Applied Security

Jamie Dicken

Manager, Security Engineering
Cardinal Health
@jamie_dicken

Aaron Rinehart

CTO, Founder
Verica
@aaronrinehart

Crafting Secure and Resilient Distributed Systems using Chaos Engineering

Aaron Rinehart, CTO, Founder

- Former Chief Security Architect @UnitedHealth
- Former DoD, NASA Safety & Reliability Engineering
- Frequent speaker and author on Chaos Engineering & Security
- O'Reilly Author: Chaos Engineering, Security Chaos Engineering Books
- Pioneer behind Security Chaos Engineering
- Led ChaoSlingr team at UnitedHealth

@aaronrinehart @verica_io #chaosengineering

Verica

Jamie Dicken

- Manager, Security Engineering at Cardinal Health
- Former software engineer and software development manager
- 11 years in healthcare
- Speaker on uniting disciplines: Software Engineering and InfoSec, InfoSec and SRE
- Future O'Reilly Contributing Author: Security Chaos Engineering Report

@jamie_dicken

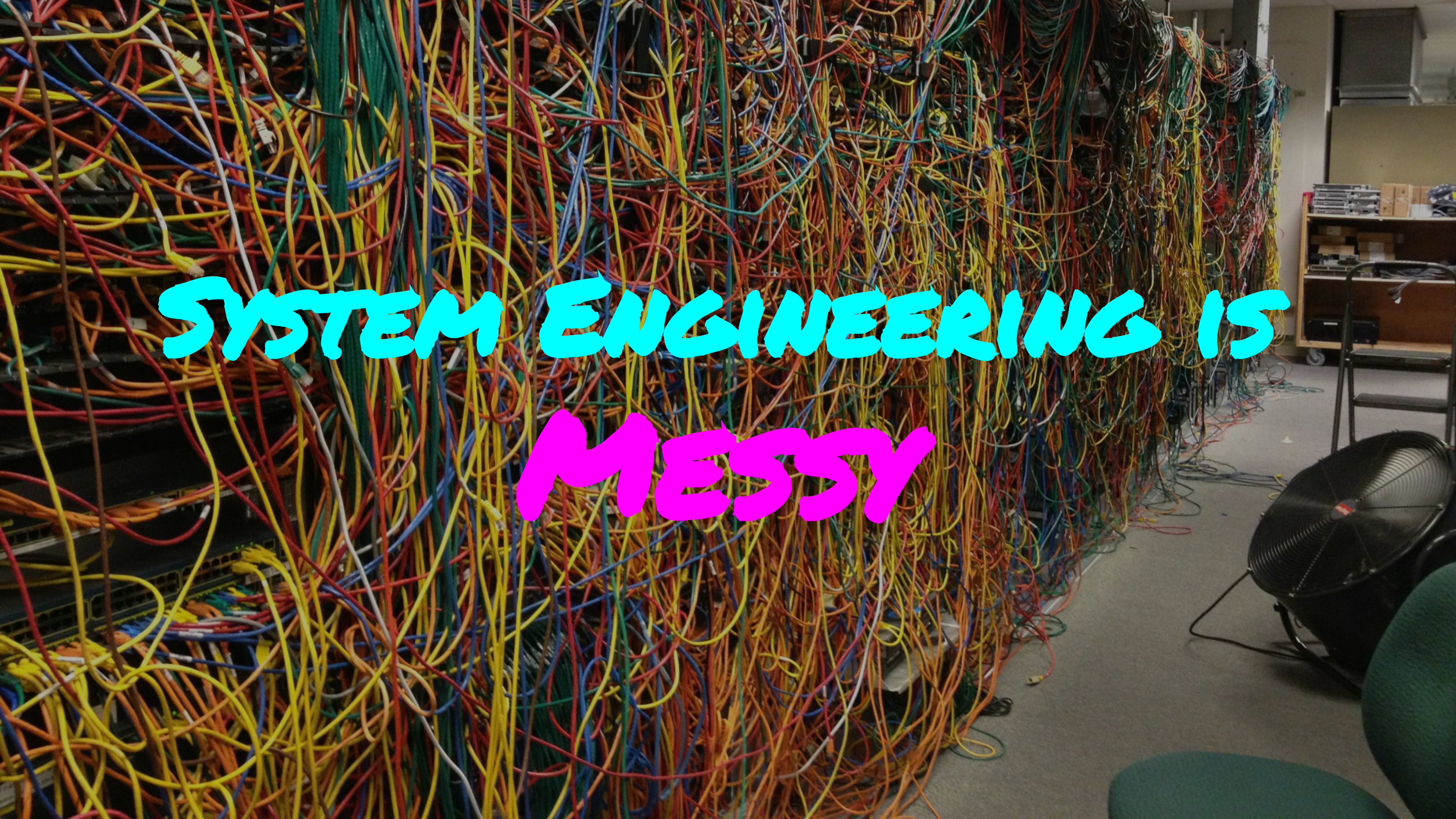
CardinalHealth
Essential to care™

Agenda

New Approaches to Security
Chaos Engineering
Continuous Learning
Security Chaos Engineering
Getting Started with Applied Security
Resilience Engineering & Security

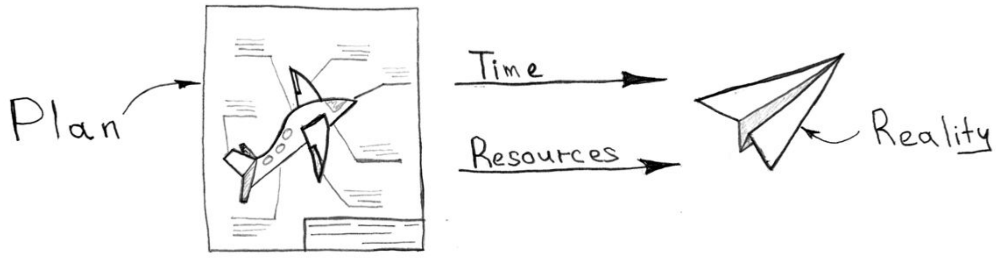
Problem

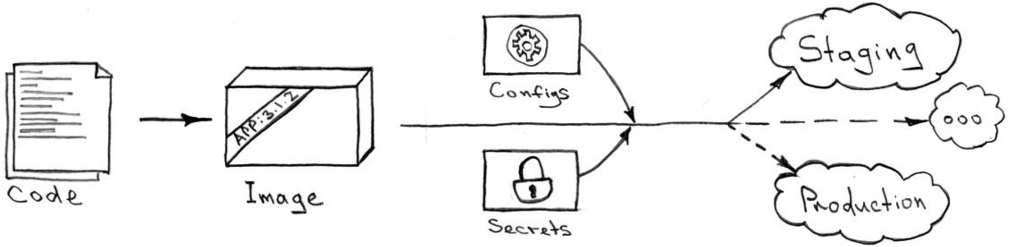
The Struggle is real

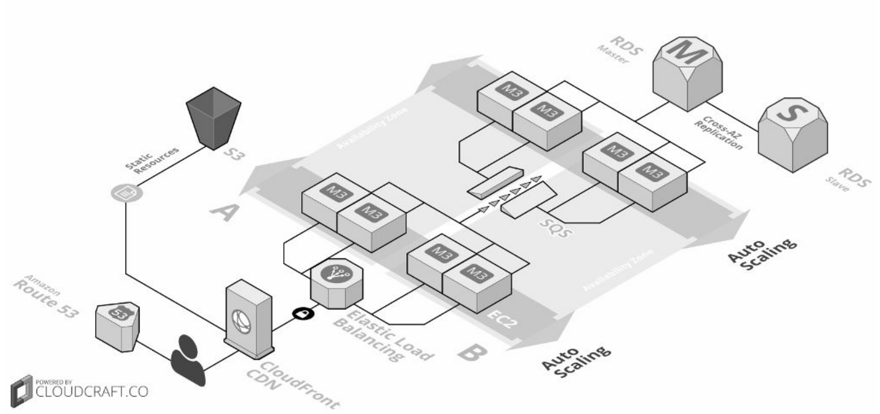


System Engineering is Messy

In the beginning...







After a few months...

Hard Coded Passwords

Network is Unreliable

New Security Tool

Autoscaling Keeps Breaking

Identity Conflicts

Regulatory Audit

Refactor Pricing

Rolling Sev1 Outage on Portal

Lead Software Engineer finds a new job at Amazon

Cloud Provider API Outage

Code Freeze

DNS Resolution Errors

Expired Certificate

300 Microservices Δ-> 850 Microservices

Scalability Issues

WAF Outage -> Disabled

Delayed Features

Large Customer Outage

Years?

Orphaned Documentation

Portal Retry Storm Outage

Regulatory Audit

Rolling Sev1 Outages on Portal

Code Freeze

Budget Freeze

Hard Coded Passwords

New Security Tool

Corporate Reorg

Identity Conflicts

Migration to New CSP

Refactor Pricing

Lead Software Engineering finds a new job at Amazon

Upgrade to Java SE 12

Expired Certificate

300 Microservices Δ-> 850 Microservices

Scalability Issues

Delayed Features

Hard Coded Passwords

New Security Tool

Identity Conflicts

Lead Software Engineering finds a new job at Google

Expired Certificate

Database Outage

Network is Unreliable

Scalability Issues

300 Microservices Δ-> 4000 Microservices

Delayed Features

Misconfigured FW Rule Outage

Cloud Provider API Outage

DNS Resolution Errors

WAF Outage -> Disabled

Large Customer Outage

Network is Unreliable

Autoscaling Keeps Breaking

Refactor Pricing

Cloud Provider API Outage

DNS Resolution Errors

Outsource overseas development

Autoscaling Keeps Breaking

Firewall Outage -> Disabled

Large Customer Outage

Exposed Secrets on GitHub

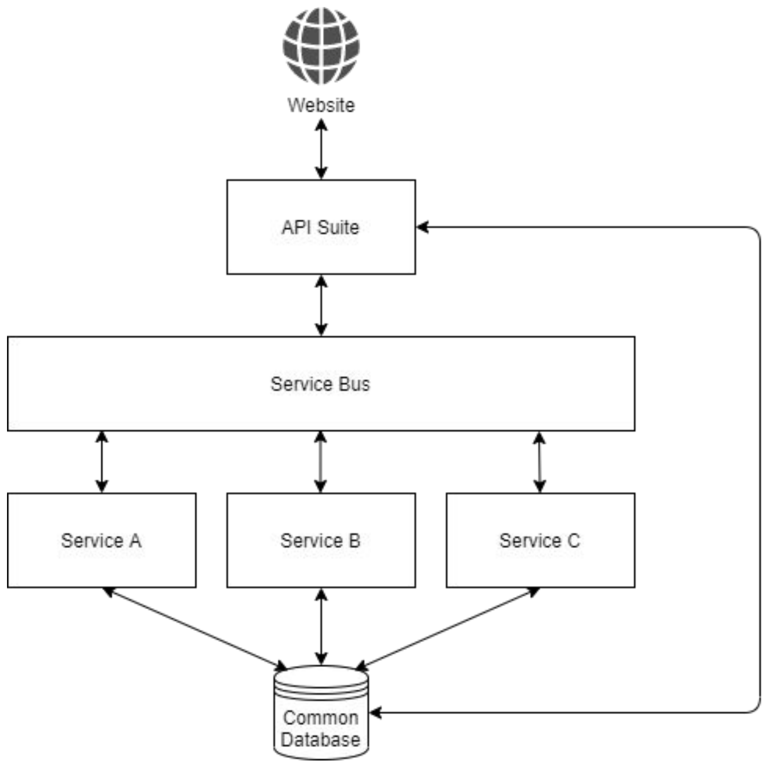
Merge with competitor

Code Freeze

Regulatory Audit

Rolling Sev1 Outage on Portal

The Design-Oriented Mindset is Old-School



Documentation stored in team archives

Oh yeah... Forgot about these!

- There is an API Gateway in place for some APIs, but not the legacy ones.
- Some APIs are publicly accessible and used by our customers directly.
- There's a monthly batch process that runs directly on the database and saves to an SFTP directory.
- Not all microservices are independent. Some level of synchronicity is required.

"The only way to understand a complex system is to interact with it."

--Dave Snowden

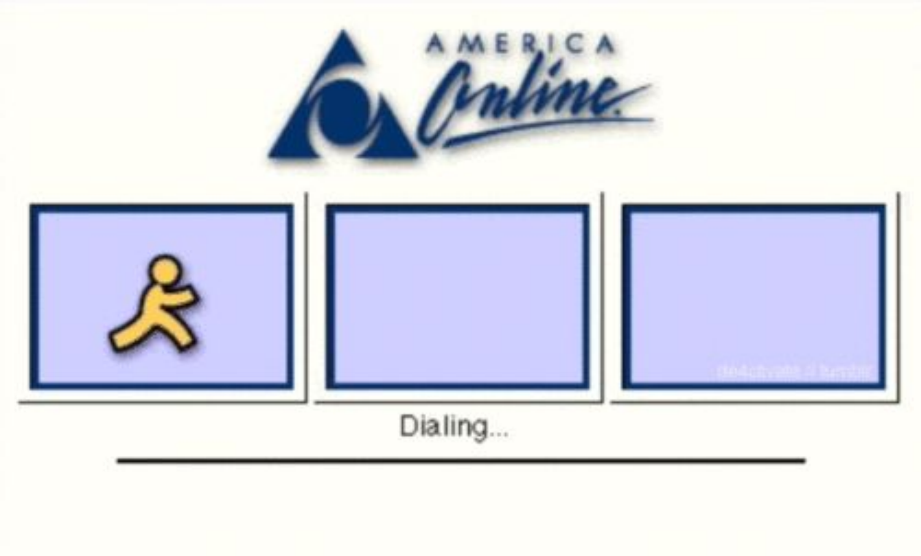
A New Approach to Learning

Continuous Learning

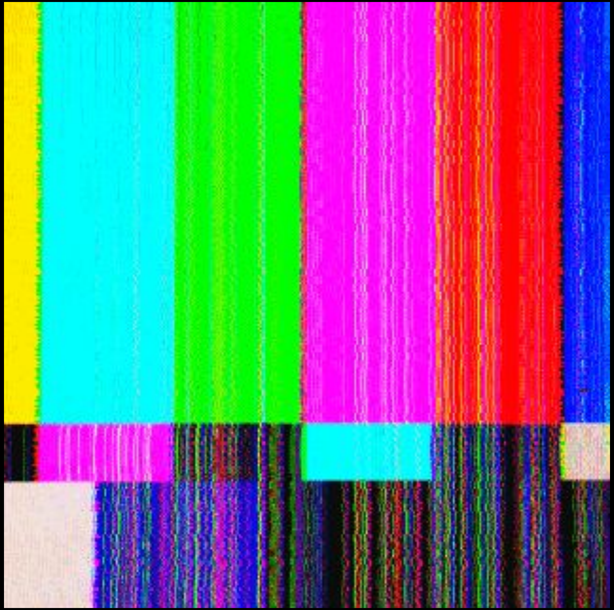
Continuous Fixing != Continuous Learning

How does a System become stable?

A Change in Mindset



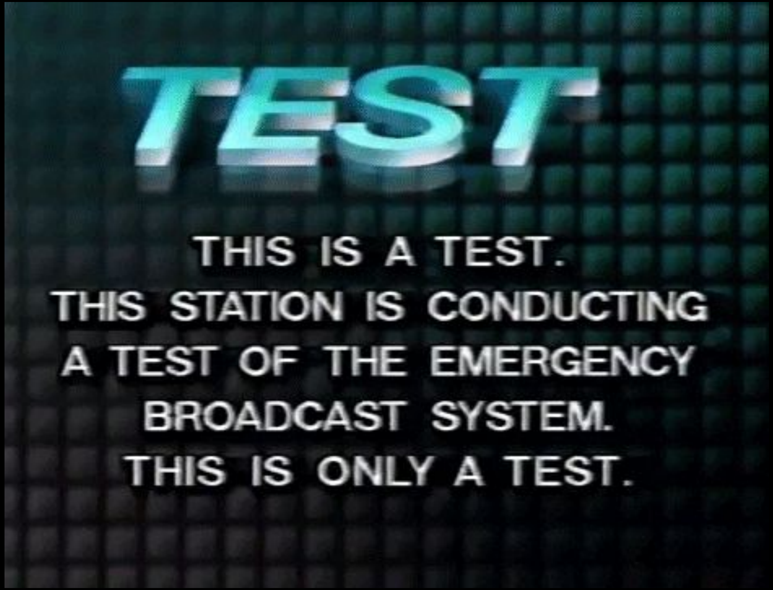
People <u>**Operate Differently**</u> when they expect things to fail



Cognitive Load & Tradeoffs Under Pressure

Instrumenting Chaos

Testing vs. Experimentation



TEST

THIS IS A TEST.
THIS STATION IS CONDUCTING
A TEST OF THE EMERGENCY
BROADCAST SYSTEM.
THIS IS ONLY A TEST.

Chaos Engineering

Is about establishing order from Chaos

SECURITY

CHAOS ENGINEERING

Hope is <u>Not</u> an Effective Strategy

"It worked in Star Wars but it won't work here"

"Understand your system and where its security gaps are before an adversary does"

WE OFTEN MISREMEMBER WHAT OUR SYSTEMS REALLY ARE, AND AS A RESULT THE OPPORTUNITY FOR ACCIDENTS & MISTAKES INCREASES

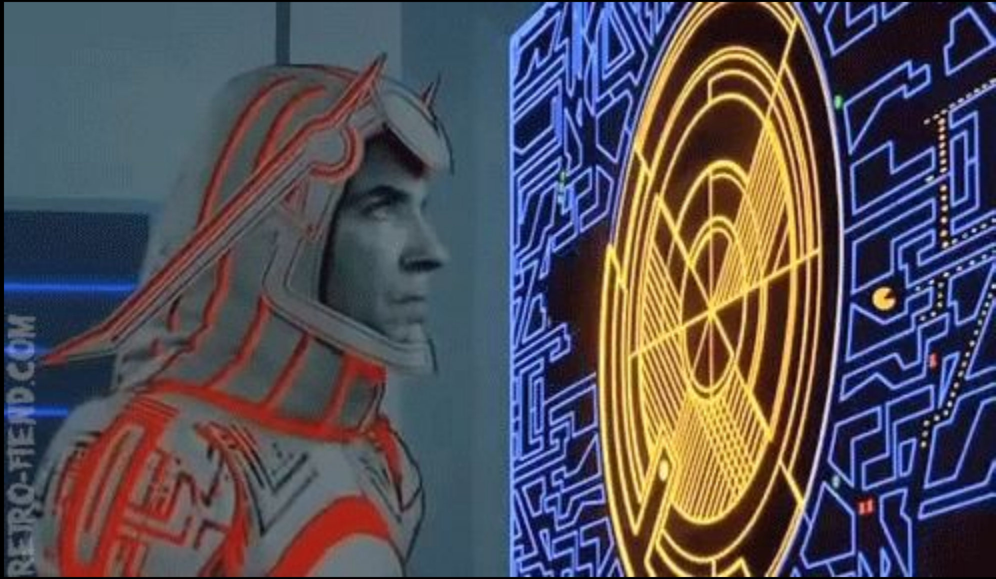
Continuous Security Verification

Reduce Uncertainty by Building Confidence in how the system actually functions

Use Cases

Use Cases

- Incident Response
- Security Control Validation
- Security Observability
- Compliance Monitoring



Incident

Response

"Response" is the problem with incident response.

Security Incidents are Subjective

No matter how much we prepare…

We really don't know very much

Where? Why? Who?

How? What?

Flip the Model

Post Mortem = Preparation

ChaoSlingr

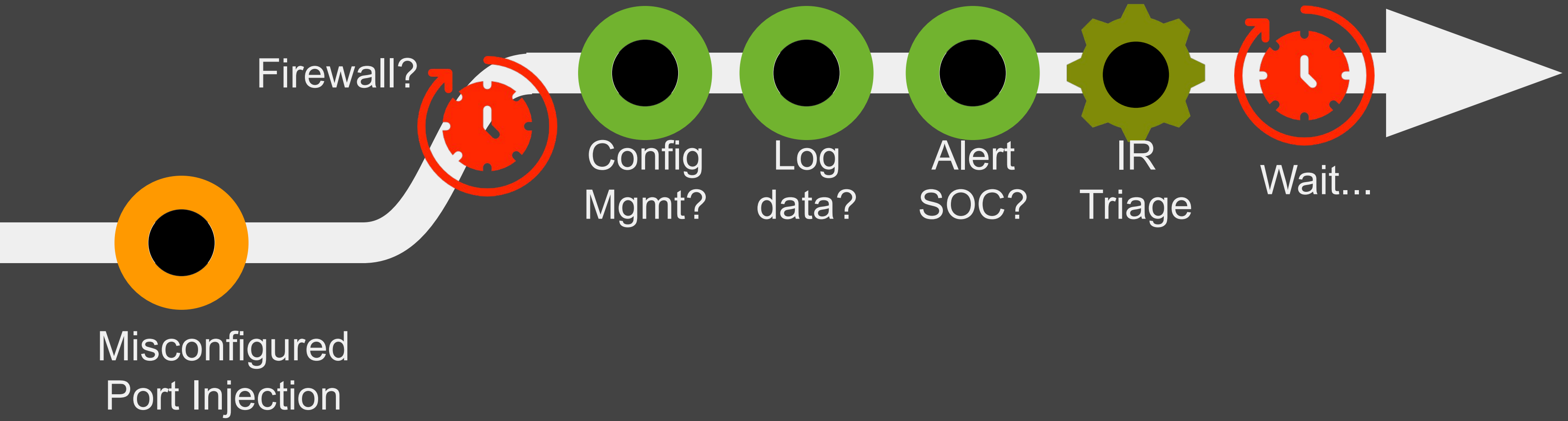
- ChatOps Integration
- Configuration-as-Code
- Example Code & Open Framework

- Serverless App in AWS
- 100% Native AWS
- Configurable Operational Mode & Frequency
- Opt-In | Opt-Out Model

slack

HashiCorp Terraform

amazon web services™



Misconfigured Port Injection

Firewall?

Config Mgmt?

Log data?

Alert SOC?

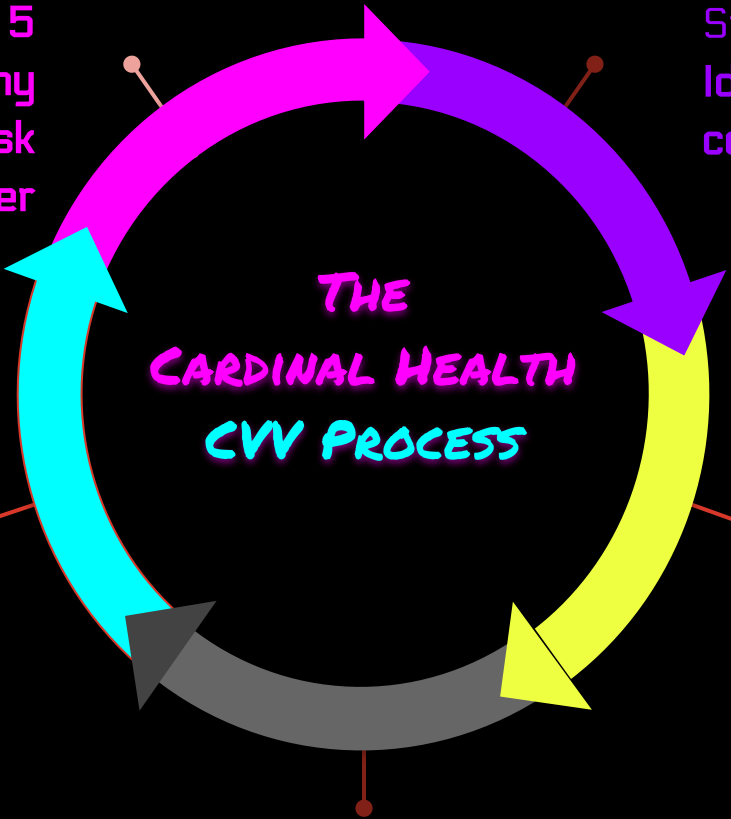
IR Triage

Wait...

Hypothesis: If someone accidentally or maliciously introduced a misconfigured port then we would immediately detect, block, and alert on the event.

Applied Security Journey

The Cardinal Health foray into Security Chaos Engineering



The Cardinal Health CVV Process

Step 1
Identify security control to validate

Step 2
Establish benchmarks to measure compliance

Step 3
Implement continuous checks to validate benchmarks

Step 4
Create a dashboard to illustrate real-time compliance

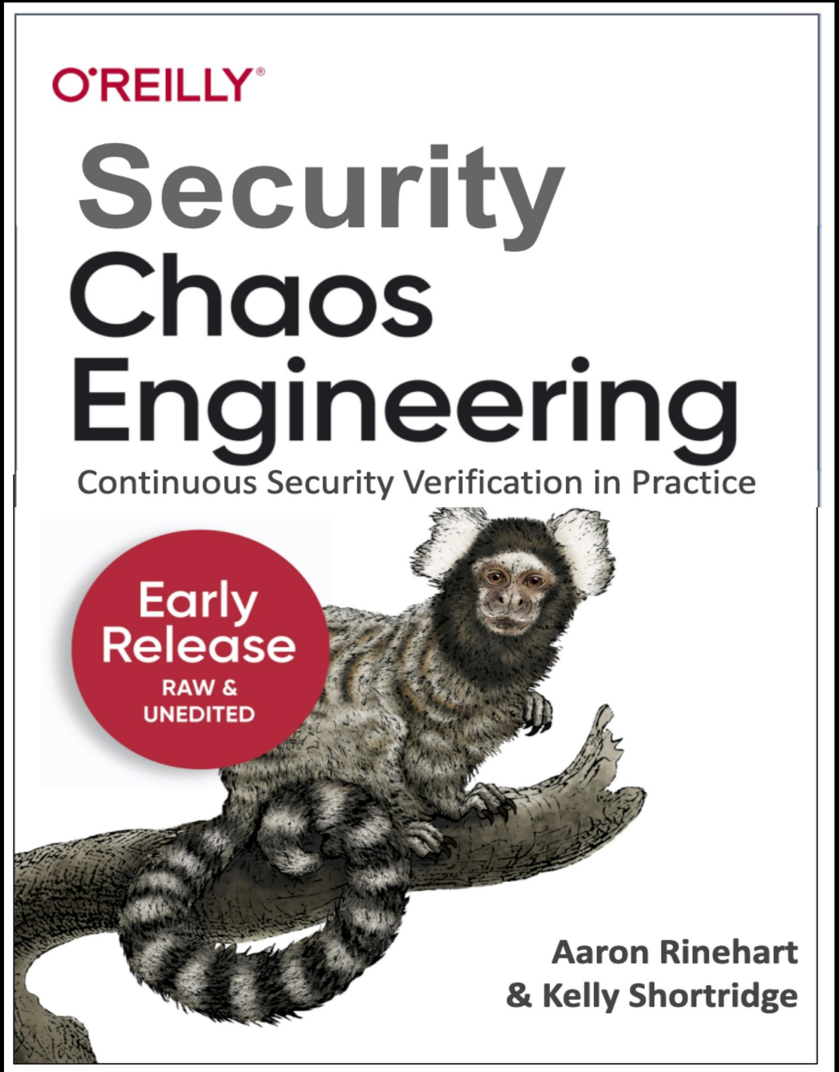
Step 5
Document any non-compliance in Risk Register

Getting Started

with

Security Chaos Engineering

Coming
later
this year...



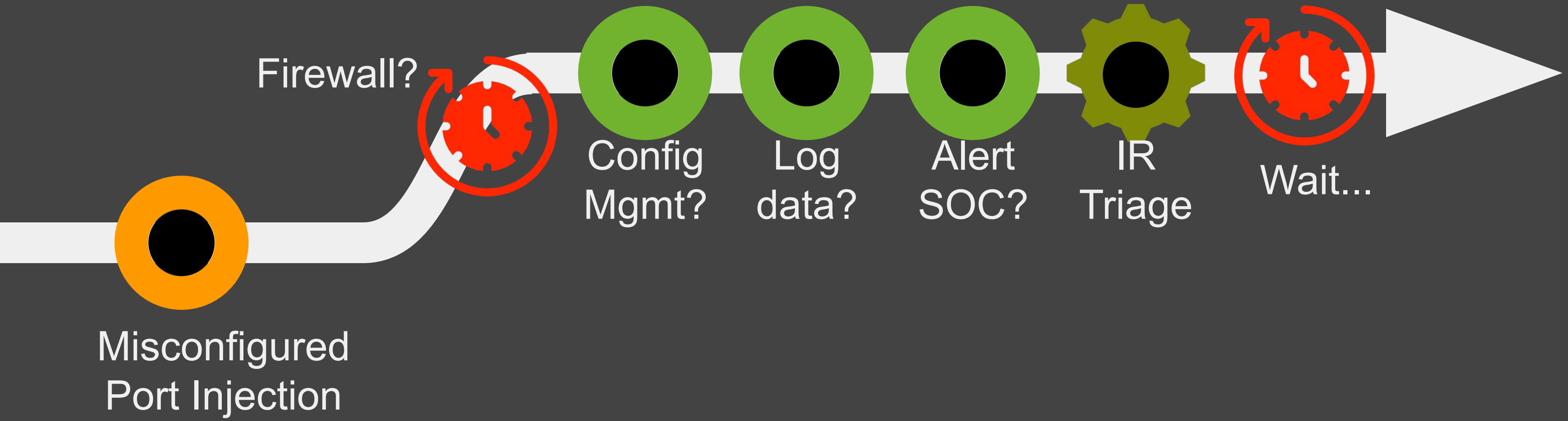
O'REILLY

Security Chaos Engineering

Continuous Security Verification in Practice

Early Release
RAW &
UNEDITED

Aaron Rinehart
& Kelly Shortridge



Firewall?

Config Mgmt?

Log data?

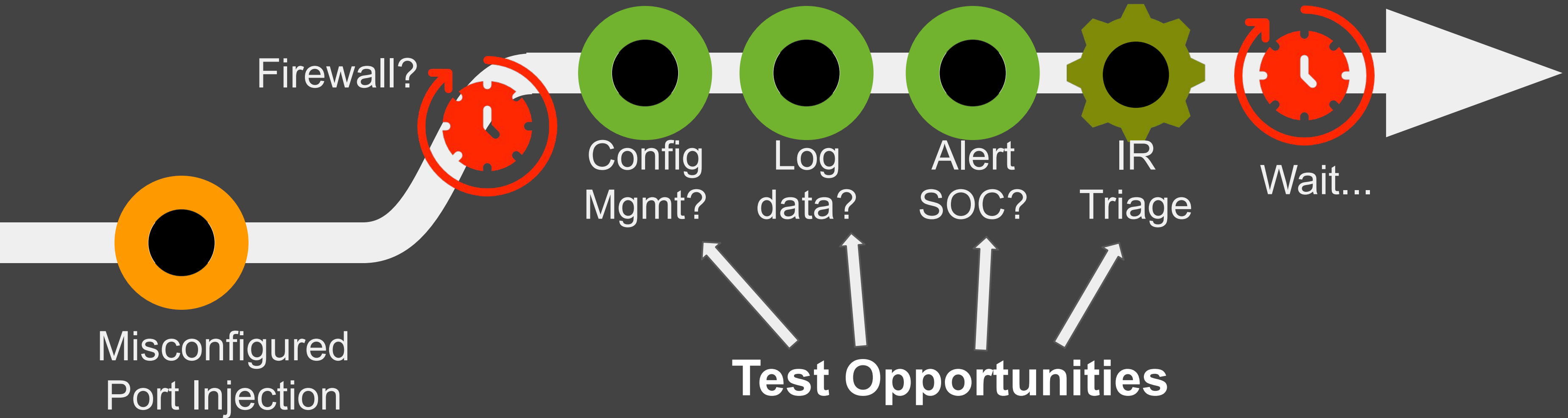
Alert SOC?

IR Triage

Wait...

Misconfigured Port Injection

Hypothesis: If someone accidentally or maliciously introduced a misconfigured port then we would immediately detect, block, and alert on the event.



Misconfigured Port Injection

Firewall?

Config Mgmt?

Log data?

Alert SOC?

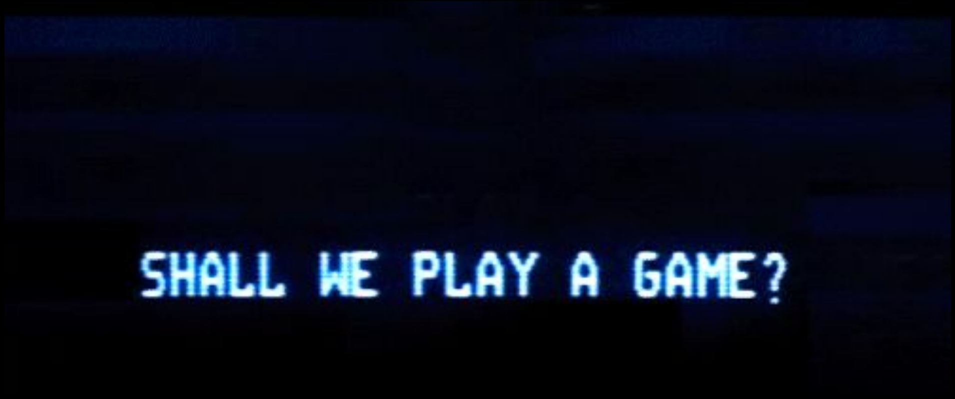
IR Triage

Wait...

Test Opportunities

Hypothesis: If someone accidentally or maliciously introduced a misconfigured port then we would immediately detect, block, and alert on the event.

Security + Resilience Engineering



SHALL WE PLAY A GAME?

The Case for Security Chaos Engineering

Conclusion