AWS re:Invent

NFX205

Monitoring anomalous application behavior

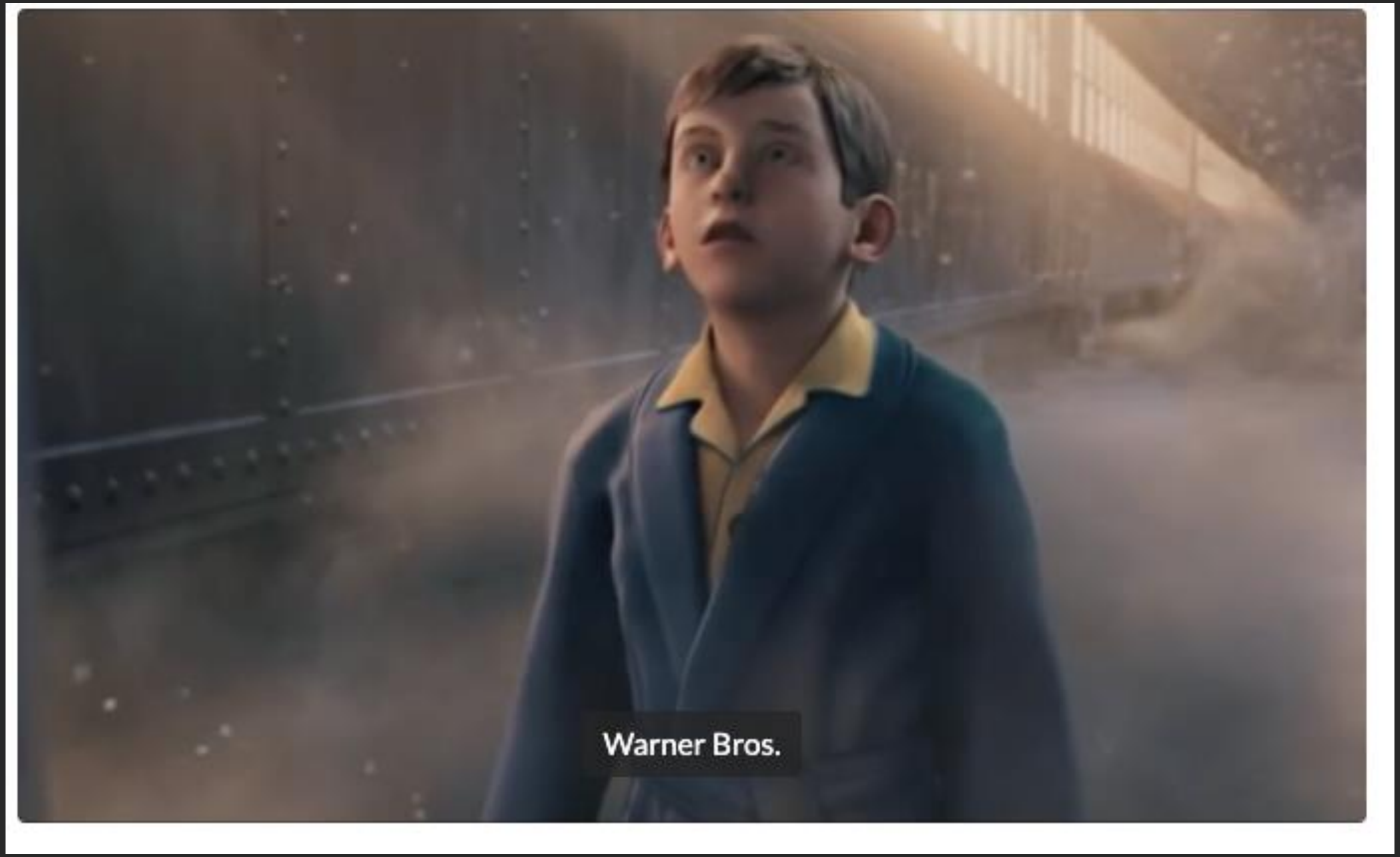
Travis McPeak
Security Engineering Manager
Netflix

Will Bengtson
Director
Threat Detection and Response
HashiCorp

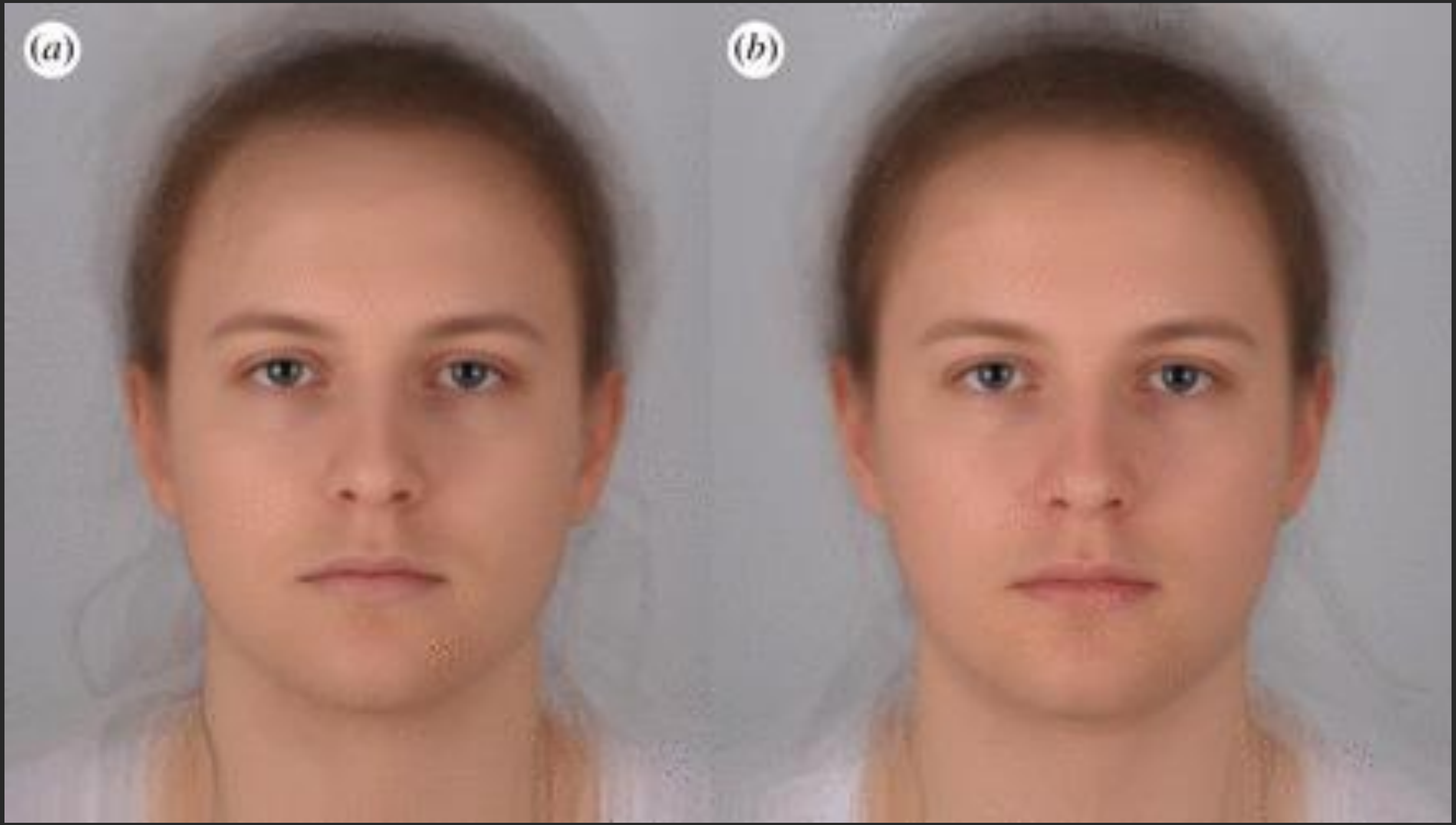
AWS re:Invent

aws

Anomalous “human” characters



Humans: Great at detecting deviation from baseline



Source: Axelsson, John; Sundelin, Tina; J. Olsson, Mats; Sorjonen, Kimmo; Axelsson, Charlotte; Lasselin, Julie; et al. (2017): Supplementary material from "Identification of acutely sick people and facial cues of sickness". The Royal Society. Collection. https://doi.org/10.6084/m9.figshare.c.3951916.v2

By understanding our principals' baselines, we can *quickly* spot unusual behavior in our environment

You can get started doing this *today* with simple (open-source) automation

Monitoring anomalous application behavior

1. Why is this important?
2. What do we need to build it?
3. How does it work?
4. Less talk, more code!

Breaches

29.6% chance breach in next 2 years

206 days to spot a breach

Breaches discovered quickly, **37%** cheaper

Source: 2019 Ponemon Institute "Cost of a Data Breach" Report

Detect early, detect often

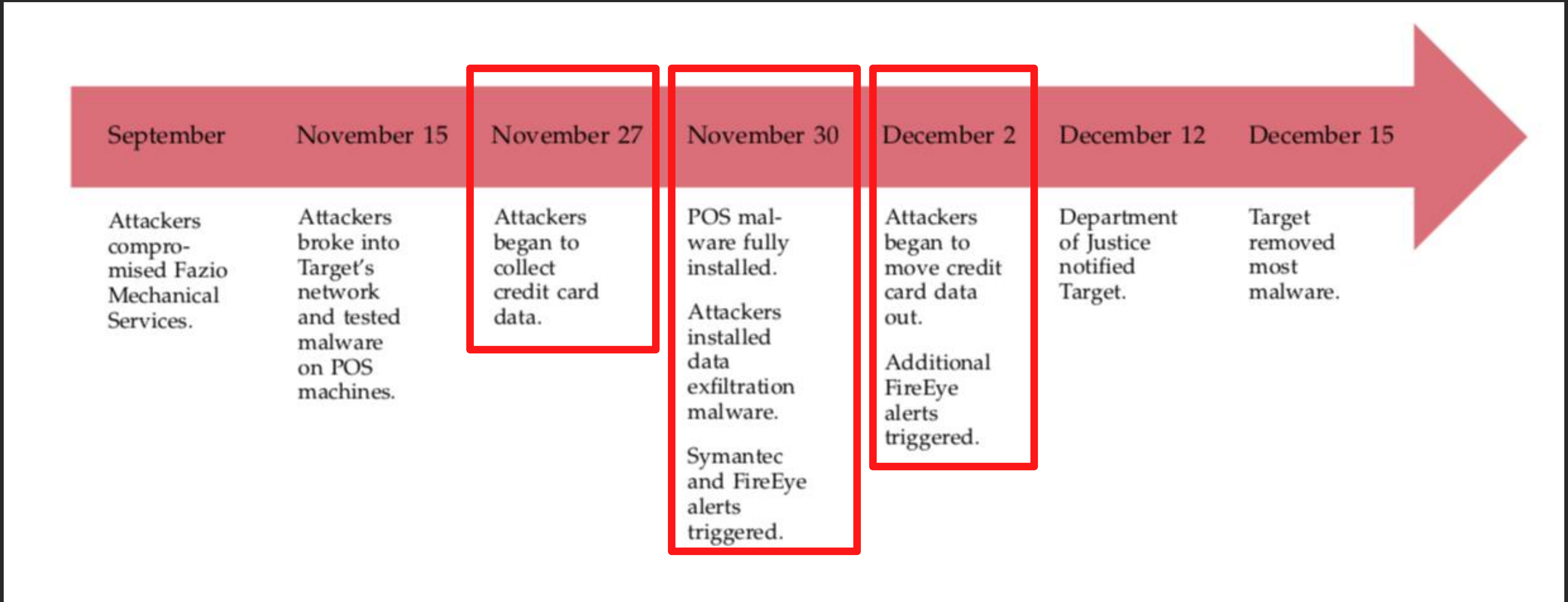

September
November 15
November 27
November 30
December 2
December 12
December 15
Attackers compromised Fazio Mechanical Services.
Attackers broke into Target's network and tested malware on POS machines.
Attackers began to collect credit card data.
POS malware fully installed.
Attackers installed data exfiltration malware.
Symantec and FireEye alerts triggered.
Attackers began to move credit card data out.
Additional FireEye alerts triggered.
Department of Justice notified Target.
Target removed most malware.

Fig. 1. Timeline of the Target data breach (2013).

Source: "Breaking the Target: An Analysis of Target Data Breach and Lessons Learned" by Xiaokui Shu, Ke Tian, Andrew Ciambrone, and Danfeng (Daphne) Yao

Image: Wikimedia Commons

Building blocks

- ✓ AWS CloudTrail
- ✓ CloudTrail somewhere query-able
- ✓ One role per application/user
- ✓ Established "burn-in" period
- ✓ Roles named consistently across accounts

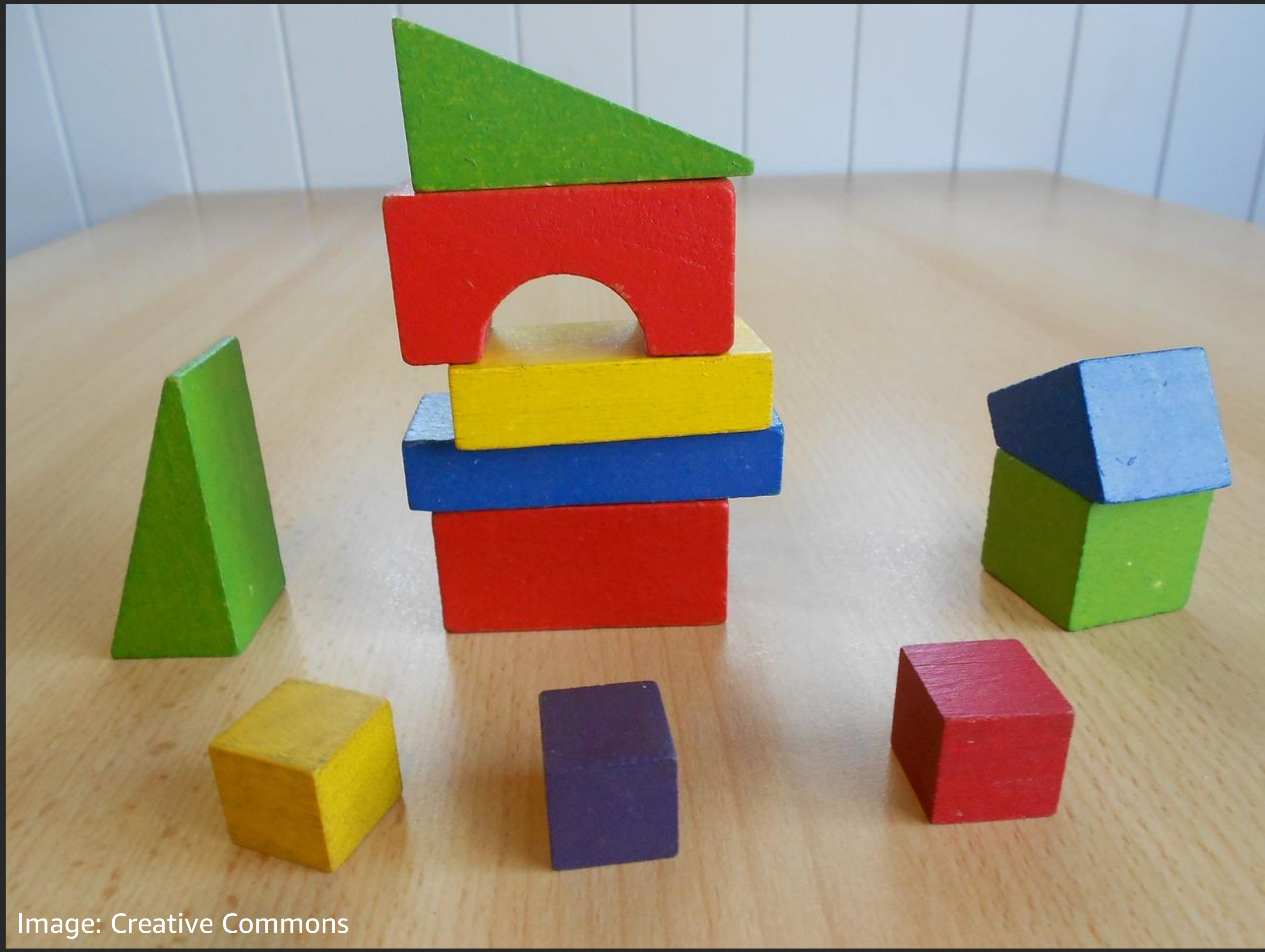


Image: Creative Commons

Doing this requires more work: make it fun/useful

When new behavior shows up:

- 🙇‍♀️ Try to predict the most likely explanation
- 👋 Reach out to the developer/user/owner, say hi, ask them (calmly)
- 🧑‍🤝‍🧑 Great way to meet people, build relationships, learn new stuff

Worst case: You learn something and meet somebody new

Best case: Detect attacker behavior <u>early!</u>

How about Amazon GuardDuty?

GuardDuty does clever stuff, including:

- **API invoked from a Tor exit node**
- **API invoked from known malicious IP/list**
- **Unusual/"recon" call(s) made by a principal**

TL;DR: GuardDuty is cool; you should check it out

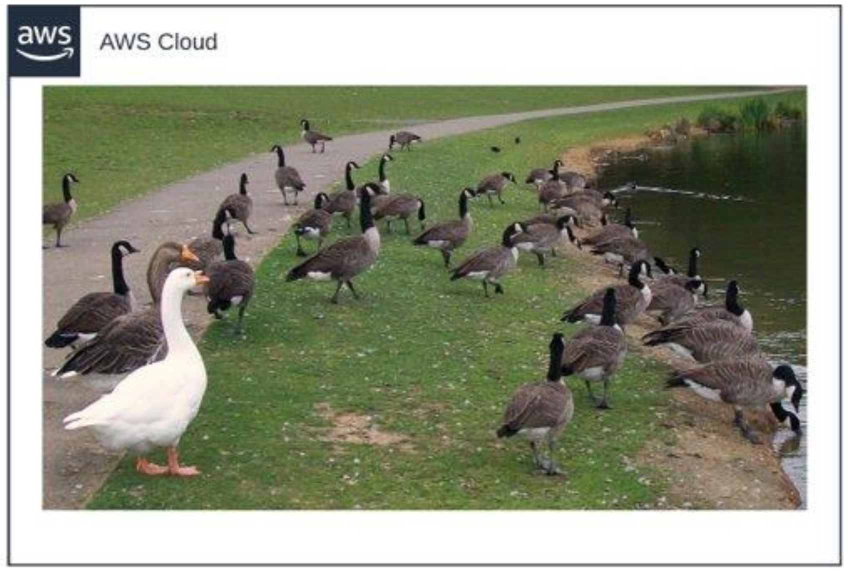
The method we are describing can be used in addition to GuardDuty

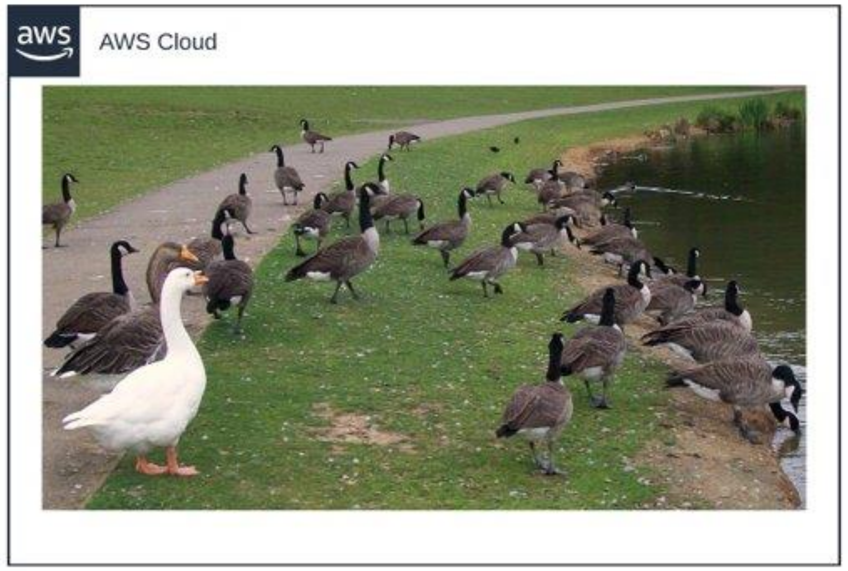
Deep dive

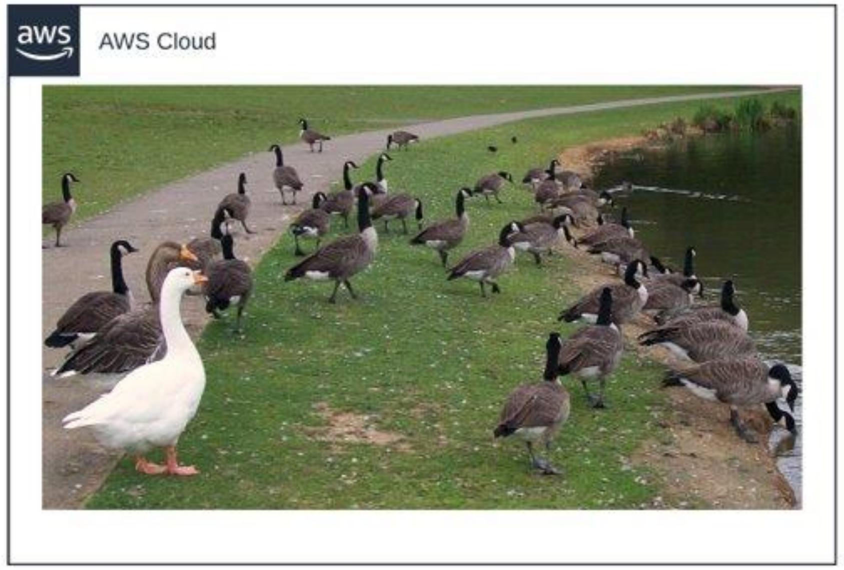
```
{"Records": [{
    "eventVersion": "1.0",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "accountId": "123456789012",
        "userName": "Alice"
    },
    "eventTime": "2014-03-06T21:22:54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "205.251.233.176",
    "userAgent": "ec2-api-tools 1.6.12.2",
    "requestParameters": {"instancesSet": {"items": [{"instanceId": "i-ebeaf9e2"}]}},
    "responseElements": {"instancesSet": {"items": [{
        "instanceId": "i-ebeaf9e2",
        "currentState": {
            "code": 0,
            "name": "pending"
        },
        "previousState": {
            "code": 80,
            "name": "stopped"
        }
    }]}}
}]}
```

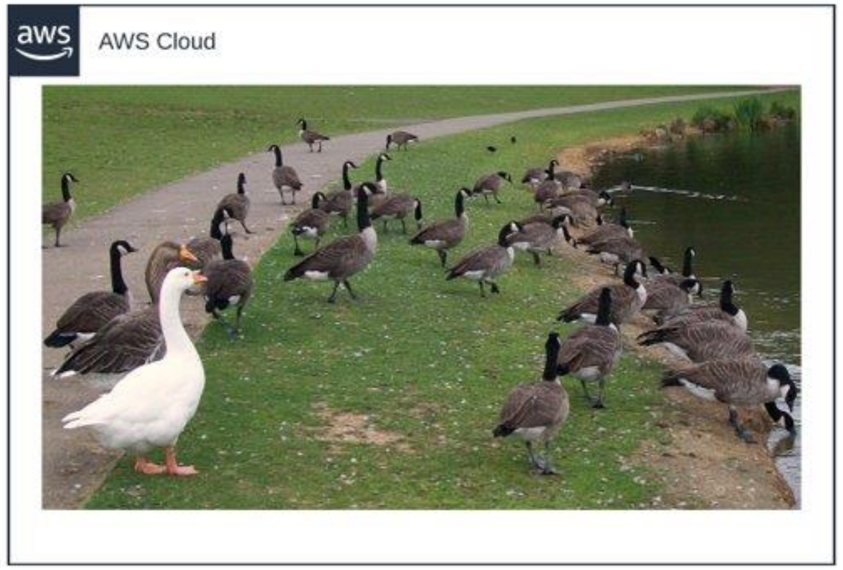
Image: Wikimedia Commons

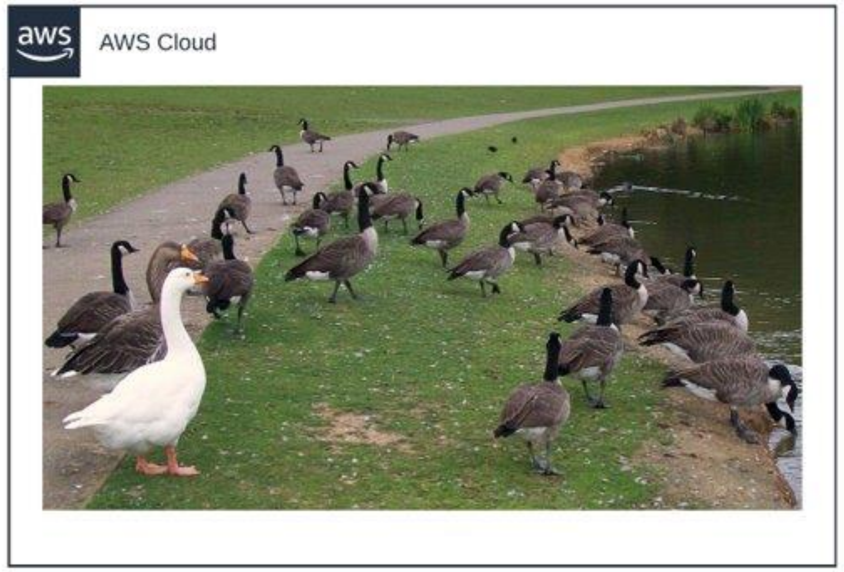
AWS Cloud

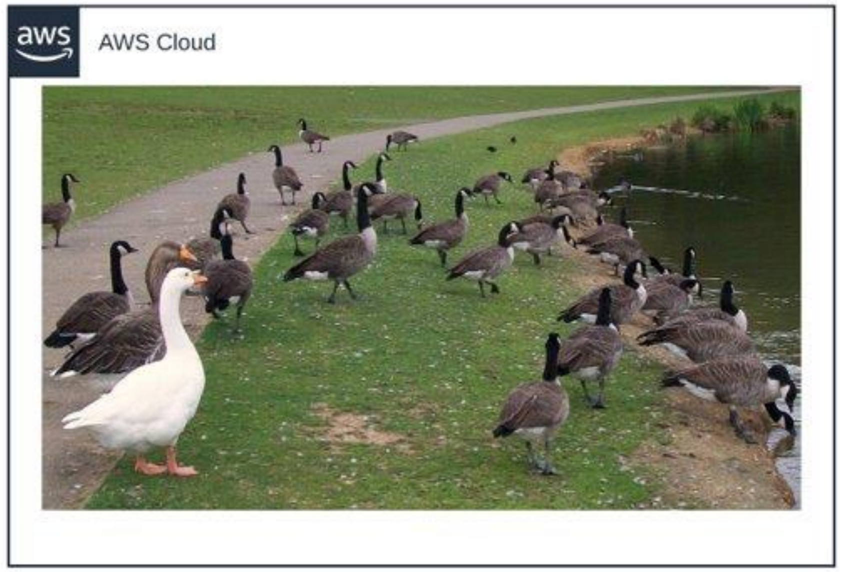

AWS Cloud


AWS Cloud


AWS Cloud


AWS Cloud


AWS Cloud


AWS Cloud

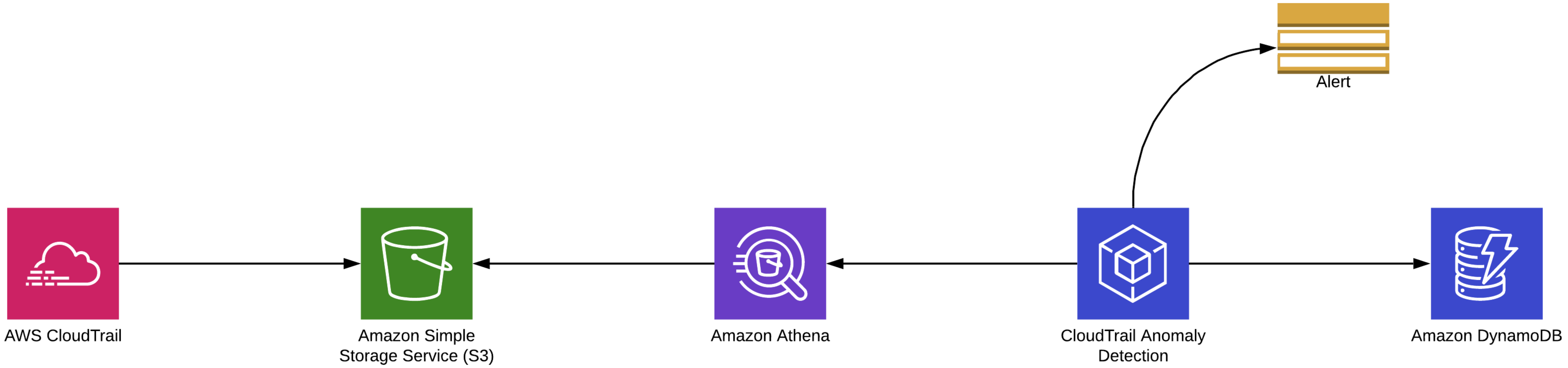
Anomaly detection iteration

- ✓ First time across all accounts
- ✓ First time in a single account
- ✓ First time for an IAM role
- ✓ First time for IAM role in N days
- ✓ First time for IAM role in Region

IAM = AWS Identity and Access Management

Considerations

- Accurate list of AWS accounts — use organizations
- One role per service principal
- One role per app per region
- Consistent role naming
  - streamingApplicationInstanceProfile
  - monitoringLambdaProfile
  - historicalMinionRole
- Amazon Simple Storage Service (Amazon S3) data events CloudTrail
  - Anomalous bucket access
- Check out CloudTrail Insights


Alert
AWS CloudTrail
Amazon Simple Storage Service (S3)
Amazon Athena
CloudTrail Anomaly Detection
Amazon DynamoDB

Less talk, more code

Pseudocode

```
for account in accounts:
    iam_roles = get_roles()

    for role in iam_roles:
        unique_calls = get_unique_calls(60)

        for call in unique_calls:
            if call not in dynamo:
                put(eventname, ttl); alert();
            else:
                update(eventname, ttl)
```

Open source

https://www.github.com/netflix-skunkworks/cloudtrail-anomaly

Thank you!

Travis McPeak

Twitter: @travismcpeak
LI: https://www.linkedin.com/in/travismcpeak/

Will Bengtson

Twitter: @__muscles
LI: https://www.linkedin.com/in/william-bengtson/

Please complete the session survey in the mobile app.

AWS re:Invent

aws