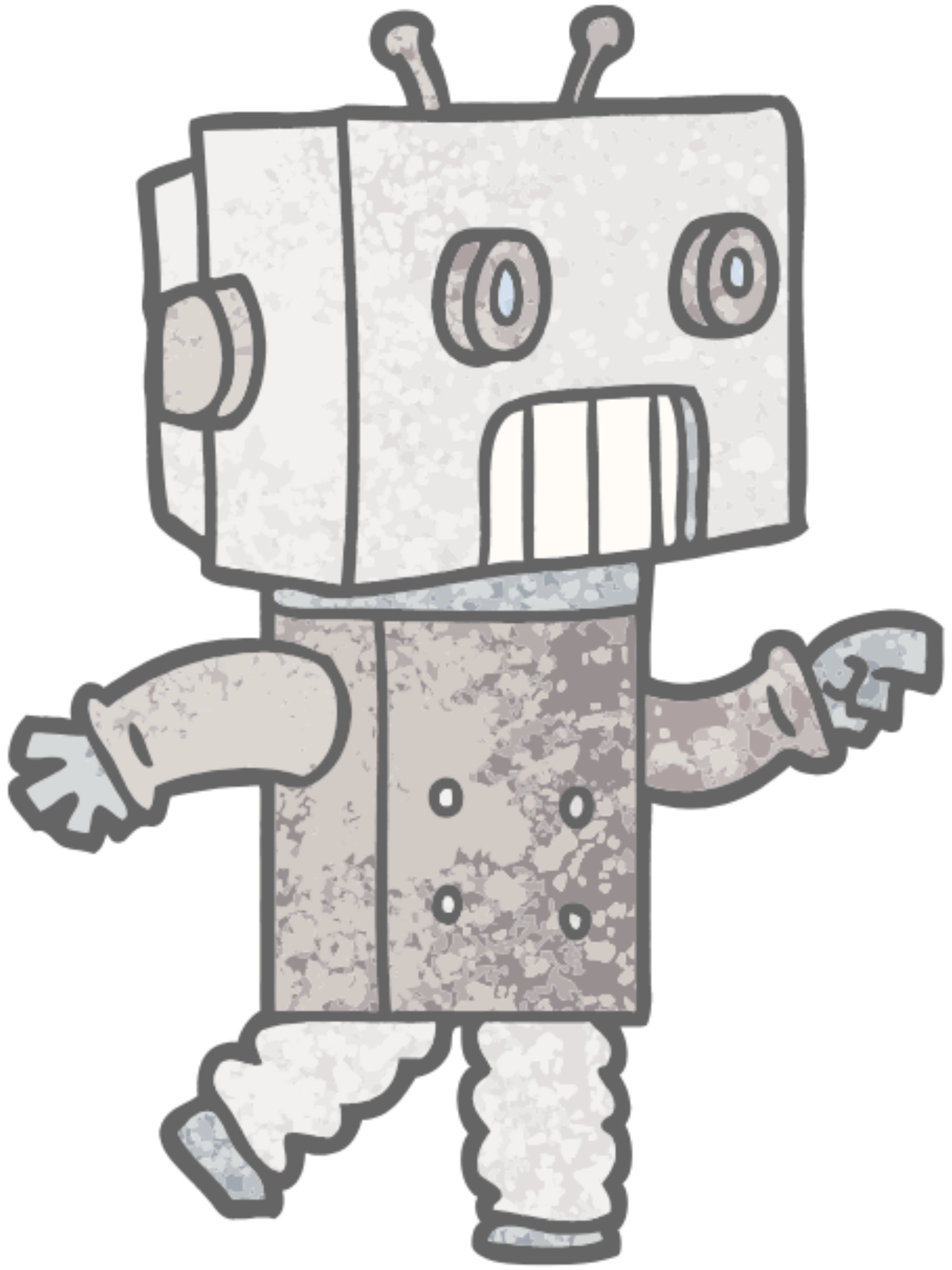
Jez Humble, David Farley



VERICA

@WICKETT

Roughly 10,000 deploys in the last 2.5 yrs



VERICA

@WICKETT

[Deploys] can be treated as standard or routine changes that have been pre-approved by management, and that don't require a heavyweight change review meeting.

O'REILLY
Agile Application Security
Enabling Security in a Continuous Delivery Pipeline
Laura Bell, Michael Brunton-Spall, Rich Smith & Jim Bird

Separation of Duties Considered Harmful

VERICA

@WICKETT

DevOps Audit Defense Toolkit: https://cdn2.hubspot.net/hubfs/228391/Corporate/DevOps_Audit_Defense_Toolkit_v1.0.pdf

Risk Management Theater: https://continuousdelivery.com/2013/08/riskmanagement-theatre/

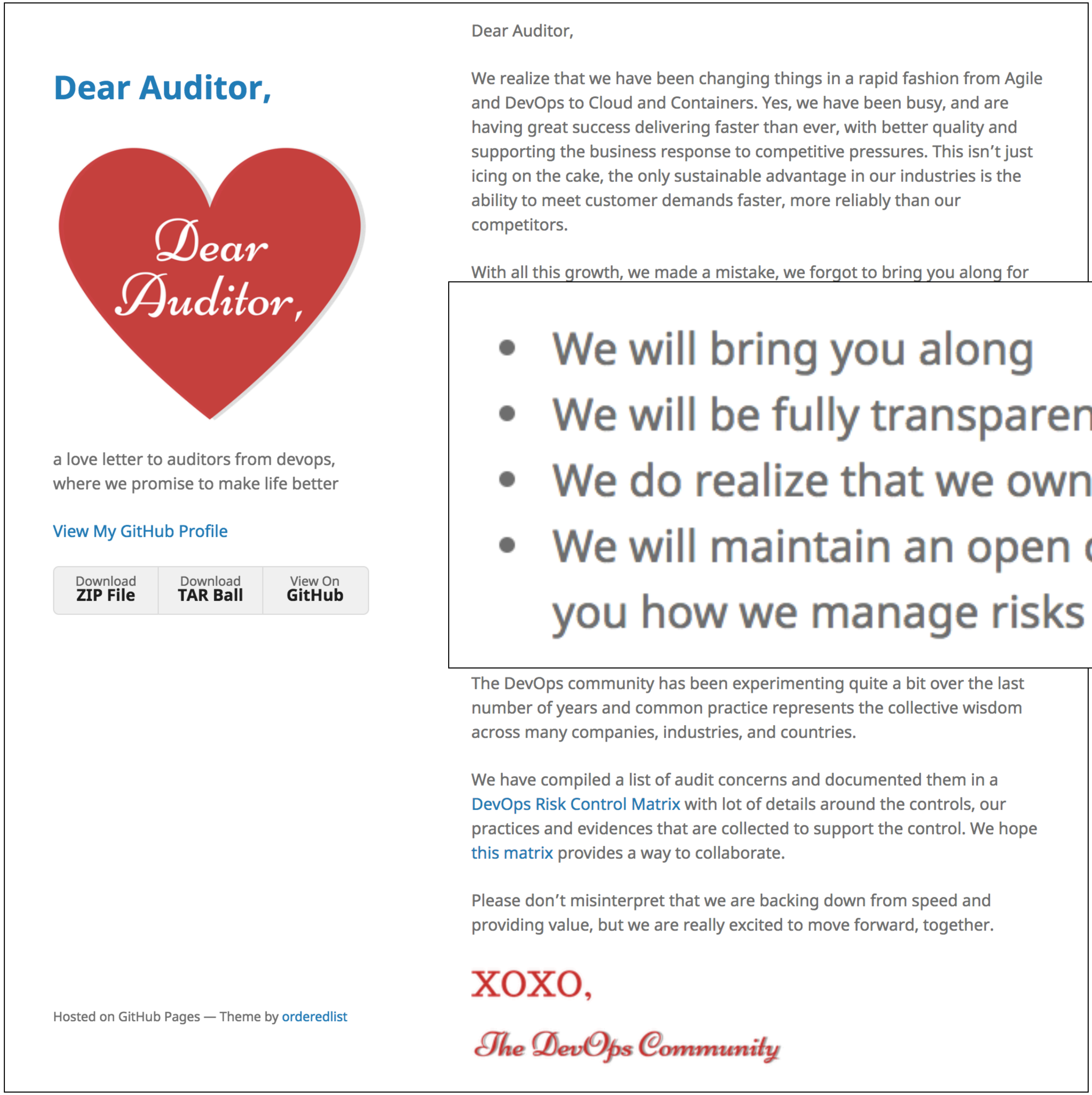
Continuous Delivery and ITIL Change management: https://continuousdelivery.com/2010/11/continuous-delivery-and-itil-changemanagement/

DevOps Kata – deploy a single line of code: http://devopsy.com/blog/2013/08/16/devops-kata-single-line-of-code/

Lean Enterprise Chapter 12: http://shop.oreilly.com/product/0636920030355.do

O'REILLY
Agile Application Security
Enabling Security in a Continuous Delivery Pipeline
Laura Bell, Michael Brunton-Spall, Rich Smith & Jim Bird

source: Jim Bird’s SANS preso



Dear Auditor,

a love letter to auditors from devops, where we promise to make life better

View My GitHub Profile

Hosted on GitHub Pages — Theme by orderedlist

Dear Auditor,

We realize that we have been changing things in a rapid fashion from Agile and DevOps to Cloud and Containers. Yes, we have been busy, and are having great success delivering faster than ever, with better quality and supporting the business response to competitive pressures. This isn't just icing on the cake, the only sustainable advantage in our industries is the ability to meet customer demands faster, more reliably than our competitors.

With all this growth, we made a mistake, we forgot to bring you along for

- We will bring you along
- We will be fully transparent about our development process
- We do realize that we own the risks
- We will maintain an open channel of discussion to demonstrate to you how we manage risks with our modern development practices

The DevOps community has been experimenting quite a bit over the last number of years and common practice represents the collective wisdom across many companies, industries, and countries.

We have compiled a list of audit concerns and documented them in a DevOps Risk Control Matrix with lot of details around the controls, our practices and evidences that are collected to support the control. We hope this matrix provides a way to collaborate.

Please don't misinterpret that we are backing down from speed and providing value, but we are really excited to move forward, together.

XOXO,

The DevOps Community

Develop | Inherit | Build | **Deploy** | Operate

Monitoring Cloud Configuration

- **Paid Cloud Config security:** Evident.io, ThreatStack, AlienVault, and more

- **Cloud Provider:** AWS CloudTrail, Inspector, GuardDuty

Develop | Inherit | Build | **Deploy** | Operate

Questions to Ask

What secrets are needed to move my application from development into production?

Am I testing for Compliance on each and every deploy?

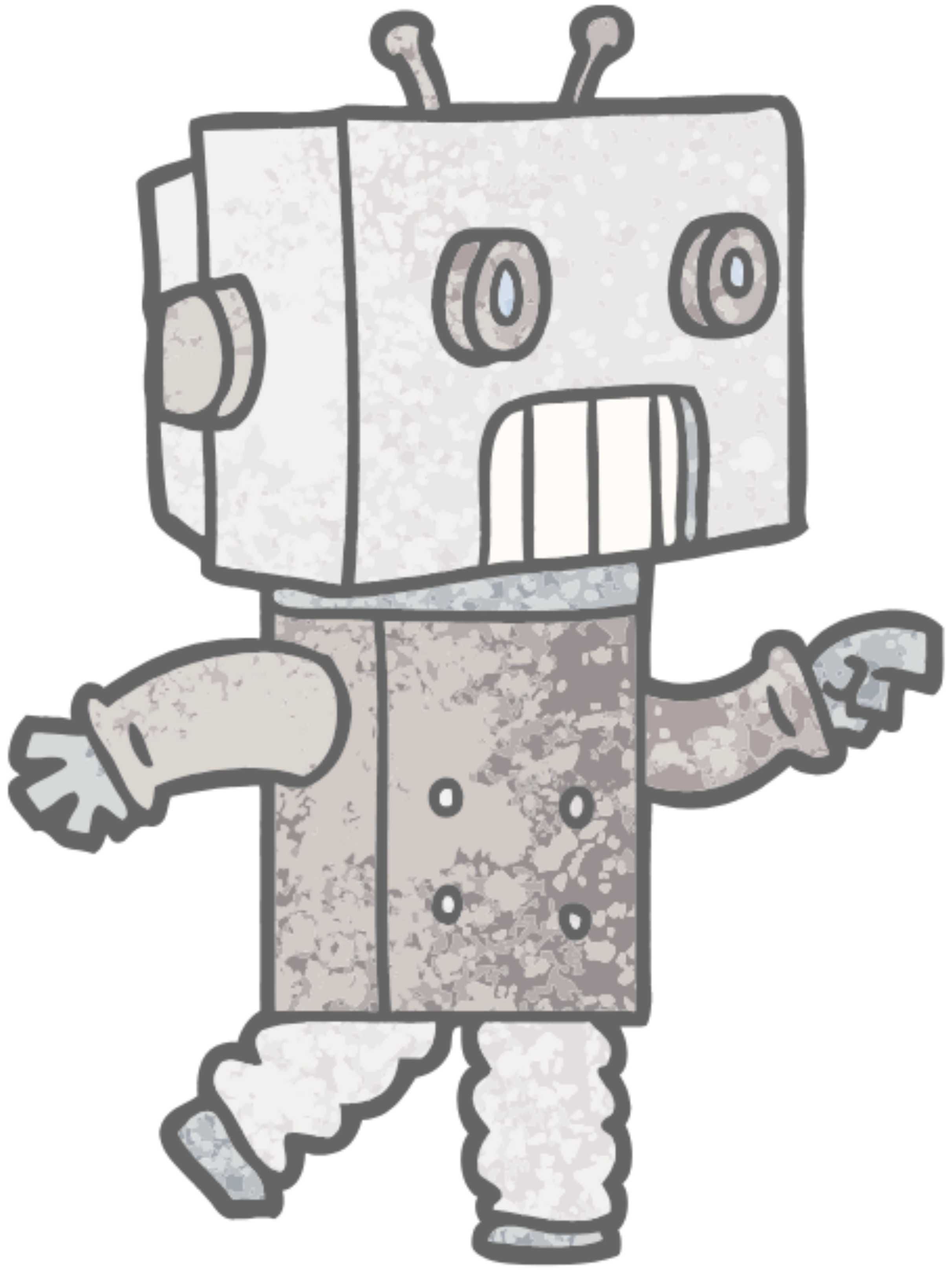
Is there a repeatable mechanism to push changes to production?

Develop | Inherit | Build | Deploy | **Operate**

The runtime state of the application, where users interact with or consume the application. Our application in production.

VAYAGIF.COM

Security in the operate phase is only successful if it creates learning feedback for developers.



Security Considerations

- **Security Chaos Engineering and creating stability through instability**
- **Circuit Breakers and Bulkheads**
- **Instrumentation and Visualization**
- **Application security and service abuse and misuse**
- **Bug Bounties**
- **Red Teaming as a Service**

Detect what matters

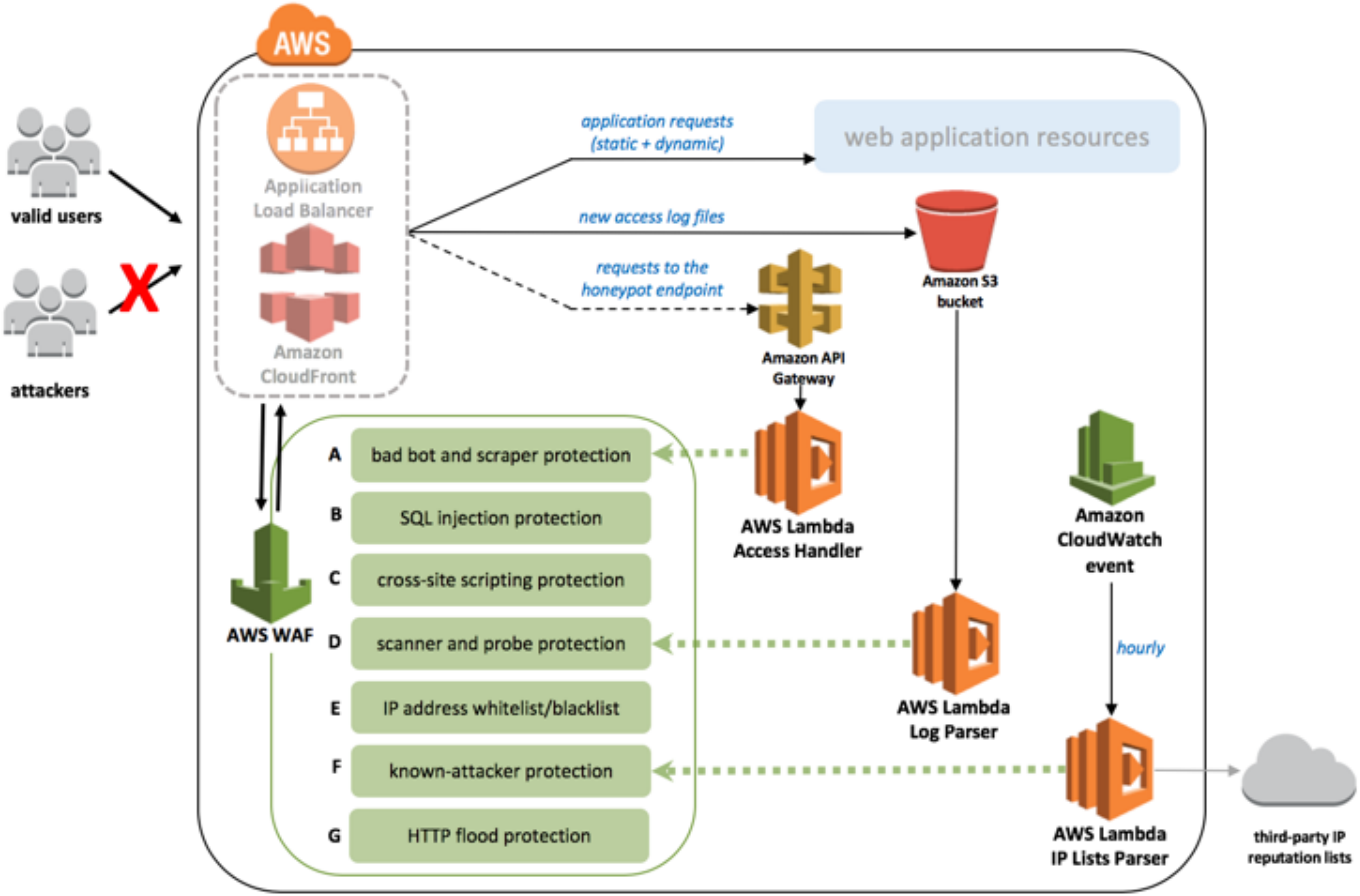
Account takeover attempts

Areas of the site under attack

Most likely vectors of attack

Business logic flows

Abuse and Misuse signals



AWS

valid users

attackers

Application Load Balancer

Amazon CloudFront

application requests (static + dynamic)

web application resources

new access log files

Amazon S3 bucket

requests to the honeypot endpoint

Amazon API Gateway

AWS WAF

A bad bot and scraper protection

B SQL injection protection

C cross-site scripting protection

D scanner and probe protection

E IP address whitelist/blacklist

F known-attacker protection

G HTTP flood protection

AWS Lambda Access Handler

AWS Lambda Log Parser

Amazon CloudWatch event

hourly

AWS Lambda IP Lists Parser

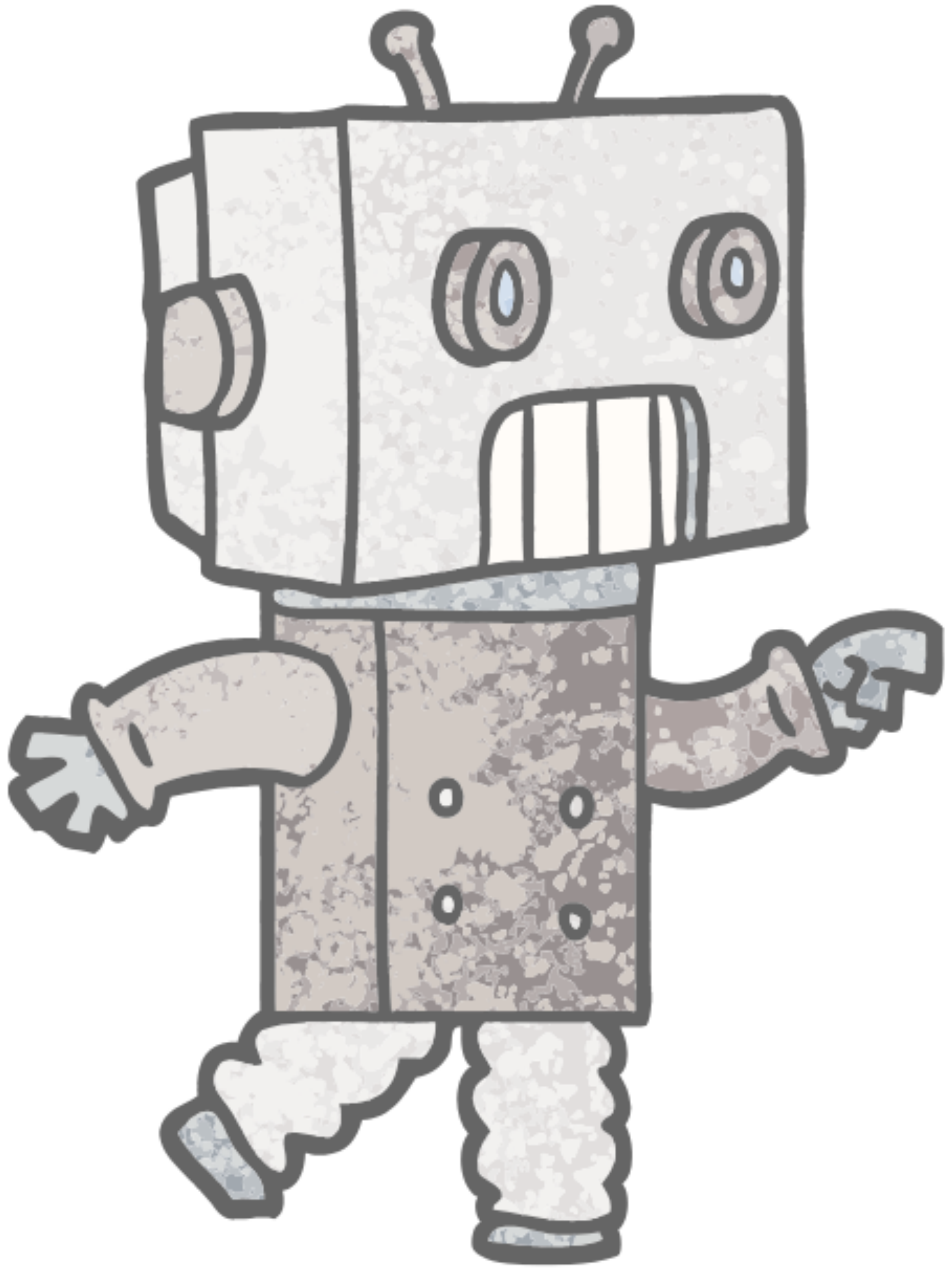
third-party IP reputation lists

Develop | Inherit | Build | Deploy | **Operate**

Runtime Defense

- **Roll your own** (previous slide)
- **Pro-tip: Avoid adding appsec defense at the CDN**
- **Paid NGWAF / RASP Options**

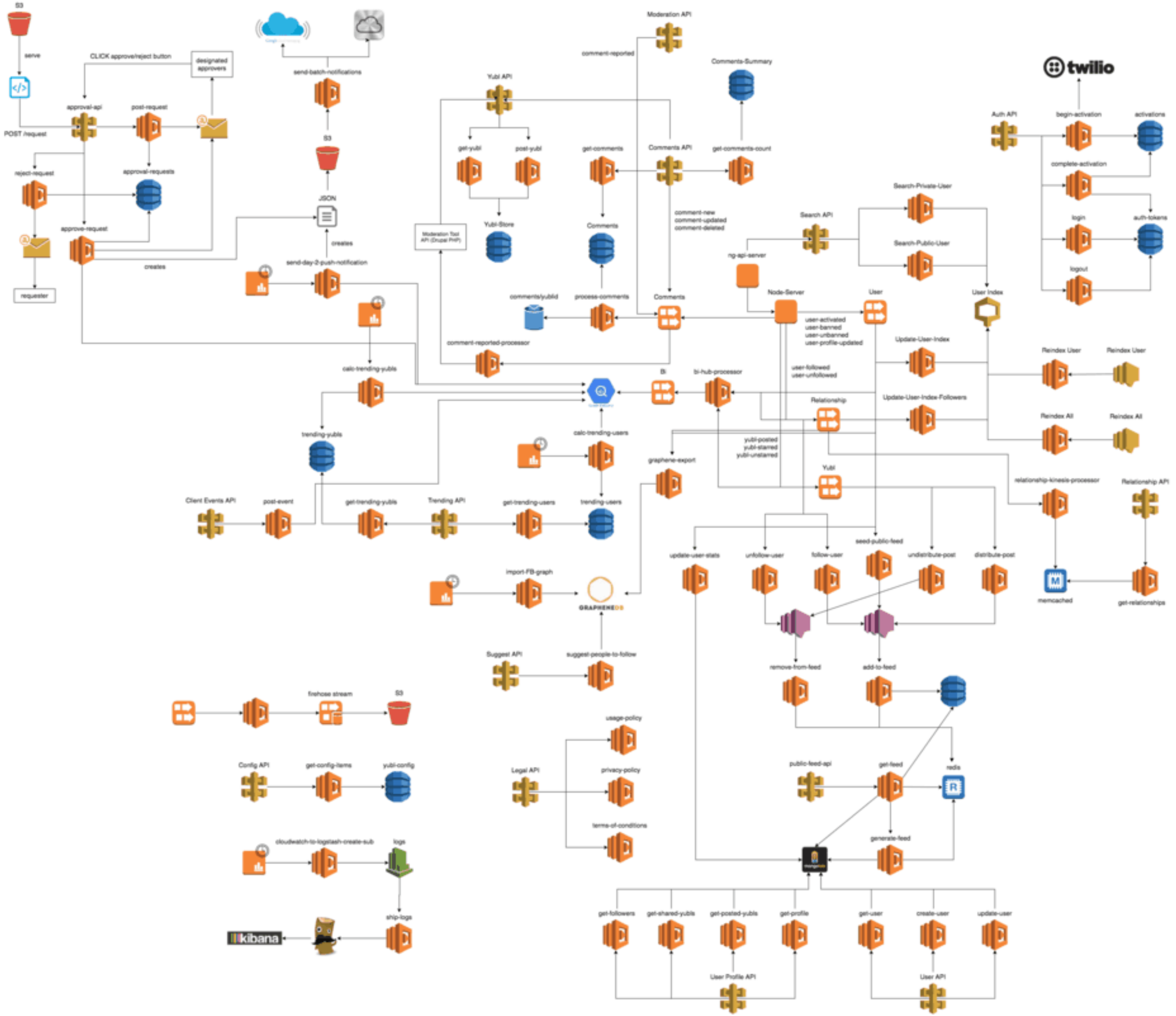
Red Team Mondays
(Intuit does it, so can you)
-Shannon Lietz



VERICA

@WICKETT

THE SHOW IS SUPPOSED TO BE ABOUT ME!
CHAOS ENGINEERING
me
SECURITY CHAOS ENGINEERING


S3
serve
CLICK approve/reject button
designated approvers
approval-api
post-request
POST /request
reject-request
approval-requests
approve-request
requester
creates
send-batch-notifications
S3
JSON
creates
send-day-2-push-notification
calc-trending-yubls
trending-yubls
Moderation API
comment-reported
Comments-Summary
Yubl API
get-yubl
post-yubl
get-comments
Comments API
get-comments-count
comment-new
comment-updated
comment-deleted
Yubl-Store
Comments
Moderation Tool API (Drupal PHP)
comments/yublid
process-comments
Comments
comment-reported-processor
ng-api-server
Search API
Search-Private-User
Search-Public-User
Node-Server
User
User Index
user-activated
user-banned
user-unbanned
user-profile-updated
user-followed
user-unfollowed
Update-User-Index
Update-User-Index-Followers
Relationship
BI
bi-hub-processor
yubl-posted
yubl-starred
yubl-unstarred
Yubl
twilio
Auth API
begin-activation
activations
complete-activation
login
auth-tokens
logout
Reindex User
Reindex User
Reindex All
Reindex All
relationship-kinesis-processor
Relationship API
memcached
get-relationships
calc-trending-users
graphene-export
Client Events API
post-event
get-trending-yubls
Trending API
get-trending-users
trending-users
import-FB-graph
GRAPHENEDB
Suggest API
suggest-people-to-follow
update-user-stats
unfollow-user
follow-user
seed-public-feed
undistribute-post
distribute-post
remove-from-feed
add-to-feed
public-feed-api
get-feed
redis
generate-feed
firehose stream
S3
Config API
get-config-items
yubl-config
cloudwatch-to-logstash-create-sub
logs
ship-logs
kibana
usage-policy
Legal API
privacy-policy
terms-of-conditions
get-followers
get-shared-yubls
get-posted-yubls
get-profile
get-user
create-user
update-user
User Profile API
User API

Develop | Inherit | Build | Deploy | **Operate**

Security Chaos Engineering

The identification of security control failures through proactive experimentation to build confidence in the system's ability to defend against malicious conditions in production.

source: Aaron Rinehart

4 Steps to Security Chaos Engineering

1. Define expected behavior of a security defense

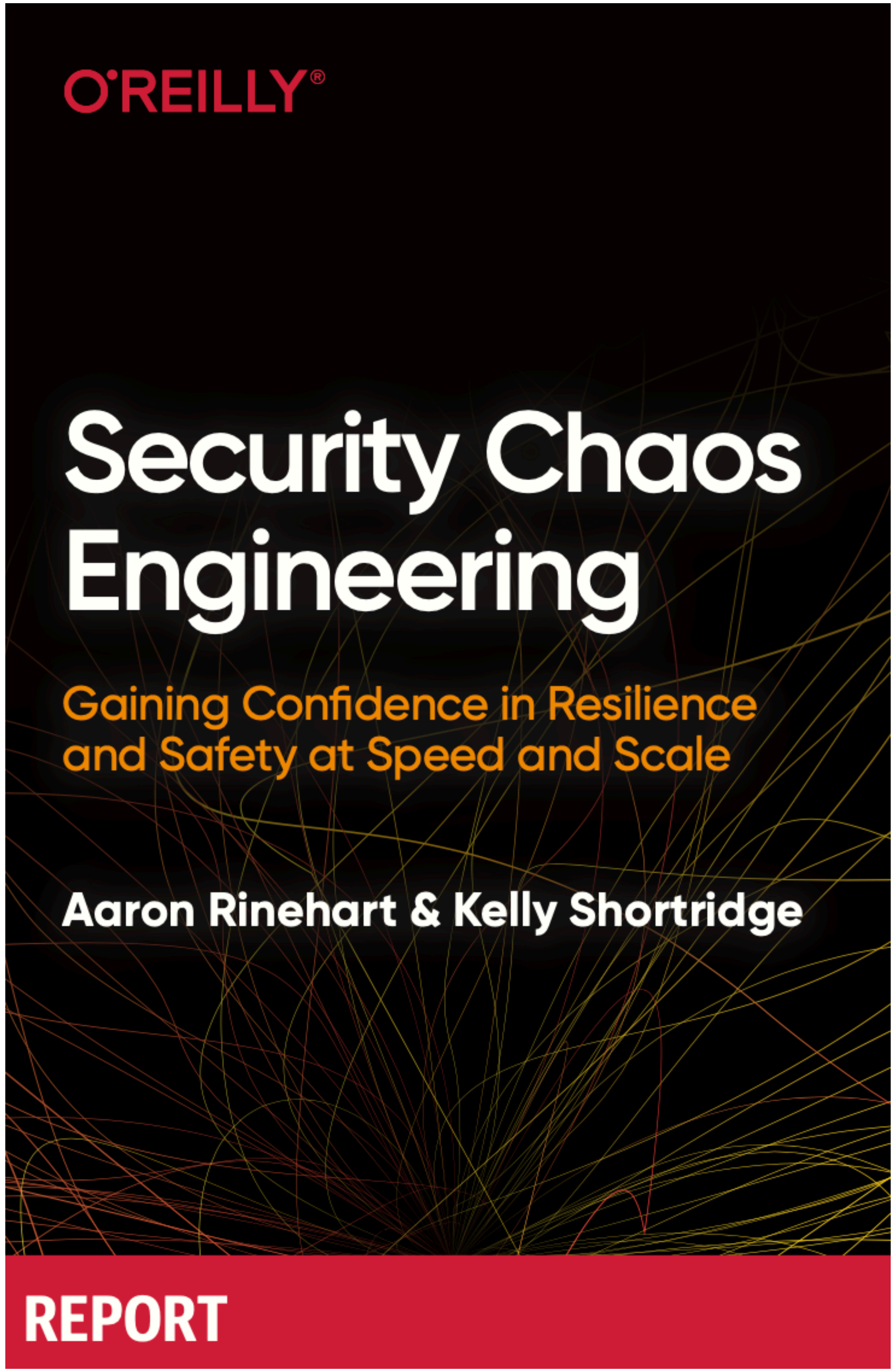
2. Hypothesize that when security turbulence is introduced it will be either prevented, remediated, or detected.

3. Introduce a variable that introduces security turbulence.

4. Try to disprove the hypothesis by looking for a difference in expected behavior and actual behavior

VERICA

@WICKETT



O'REILLY®

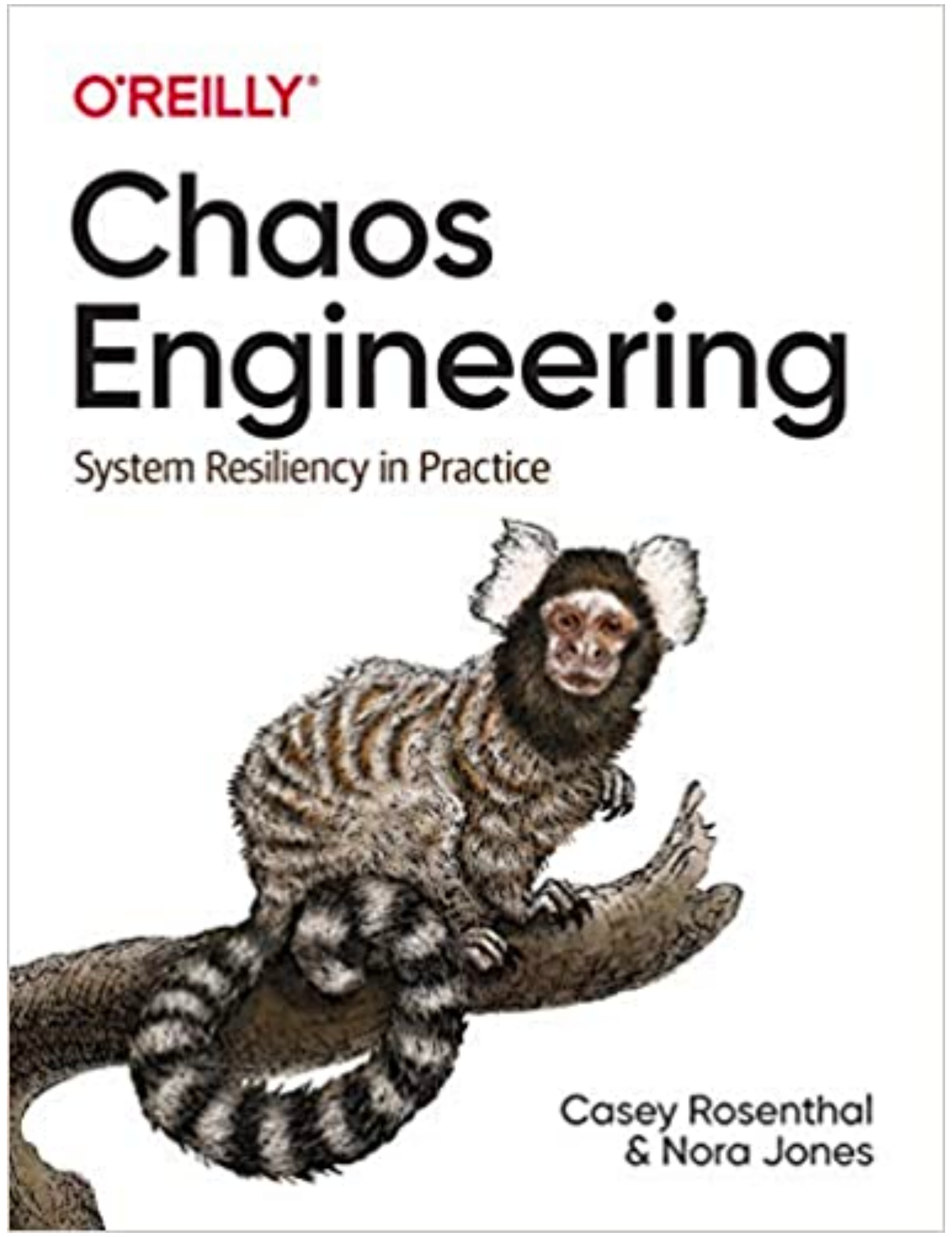
Security Chaos Engineering

Gaining Confidence in Resilience and Safety at Speed and Scale

Aaron Rinehart & Kelly Shortridge

REPORT

verica.io/book



O'REILLY

Chaos Engineering

System Resiliency in Practice

Casey Rosenthal
& Nora Jones

The Pragmatic Programmers

Release It!

Second Edition

Design and Deploy
Production-Ready Software

Michael T. Nygard
Edited by Katharine Dvorak

Develop | Inherit | Build | Deploy | **Operate**

Questions to Ask

Do you know if you are under attack at this current moment?

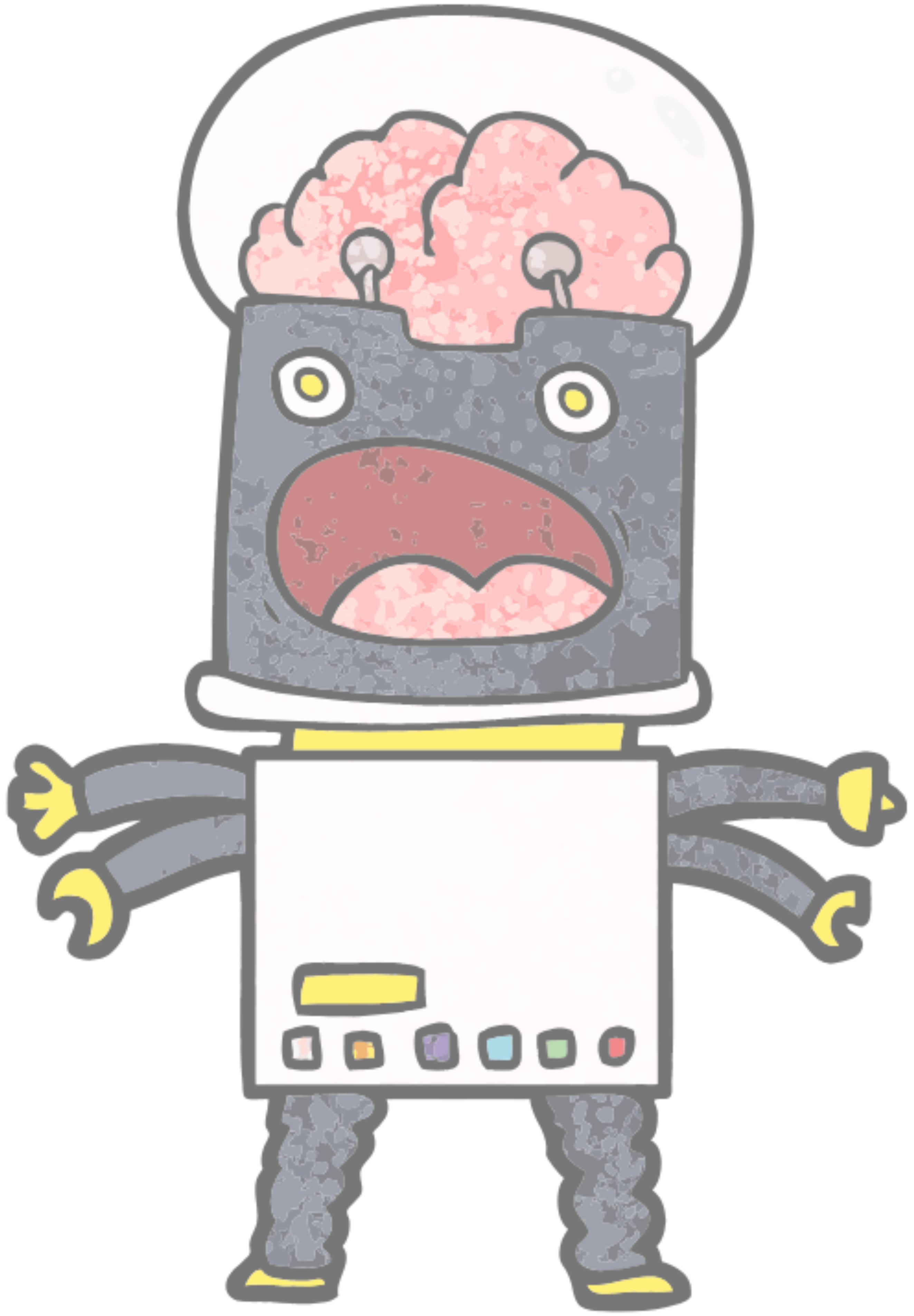
Do you know what the attackers are going after?

Can I turn on and off services independently if being attacked?

Are we doing security chaos experiments?

Stay in touch

wickett@verica.io



VERICA

@WICKETT