



**Contact Information**

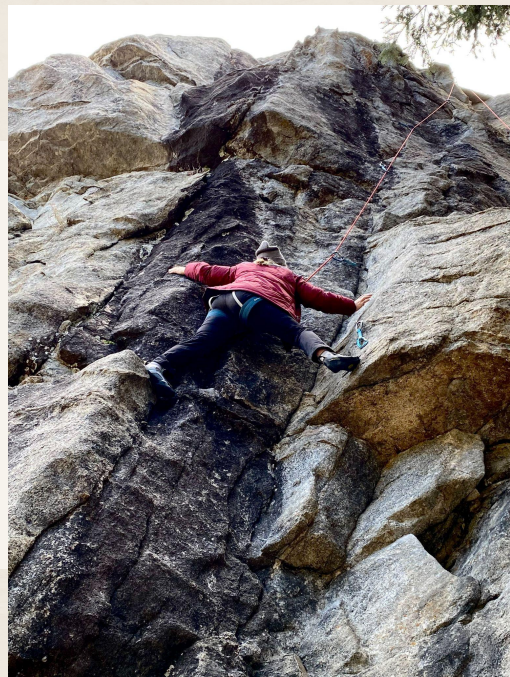
@julieasparks  
juliesparks.io

# Crash Course in... Detection Engineering

Julie Agnes Sparks

# About Me

- Currently a Detection & Response Security Engineer at Brex
- Previously worked @ Cloudflare & Palo Alto Networks
- Located in Denver, CO ☀️
- Obsessed with rock climbing, learning new things, and trying new recipes.



# Detection & Response

## What do we do?

- Investigate and remediate security incidents.
- Build internal tooling to ingest logs.
- Build automation workflows to assist with security incidents.
- Threat modeling/threat research.
- Write security detections/alerts for suspicious behavior.

## Similar (Blue) Teams:

- Security Operations (SOC)
- Threat Research
- Security Detection Engineering
- Security Tooling
- Computer/Security Incident Response

# Security Detections 101



# Definitions & Concepts

## Logs

Logs are records of events that happen on a computer system or application, either by a person or by a running process.

## SIEM

is a software solution that aggregates and analyzes logs from many different resources across your entire company infrastructure.

## GCP

Stands for Google Cloud Platform.

## Detection/Rule

Logic or function that match on a specific pattern.

## SOAR

Stands for Security Orchestration, Automation, and Response. These tools aid in automating incident response.

## Incident Response

Is an organized approach to addressing and managing the aftermath of a security breach or cyberattack.

## Threat Research

Researching and building a story around an attacker and attack that could occur in your environment.

## Alert

Notification triggered from a detection/rule that the pattern occurred.



# What is the purpose of writing security detections?

Catch the bad things but not the good things!

(A.k.a low false positive rates)



Detections are  
one part of the  
greater “Incident  
Response &  
Recovery” Cycle





# Security Detection Lifecycle

1. Determine what you have to work with.
2. What threat do you want to detect? Reference MITRE ATT&CK
3. Research the threat in your environment and externally.
4. Build the logic & test your hypothesis.
5. Evaluate next steps (documentation, peer review, etc.)





# What spurs detection writing?

## Threat Hunting

Example: A threat hunting or modeling exercise into a system generates findings of potential “bad events.”

## Security Incident

Example: The company finds a user in AWS has been compromised. This investigation is used as research input to detect future occurrences.

## New Asset or Regulation

Example: IT purchases a new data lake platform.

Example: US passes new sanctions and Legal requests security to detect on violations.



# What could you write a security detection on?

- Internal Network Traffic Logs
- Administrator Logs to the Corporate Outlook Account
- Access to SaaS Resources 
- Process on Corporate Endpoints

TL;DR Anything that produces logs and you could imagine a malicious scenario related to that system.

# MITRE SaaS Matrix

## SaaS Matrix

Below are the tactics and techniques representing the MITRE ATT&CK® Matrix for Enterprise covering cloud-based techniques. The Matrix contains information for the SaaS platform.

[View on the ATT&CK® Navigator](#)



[Version Permalink](#)

layouts ▾

show sub-techniques

hide sub-techniques

help

| Initial Access       | Persistence        | Privilege Escalation | Defense Evasion                           | Credential Access              | Discovery                       | Lateral Movement                          | Collection                             | Impact                         |
|----------------------|--------------------|----------------------|---|--------------------------------|---------------------------------|---|--|--------------------------------|
| 4 techniques         | 1 techniques       | 1 techniques         | 2 techniques                              | 5 techniques                   | 4 techniques                    | 2 techniques                              | 1 techniques                           | 2 techniques                   |
| Drive-by Compromise  | Valid Accounts (2) | Valid Accounts (2)   | Use Alternate Authentication Material (2) | Brute Force (3)                | Account Discovery (1)           | Internal Spearphishing                    | Data from Information Repositories (1) | Endpoint Denial of Service (3) |
| Phishing (1)         |                    |                      | Valid Accounts (2)                        | Forge Web Credentials (2)      | Cloud Service Discovery         | Use Alternate Authentication Material (2) |  | Network Denial of Service (2)  |
| Trusted Relationship |                    |                      |   | Steal Application Access Token | Permission Groups Discovery (1) |   |  |                                |
| Valid Accounts (2)   |                    |                      |   | Steal Web Session Cookie       | Software Discovery (1)          |   |  |                                |
|                      |                    |                      |   | Unsecured Credentials          |                                 |   |  |                                |



# (SaaS) Detection Rule Examples

## Single Event/Simple

For example, if a user logs in from a known malicious IP address.

## Multiple Event/Composite

For example, if there are multiple Google Drive API calls by a non-internal application, then trigger an alert.

## Machine Learning Model

A user logs in remotely at 3 a.m. (usually only doing so locally during normal business hours), then makes repeated attempts to connect to a cloud database.








# Let's Build







**Threat Scenario:** A social engineering attack has started at your company. The attacker is calling employees asking them to login to a fake Okta page that captures their credentials.



# Okta Authentication Log


Based on these authentication logs, how can you tell if an account was compromised through the social engineering attack?

```
{
  "actor":{
    "id":"00u17b6c3rwVP7kqo1d8",
    "type":"User",
    "alternateId":"kyle.diedrich@company.com",
    "displayName":"Kyle Diedrich",
    "detailEntry":null
  },
  "client":{
    "userAgent":{
      "rawUserAgent":"PostmanRuntime/3.0.11-hotfix.2",
      "os":"Unknown",
      "browser":"UNKNOWN"
    },
    "zone":"null",
    "device":"Unknown",
    "id":null,
    "ipAddress":"12.97.85.90",
    "geographicalContext":{
      "city":"San Francisco",
      "state":null,
      "country":"United States",
      "postalCode":"94107",
      "geolocation":{
        "lat":37.7697,
        "lon":-122.3933
      }
    }
  },
  "authenticationContext":{
    "authenticationProvider":null,
    "credentialProvider":null,
    "credentialType":null,
    "issuer":null,
    "interface":null,
    "authenticationStep":0,
    "externalSessionId":"trsp5PU70IoTgC0dFBgJ0QWIA"
```

# Writing Detection Logic

What behaviors in the Okta logs would help us detect a compromised account?

- Sign in from a new device (user agent)
- Use of IP address that is not an ISP or has known malicious behavior.
- Events from an abnormal time for their region



```
event.outcome.result = "SUCCESS" &&  
event.request.geographicalContext.country  
!= "United States" &&  
has_okta_group(event.alternateId, "US  
Employee");
```

```
},  
"actor": {  
  "type": "User",  
  "alternateId": "jsparks@cloudflare.com",  
  "displayName": "Julie Agnes Sparks",  
  "id": "00uo80ivdBKOW5ogo356"  
},  
"authenticationContext": {  
  "externalSessionId": "unknown"  
},  
"severity": "INFO",  
"displayMessage": "User single sign on to app",  
"version": "0",  
"outcome": {  
  "result": "SUCCESS"  
},  
"published": "2021-06-15T15:13:22.028Z",  
"request": {  
  "ipChain": [  
    {  
      "ip": "2a06:98c0:3600::103",  
      "version": "v6"  
    },  
    {  
      "ip": "172.68.34.119",  
      "geographicalContext": {  
        "state": "Colorado",  
        "geolocation": {  
          "lat": 39.7388,  
          "lon": -104.9868  
        },  
        "country": "United States"  
      },  
      "version": "v4"  
    }  
  ]  
}
```

# Testing Your Detection

Run your newly written detection in a testing environment.  
Does it detect the behavior you wanted it to?  
Are there any false positives?  
What can be improved?

Example: Your IT department has several old user accounts that act as service accounts which create false positives.





# Evaluate the Detection

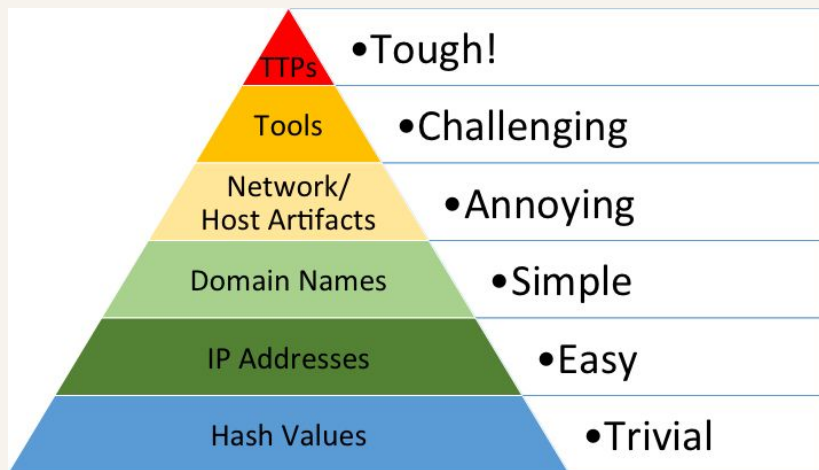
If you believe the detection should be in production, document all your research and testing!

Then...

1. Pass it on for peer feedback from your teammates.
2. Write a playbook with the threat scenario and specific actions that can be taken to investigate this alert.
3. Evaluate if response can be automated.

# Increasing Complexity

As you cover some of the simpler attack scenarios you begin to increase the complexity of your detections.



<http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html>

## Just starting?

- [A SOCless Detection Team at Netflix](#)
- [Lessons Learned in Detection Engineering](#)
- [Distributed Security Alerting at Slack](#)
- [How to Write Security Alerts](#)

## Diving in deeper?

- [Lessons Learned in Detection Engineering](#)
- [Can we have Detection as Code?](#)
- [O'Reilly's Machine Learning & Security Book](#)