Sensible Security for AWS Workloads

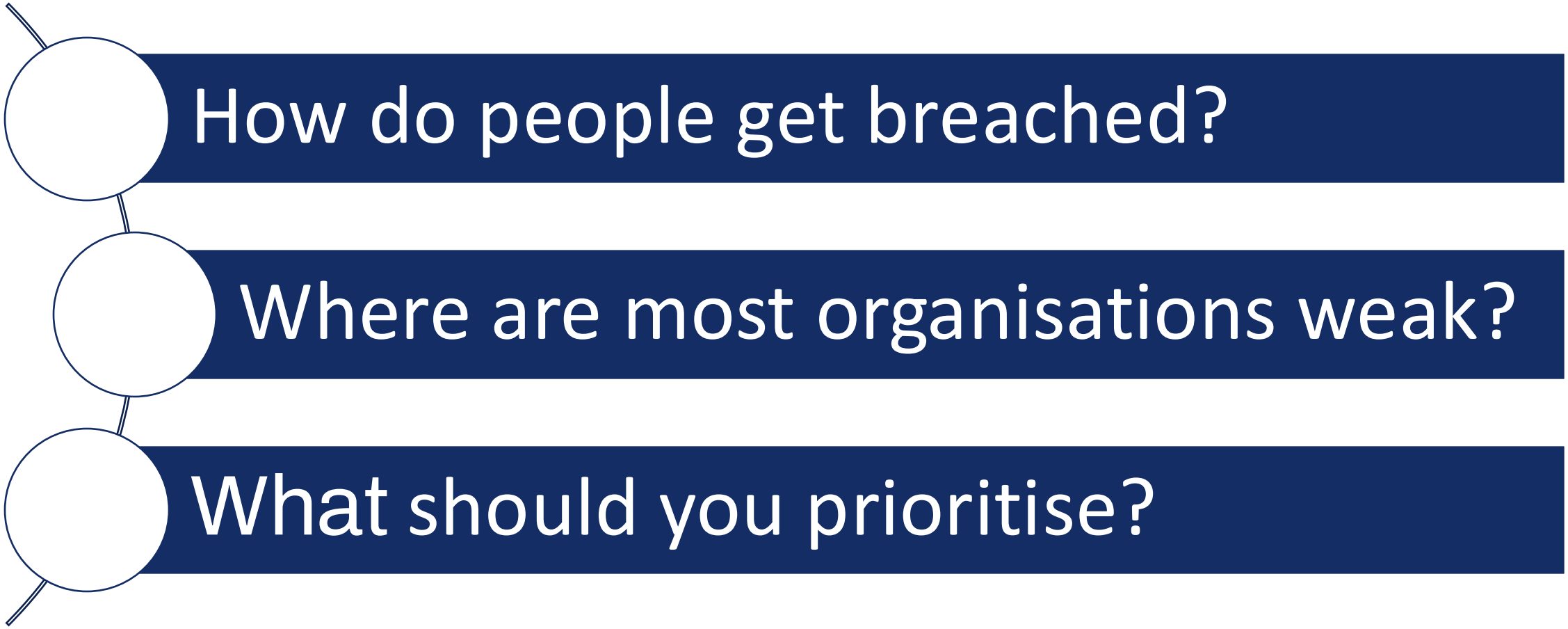
Nick Jones – AWS Community Day NL 2024

About Me

Nick Jones

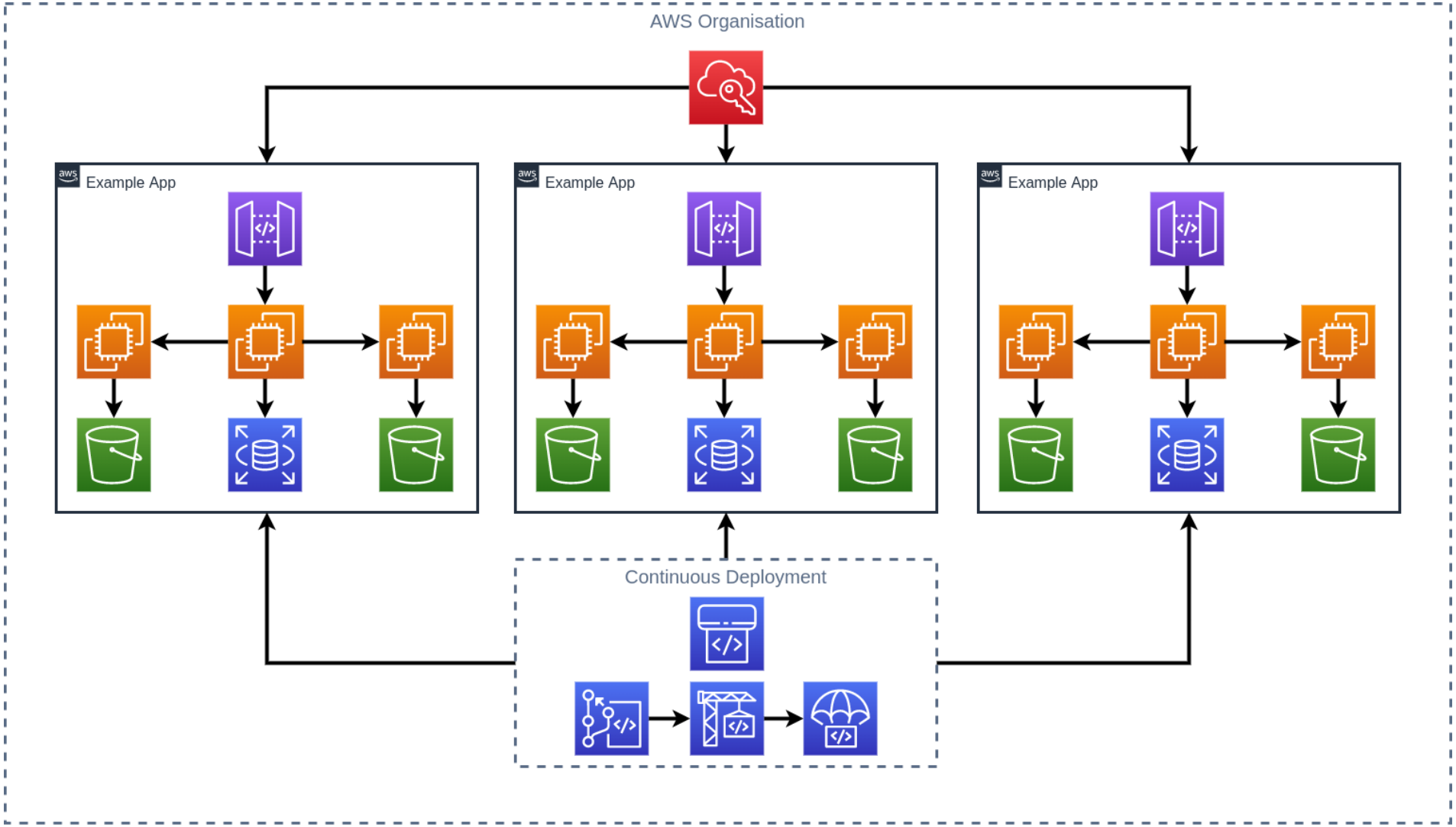
- Global Head of Research @ WithSecure Consulting
- Ex-Cloudsec Consulting Lead
- AWS Community Builder

Agenda

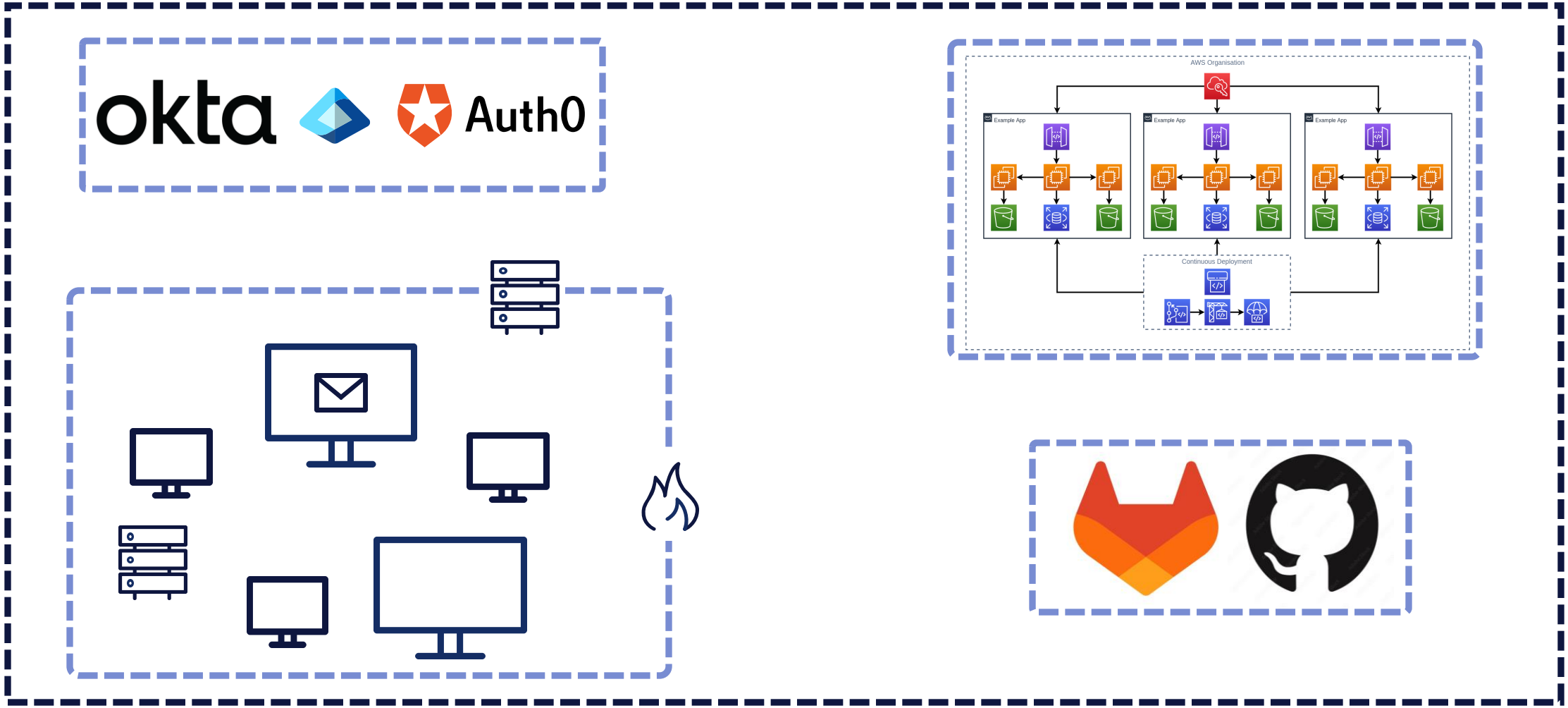


Security's Idea of the Cloud

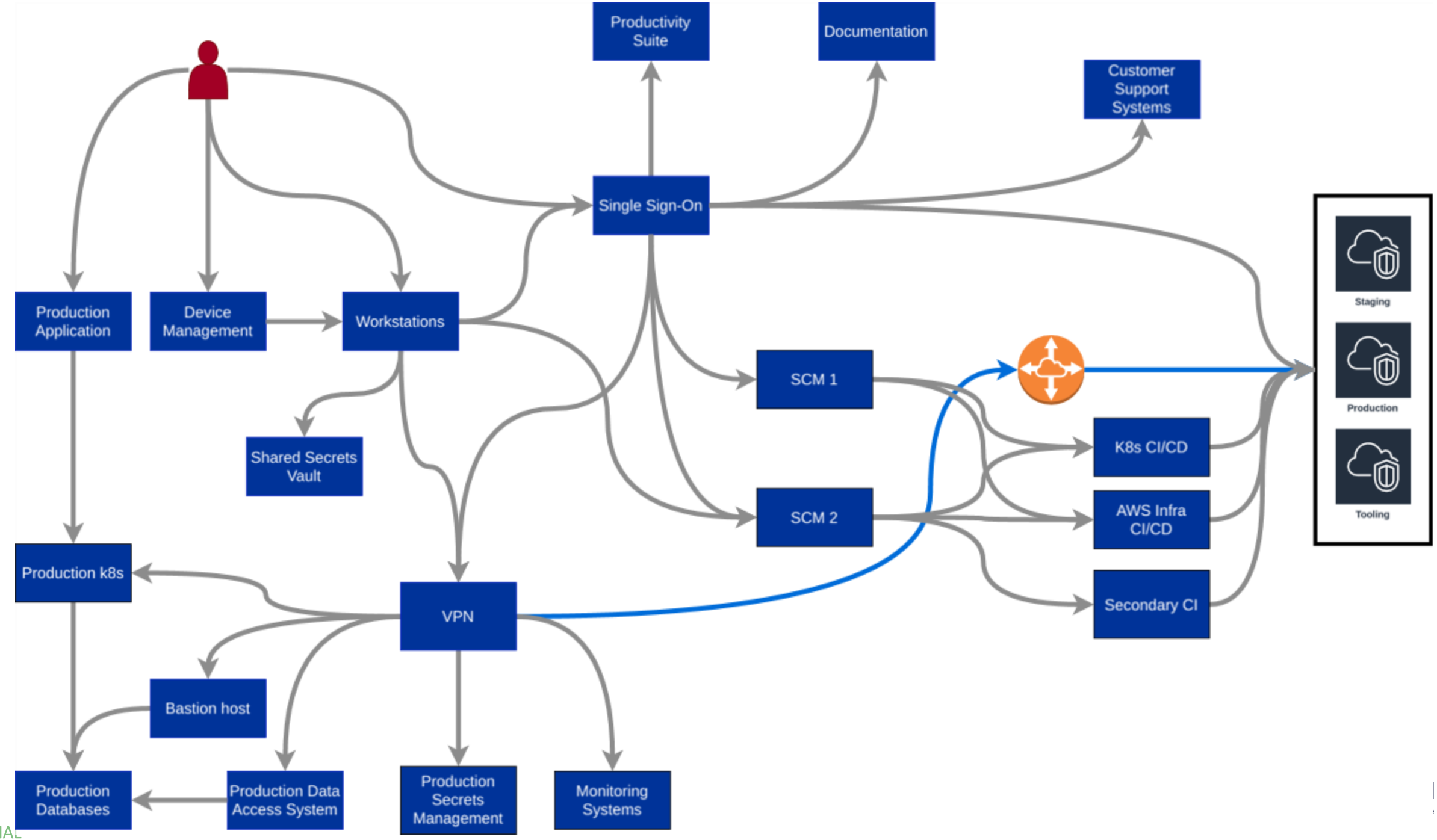
Cloud's Idea of Security



Reality


okta
Auth0
AWS Organisation
Example App
Example App
Example App
Continuous Deployment

Attack Vectors


Productivity Suite
Documentation
Customer Support Systems
Single Sign-On
Production Application
Device Management
Workstations
Shared Secrets Vault
SCM 1
SCM 2
K8s CI/CD
AWS Infra CI/CD
Secondary CI
Staging
Production
Tooling
Production k8s
VPN
Bastion host
Production Databases
Production Data Access System
Production Secrets Management
Monitoring Systems

Cloud Native Management Services

Native SSH/RDP aren't great

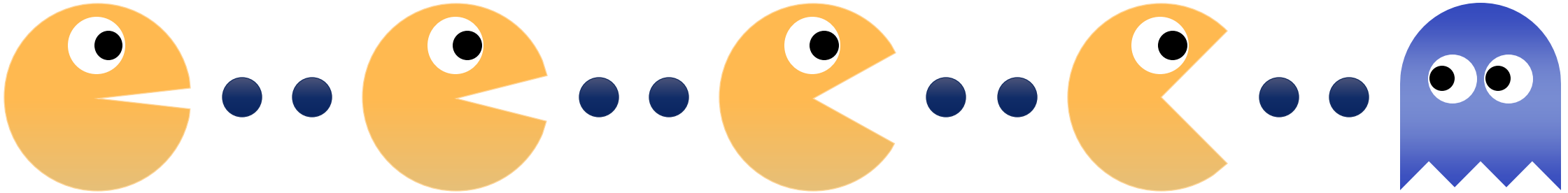
- Network level access to manage
- Overhead of separate authentication systems
- Harder to log & audit

Cloud Native Admin Tools are *mostly* better

- (Usually) easier identity management, fewer networking concerns
- Caveat: It joins two previously separate security domains
- Your IAM/permissions model needs to be solid!

Cloud-Style Shell Popping!

Objective
Root on an EC2 instance full of sensitive data



Compromise Credentials
Find IAM User keys in a public Github Repository

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

Recon
What services is the client using?

Pop Shells
Use SSM to get shell access on EC2 instances

Cloud Native Phishing

Identity Platforms / SSO

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

Interesting security properties

- Multi Factor Authentication, Conditional Access Policies etc.
- Often poor session management
- Get the session token, get access to ***everything***

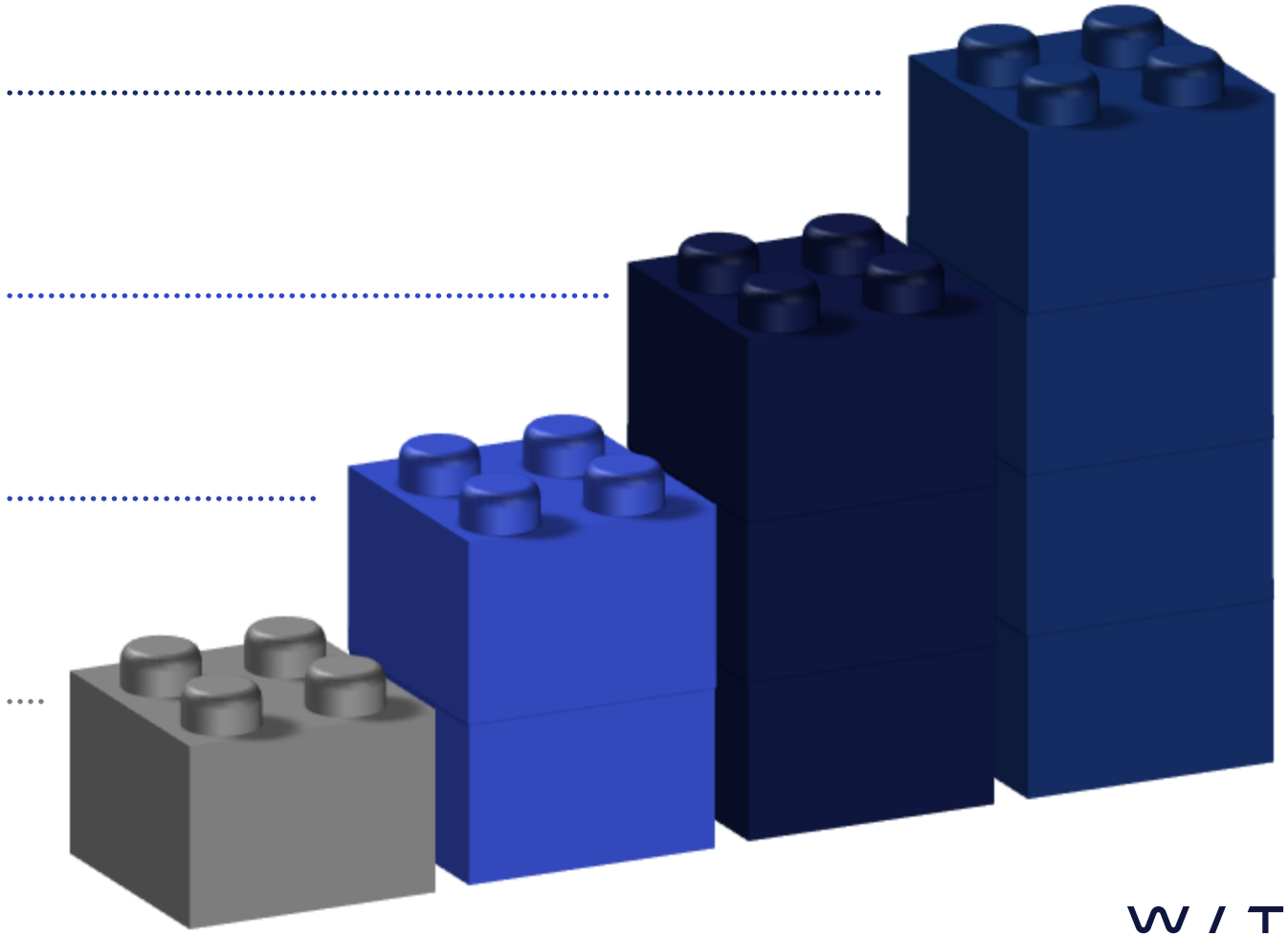
Exploiting Development Workflows

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

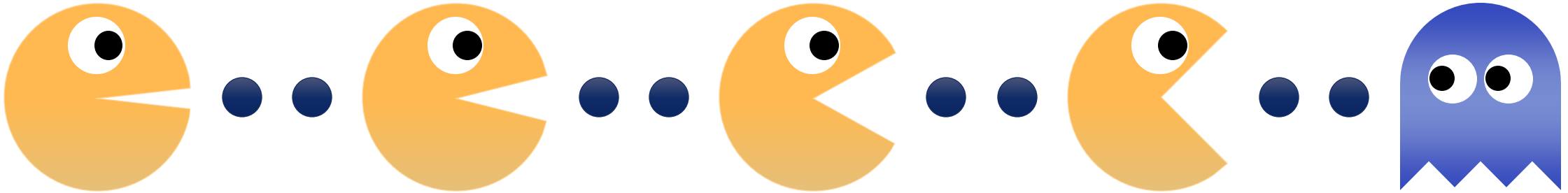
Dev Usability > Security
Enabling devs to move at speed often means system architectures and controls are not well hardened

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



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

Common Breach Scenarios

Breach Dataset

Inspired by Rami McCarthy's Breach Dataset

- Curated dataset of AWS related security incidents
- https://github.com/ramimac/aws-customer-security-incidents

Highlights

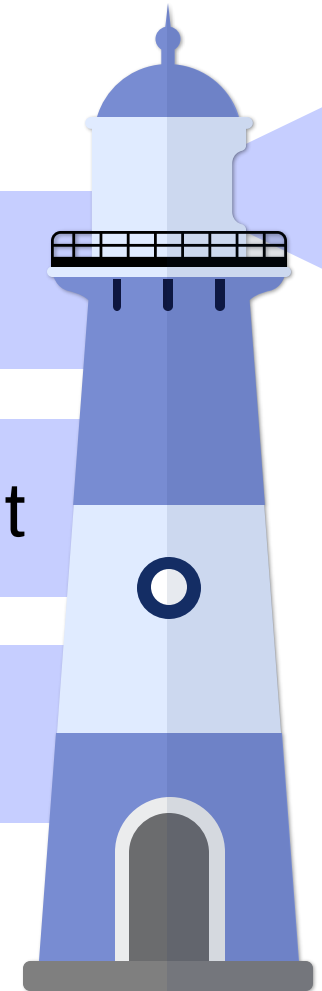
- >50 breaches back to 2014
- >30 incident reports
- Ignores S3 buckets – too many to count!

Inherently Flawed Data

Not all breaches get spotted

Providers hate talking about it

Focus on low hanging fruit



Open S3 Buckets

The perennial problem

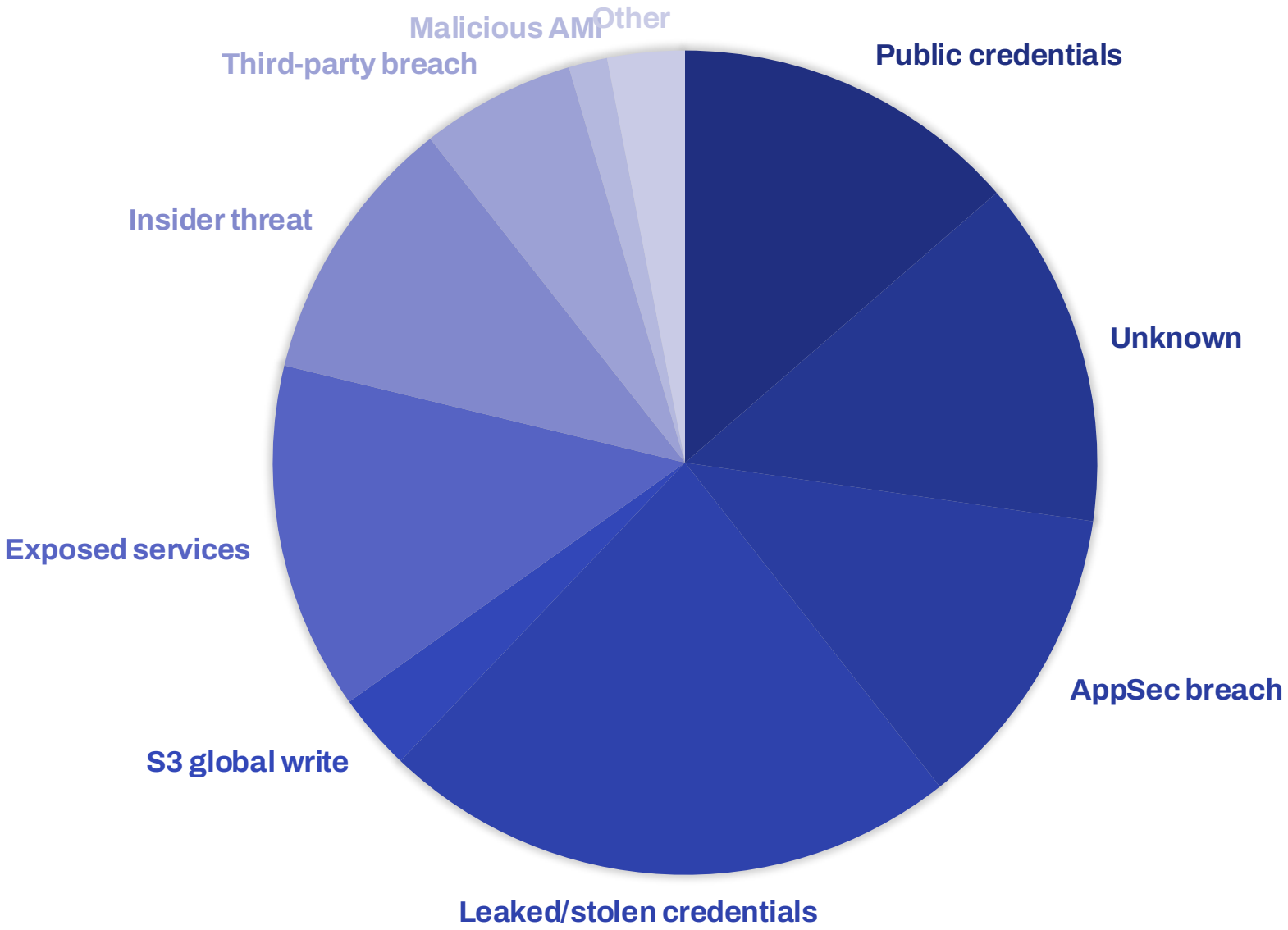
- Biggest source of breaches for years now
- Trivial to find and exploit

Situation is Improving

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

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

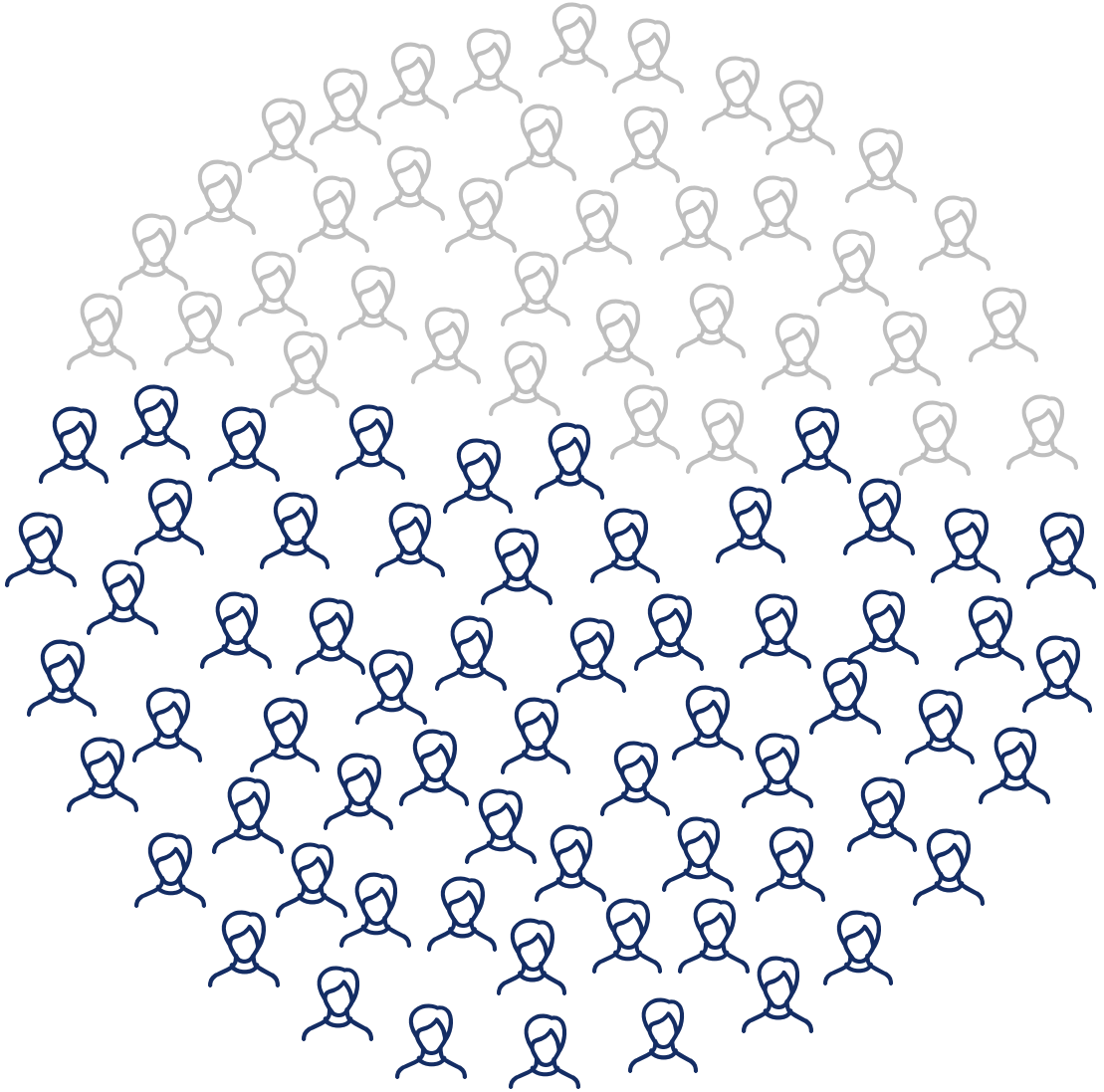
Breach Causes



WITH secure

44%*

Breaches involving IAM users



WITH secure


NUKE IT FROM ORBIT
ITS THE ONLY WAY TO BE SURE
memegenerator.net

Summary

Attackers look for the easiest path

- Most attacks are opportunistic
- Your org is likely not a priority target
- The basics helps stop APTs too

Most get breached by the basics:

- **Public Storage Accounts**
- Forgotten accounts
- Leaked credentials
- Bad leaver handling

You **probably** won't get breached by:

- Encryption at rest
- Not using *[insert shiny security feature]*
- Zero days
- CSP Insider threat

Key Security Controls

Strong Identity Controls

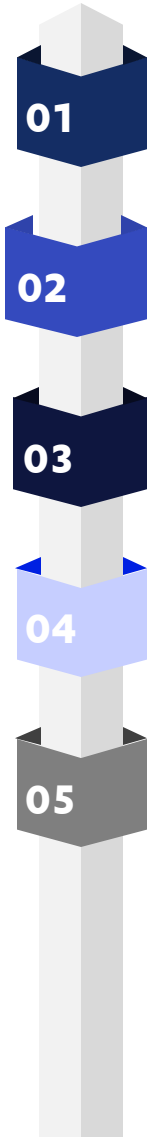
Enforce Multi-Factor Authentication (MFA) everywhere

Apply principle of not-very-much privilege

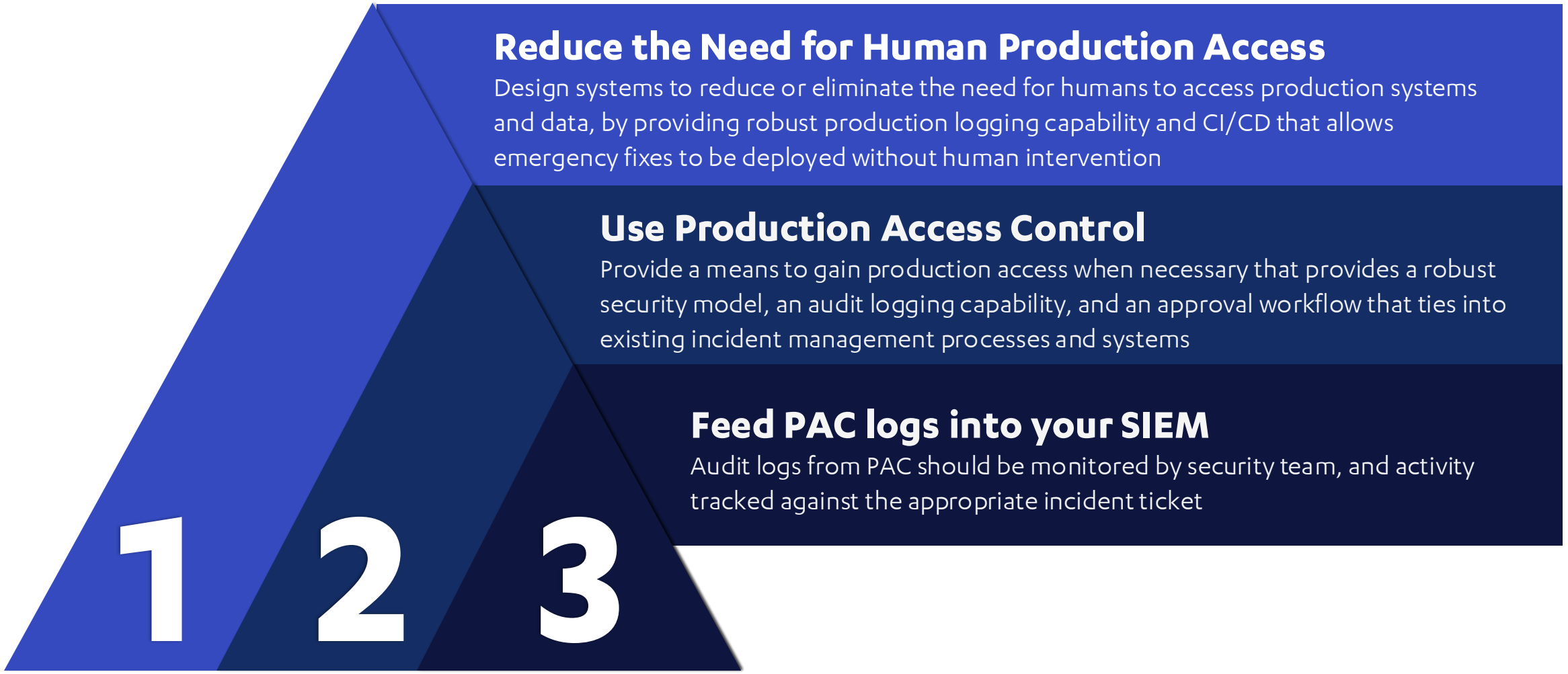
Eliminate long-lived credentials

Use provider-backed authentication where possible

Automate credential management and rotation



Production Access Control



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

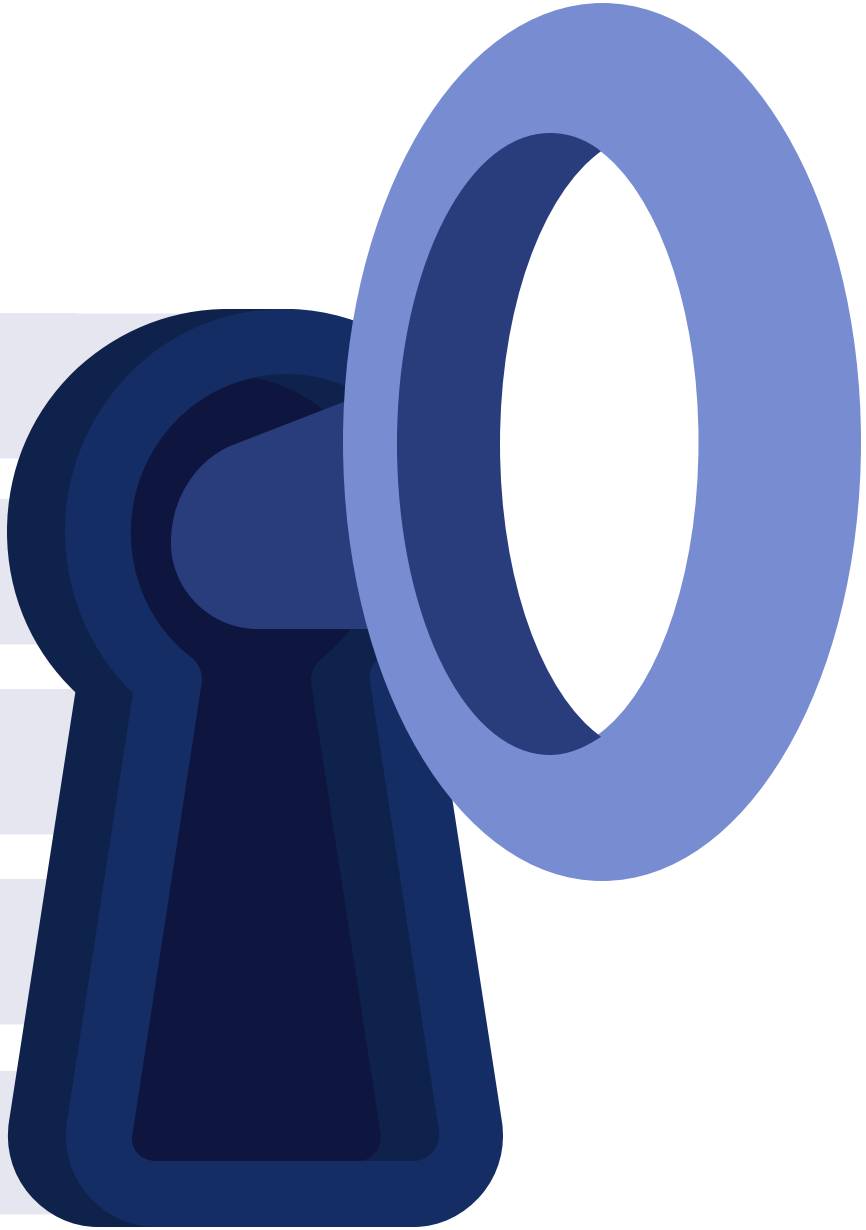
Provide a means to gain production access when necessary that provides a robust security model, an audit logging capability, and an approval workflow that ties into existing incident management processes and systems

Feed PAC logs into your SIEM

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

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?
- Use Secrets Manager / SSM Parameter Store!



Security Automation

04 Secrets Scanning

Scan repositories for keys, certificates etc.

TruffleHog
detect-secrets

02 IaC Scanning

Scan Infrastructure as Code in pipelines

Checkov
TFLint

IAM 03

Identify IAM misconfigurations

Cloudsplaining
StormSpotter
BloodHound
IAMSpy

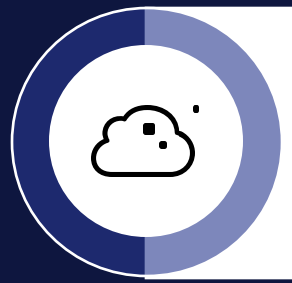
01 Configuration

Assess resources for configuration issues

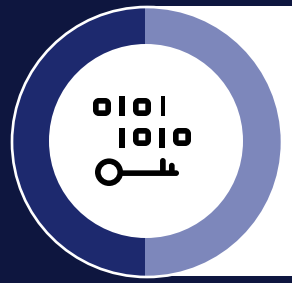
Prowler
ScoutSuite

Conclusions

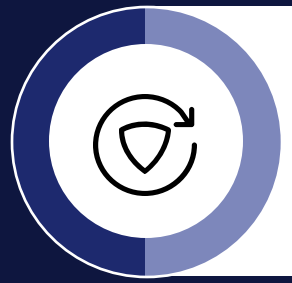
Conclusions



Security of the cloud extends to include a lot of external factors



Focus on identity, secrets management and CI/CD



Leverage automation and be smart about how you use humans

Thanks for listening!

Twitter: @nojonesuk

Blog: www.nojones.net