A CHAOTIC CAUSE FOR DETECTION ENGINEERING

Applying the principles of Security Chaos Engineering to architect Resilience into your Detection Posture

Dayspring Johnson

Texas Cyber Summit
September 23rd 2022

$~: WHOAMI

ABISOLA DAYSPRING JOHNSON (DAY)
Detection Engineer, Datadog

DATADOG

- Detection Engineer at Datadog (Cloud SIEM Product)
- Information Technology at WGU
- Cybersecurity Content Creator on Youtube (13k+ Subscribers)
- Founder & Community Manager at Cyberwox Academy
- AWS Community Builder
- Cybersecurity Tutor

@daycyberwox | Day Johnson | daycyberwox.com | cyberwoxacademy.com

AGENDA

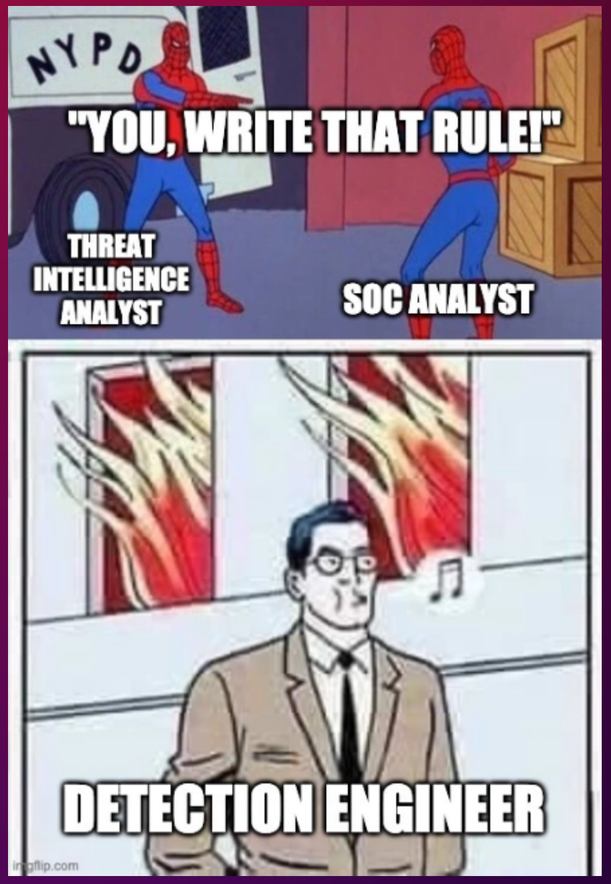
A crash course on How to Chaos

1. Traditional Threat Detection
2. Chaos Engineering
3. Security Chaos Engineering
4. How to Chaos
5. Why we need chaos
6. How to scale chaos
7. A Chaotic Cause for Detection Success

Source: https://www.nytimes.com/2016/08/06/arts/this-is-fine-meme-dog-fire.html

TRADITIONAL THREAT DETECTION

- Surface Level Atomic Rules & Signatures
- Traditional IDS/IPS Systems
- Lack of Automation
- Lack of Testing
- Lack of Resiliience
- Set & Forget it mindset



Source: https://twitter.com/cyb3rops/status/1405198218745495557?s=20&t=8eQ_4ituMoG-TJMNF4uFLgl

IT'S NOT WORKING

Surface Level Atomic Rules & Signatures

Allows for blind spots and easy evasion

Traditional IDS/IPS Systems

Limited capabilities

Lack of Automation

Slower response & more investigation overhead

Set & Forget it mindset

No room for improvement

Lack of Testing

You simply don't know if it works

Lack of Resilience

Lack of confidence in detection efficacy and fidelity

MOVING THREAT DETECTION

TOWARDS ENGINEERING

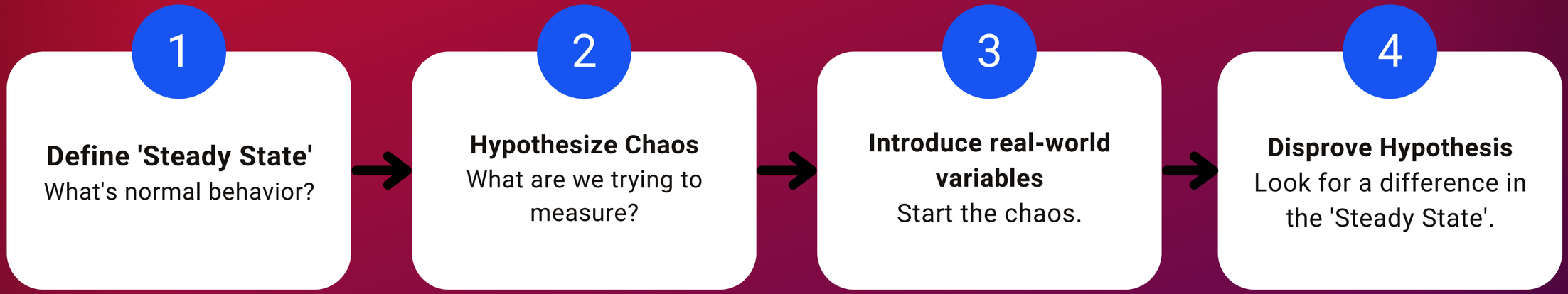
CHAOS ENGINEERING

> Chaos Engineering is the discipline of experimenting on a system in order to build confidence in the system's capability to withstand turbulent conditions in production.

Source: https://principlesofchaos.org

CHAOS IN PRACTICE

STAGES OF A CHAOS EXPERIMENT



The harder it is to disrupt the **steady state**, the more confidence we have in our systems. However, if the **steady state** is a disturbed then that's an area of improvement to focus on before that occurence surfaces at large.

Source: https://principlesofchaos.org

WHAT GOOD CAN COME FROM

CHAOS + SECURITY = ?

SECURITY CHAOS ENGINEERING

> Security Chaos Engineering is 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, "Security Chaos Engineering: A New Paradigm for Cybersecurity," Opensource.com, January 24, 2018, https://opensource.com/article/18/1/new-paradigm-cybersecurity

HOW TO CHAOS

APPLYING THE PRINCIPLES OF SECURITY CHAOS ENGINEERING

1. **Define 'Steady State'**
   What's normal behavior?
2. **Design your hypothesis**
   How's this detection supposed to work?
3. **Contain the blast radius**
   Consider the impact of your experiments.
4. **Measure Impact**
   What detections failed and why?

The more actionable our findings are, the more confidence we have in our detections. However, if blind spots are uncovered then that's an area of improvement to focus on before actual malicious activity surfaces at large.

Source: Aaron Rinehart & Kelly Shortridge. *Security Chaos Engineering*. O'Reilly Media, 2021.

4

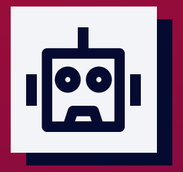
Measure Impact
What detections failed and why?

MEASURING IMPACT

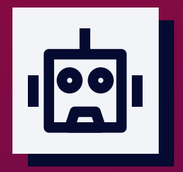
ACTIONABLE FINDINGS & DETECTION IMPROVEMENTS

WHY WE NEED CHAOS

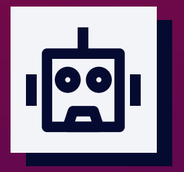
Applying the principles of security chaos engineering gives us a **framework** to architect resilience and improve the confidence in our detections.



Detection Validation



Detection Resilience



Detection -in-depth

DETECTION VALIDATION

TRUST BUT VERIFY

Objectives

- Verify that a detection works as intended
- Reduce gaps caused by false positives/negatives

How

- Automated validation & testing - Datadog's Threatest
- Manual Testing - replicating and validating desired attack scenarios, tactics or techniques

Source: https://securitylabs.datadoghq.com/articles/threatest-end-to-end-testing-threat-detection/

DETECTION RESILIENCE

PLANNING FOR FAILURE

Chaotic situations

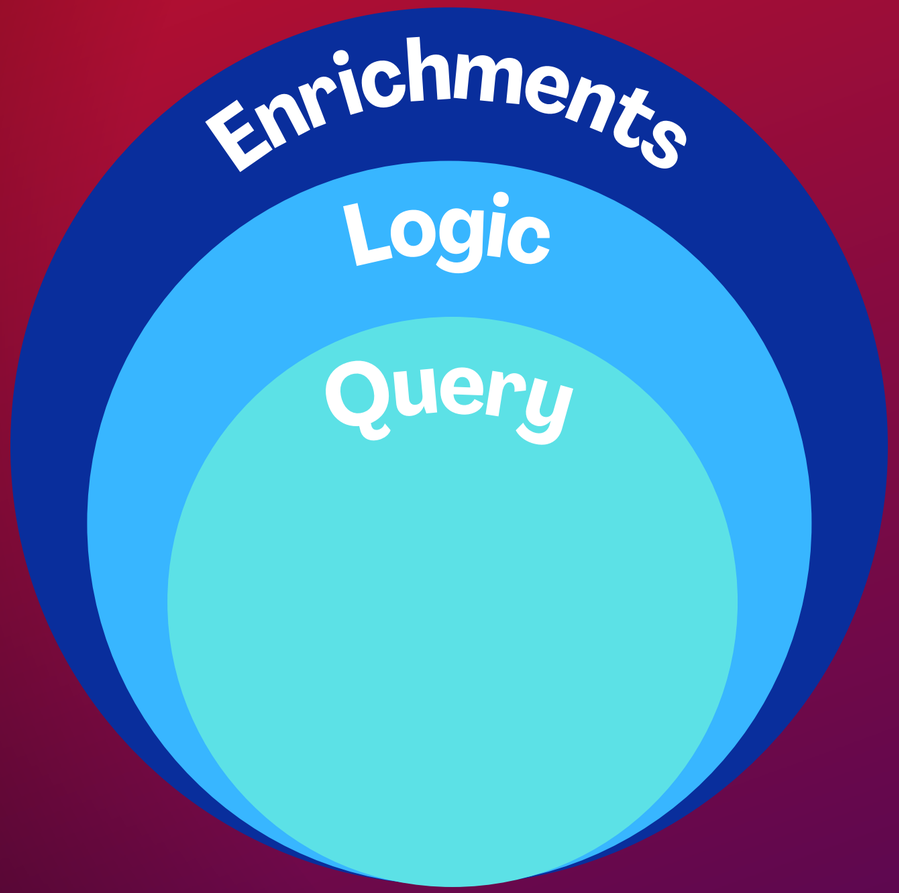
- Log pipeline breaks
- Vendor changes pricing model
- Query/Logging Quotas get exceeded
- Log format changes
- Team/Organizational Changes

Source: https://www.pinterest.co.uk/pin/843510205203462688/

HOW DETECTION-AS-CODE HELPS WITH DETECTION RESILIENCE

DETECTION-AS-CODE

DETECTION BUILDING BLOCKS



Query

The foundation of your detection. Specific attributes & parameters that define the adversary's activity.

Logic

Defines the methodology of your detection. Here you define rules, thresholds, baselines and logic operators (AND/OR).

Enrichments

Additions like metadata, tagging, exclusions, suppressions, alerting, threat intel and more are considered enrichments.

DETECTION-AS-CODE WITH SIGMA

DEVICE REGISTERATION OR JOIN WITHOUT MFA

```yaml
title: Device Registration or Join Without MFA
id: 5afa454e-030c-4ab4-9253-a90aa7fcc581
description: Monitor and alert for device registration or join events where MFA was not performed.
author: Michael Epping, '@mepples21'
date: 2022/06/28
references:
  - https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/security-operations-devices#device-registrations-and-joins-outside-policy
logsource:
  product: azure
  service: signinlogs
detection:
  selection:
    ResourceDisplayName: 'Device Registration Service'
    conditionalAccessStatus: 'success'
  filter_mfa:
    AuthenticationRequirement: 'multiFactorAuthentication'
  condition: selection and not filter_mfa
falsepositives:
  - Unknown
level: medium
status: experimental
tags:
  - attack.valid_accounts
  - attack.t1078
```

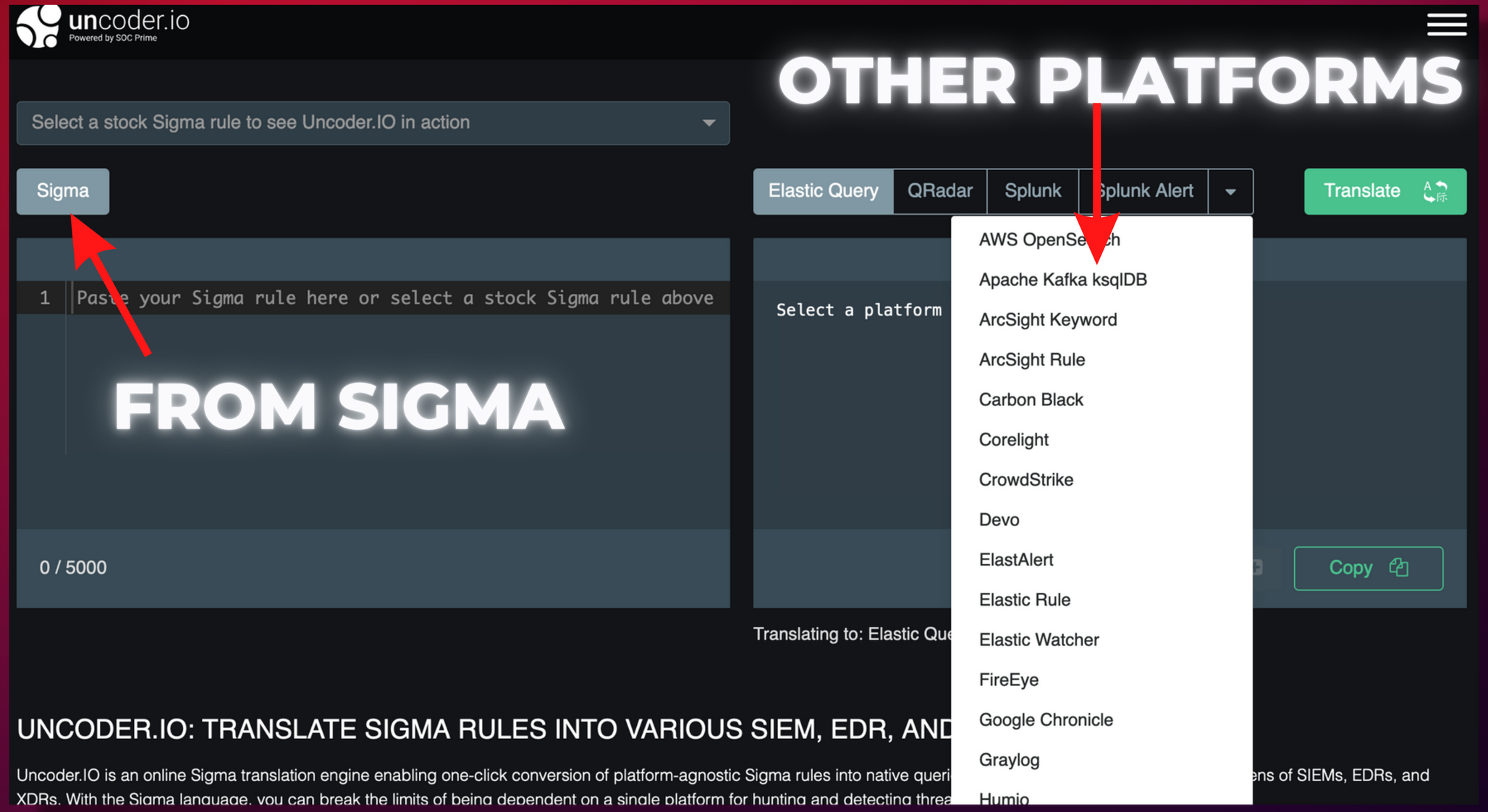
ENRICHMENTS

QUERY

LOGIC

ENRICHMENTS

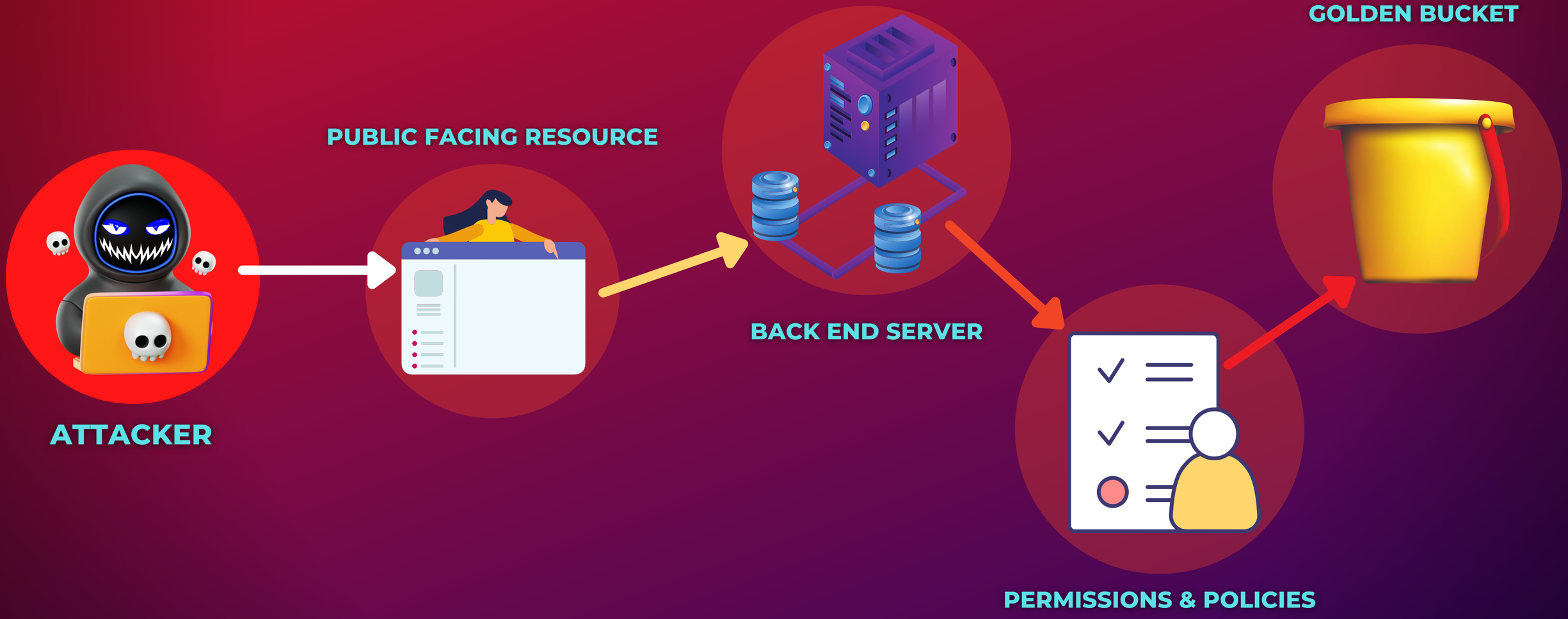
DETECTION RESILIENCE WITH SIGMA



Source: https://uncoder.io/

DETECTION RESILIENCE USING LAYERS

ATTACK STAGES

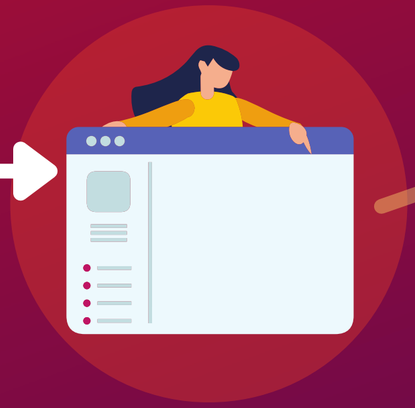


DETECTION RESILIENCE USING LAYERS

DETECTION OPPORTUNITIES

ATTACKER

PUBLIC FACING RESOURCE



DETECT

- *Vulnerability Scanning*
- *XSS Attacks*
- *SSRF Attacks*
- *SQL Injections*
- *LFI Attacks*

GOLDEN BUCKET

BACK-END SERVER

PERMISSIONS & POLICIES

Source: https://docs.datadoghq.com/security_platform/default_rules/#cat-application-security

DETECTION RESILIENCE USING LAYERS

DETECTION OPPORTUNITIES

DETECT

- *Unusual Shell Activity*
- *SSH Key Tampering*
- *Modification of critical binaries*
- *Shell history tampering*
- *Cryptomining - DNS Lookups, Suspicious Process Arguments*

BACK END SERVER

GOLDEN BUCKET

PERMISSIONS & POLICIES

Source: https://docs.datadoghq.com/security_platform/default_rules/#cat-workload-security

DETECTION RESILIENCE USING LAYERS

DETECTION OPPORTUNITIES

DETECT

- *Policy Assigments Tampering*
- *Priviliged Policy Activity*
- *IAM Policy Creation, Modification & Deletion*
- *Resource Policy Creation, Modification & Deletion*
- *User Account Tampering or Creation*

Source: https://docs.datadoghq.com/security_platform/default_rules/#cat-cloud-siem

DETECTION RESILIENCE USING LAYERS

DETECTION OPPORTUNITIES

DETECT

- *Bucket Enumeration*
- *Disabling public access*
- *Bucket policy modification*

Source: https://docs.datadoghq.com/security_platform/default_rules/#cat-cloud-siem

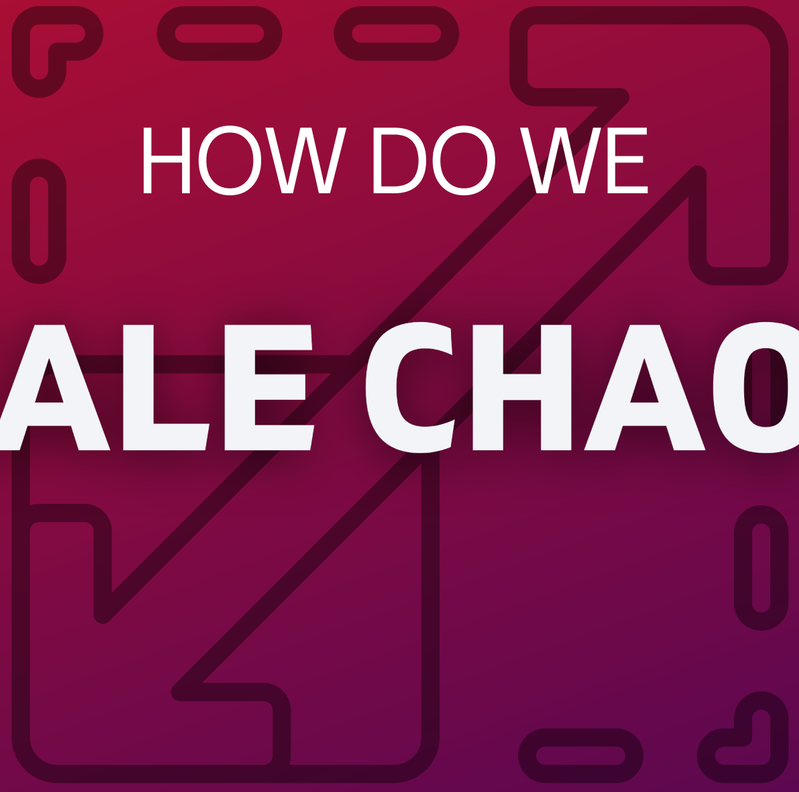
DETECTION-IN-DEPTH

PRECISION, BLIND SPOTS & EVASION

4

Measure Impact
What detections failed and why?

- [ ] **Did the detection identify the malicious activity?**
- [ ] **Consider more specificity or casting a wider net**
- [ ] **How would an attacker possibly evade this detection?**
- [ ] **Consider alternative detection methods**



HOW DO WE

SCALE CHAOS?

DETECTION ENGINEERING LIFE CYCLE

- Spin off the *SDLC* but for Detection Engineering
- Well-defined and iterative process to build and maintiain detections
- Basis of scaling chaos for detection engineering

IDEATION

- **Initial detection idea -** what are we trying to detect?
- **Threat models -** partner with other security or product teams
- **Sources -** Threat Intel, Previous Incidents, Pentests/Red Team Findings, Compliance Findings

RESEARCH

- **Breaking down the attack -** how does it work?
- **What are the attack indicators -** How do you detect it? (logs & telemetry)
- **Sources -** Threat Intel Report, IR Report, Pentest/Red Team Findings, Compliance Findings

REQUIREMENTS

- **What do you need to build this detection?**
- **Logs, Data Sources & Event parameters**
- **Log pipelines -** log parsing, log indexes

DEVELOPMENT

- **Detection Mechanism -** Threshold, Anomaly
- **Detection-as-code -** Python, JSON, YAML (Sigma, Splunk & Sentinel)
- **Code -** Query, Logic, Alerting, Tagging

DETECTION-AS-CODE IN PYTHON

```python
from panther_base_helpers import deep_get, okta_alert_context


def rule(event):
    return (deep_get(event, 'outcome', 'result') == 'FAILURE' and
            event['eventType'] == 'user.session.start')


def title(event):
    return 'Suspected brute force Okta logins to account {} due to [{}]'.format(
        deep_get(event, 'actor', 'alternateId'),
        deep_get(event, 'outcome', 'reason'))


def alert_context(event):
    return okta_alert_context(event)
```

Copy

Modules

Event Parameters

Alert

Okta Brute Force Login Rule in Panther

Source: https://panther.com/cyber-explained/detections-as-code/

TESTING & DEPLOYMENT

- **Automated Tests -** Datadog Stratus Red Team, Red Canary Atomic Red Team
- **Writing your own scripts**
- **Unit Testing** (**CI**/**CD**) - Query syntax, code format, tagging, event parameters
- **Validation -** Datadog's Threatest, Manual validation
- **Peer Review -** IR, Internal Security, Product Security, e.t.c

MAINTENANCE

- **Improvements -** updates, docs, enrichment, alerting, suppressions, reports
- **Response & Triage Elements** - Automation use cases
- **How do you retire this detection if necessary?**
- **Version Control -** updates, bug fixes, deprecation, e.t.c

DEMOCRATIZING DETECTION
DECENTRALIZATION

PROBLEM: Individual teams lack detection expertise

- Reduce isolation and exclusivity of detection engineering
- Partner with other teams to deliver detection capabilites across the org
- Prioritize a strong feedback loop across the org
- Implement a detection championship program

DEMOCRATIZING DETECTION
AUTOMATION

PROBLEM: Burnout and Alert Fatigue

1

Account Takeover
The attacker authenticates to a sensitive resource.

2

Detection Strategy
Monitor audit logs for access to sensitive resource

Response
Slack bot delivers a message to the user with context.

4

Escalation
User responds. Alert gets closed or severity is adjusted.

Context
- Timestamp
- Sensitive resource
- IP Address
- Location

Applying automation use cases to reduce investigation overhead

Source: https://blog.palantir.com/democratizing-security-detection-71c689b667a5

EMBRACE FAILURE, DISGRACE BLAME

- Acknowledge detection gaps and learn how to slowly close them
- Embrace the imperfections in your detections
- Learn from failures in order to improve your detections
- Humans are at the center of everything

START SMALL

- Test simpler scenarios with little impact
- Experiment in low risk environments
- Validate current configurations
- Scale

A CHAOTIC CAUSE FOR DETECTION SUCCESS

THE EFFECTS OF CHAOS CAN BE HARNESSED FOR GOOD

- **Proactively test your detections before real attacks happen**
- **Identify gaps and failures in detection capabilities**
- **Build confidence in detection capabilities**
- **Measurable success of detection organization**

What could possibly go wrong?

Source: https://en.wikipedia.org/wiki/Disaster_Girl

SOURCES & REFERENCES

Chaos Engineering

https://principlesofchaos.org/

https://www.oreilly.com/library/view/chaos-engineering/9781492043850/

Security Chaos Engineering

https://www.verica.io/blog/announcing-the-security-chaos-engineering-report/

https://www.oreilly.com/library/view/security-chaos-engineering/9781492080350/

https://opensource.com/article/18/1/new-paradigm-cybersecurity

https://www.verica.io/blog/security-chaos-engineering-how-to-security-differently/

https://youtu.be/QcTQEn3uzKg

https://youtu.be/jgZAfgDWkOw

https://youtu.be/OTOIOXjL66O

Detection Engineering Lifecycle

https://medium.com/snowflake/detection-development-lifecycle-af166fffb3bc

https://www.securonix.com/blog/managing-security-threats-with-detection-development-life-cycle/

Detection-in-depth

https://posts.specterops.io/detection-in-depth-a2392b3a7e94

https://inquest.net/blog/2020/08/28/Detection-in-Depth

https://medium.com/anton-on-security/detection-coverage-and-detection-in-depth-16137e6c203b

Democratizing Detection

https://blog.palantir.com/democratizing-security-detection-71c689b667a5

Detection Validation

https://securitylabs.datadoghq.com/articles/threatest-end-to-end-testing-threat-detection/

Detection Engineering

https://panther.com/cyber-explained/detection-engineering-benefits/

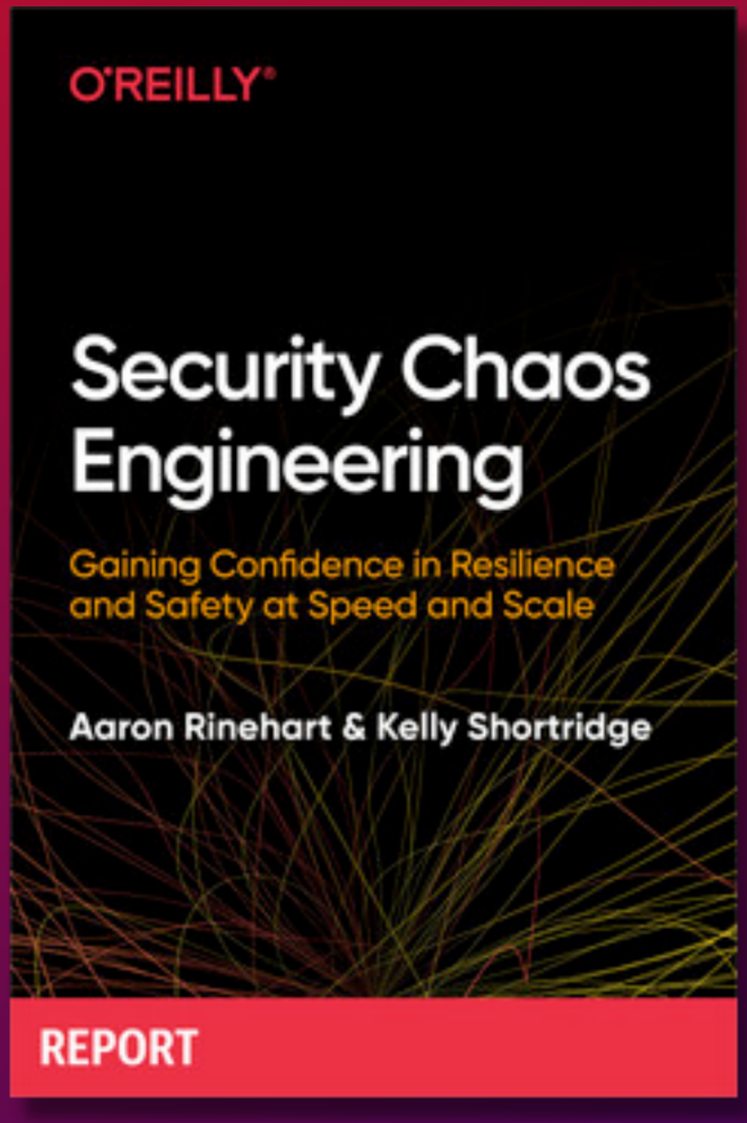
https://github.com/daycyberwox/Azure-Detection-Engineering

https://redcanary.com/blog/false-negatives/

https://www.intezer.com/blog/threat-hunting/scale-incident-response-detection-engineering/

https://redcanary.com/blog/detection-engineering/

SOURCES & REFERENCES



Source: https://www.oreilly.com/library/view/security-chaos-engineering/9781492080350/

THANK YOU