
13TH ANNUAL LEADERSHIP EVENT
CYBER SECURITY SUMMIT
cybersecuritysummit.org
RESILIENCE UNLOCKED
TITLE SPONSOR
Island
#cybersecuritysummit #css13

Unmasking AWS Deceptions

Unraveling Cloud Security's Sneaky Side

Itay Nachum

Sr. Director, Identity Threat Defense

proofpoint.

Agenda

- Introduction to Deceptions
- Introduction to AWS IAM
- Deceptions in AWS
- Tools

Clifford Paul “Cliff” Stool
(born June 4, 1950)

American Astronomer, Author and Teacher

BERKELEY LAB
LAWRENCE BERKELEY NATIONAL LABORATORY

BERKELEY LAB
LAWRENCE BERKELEY NATIONAL LABORATORY

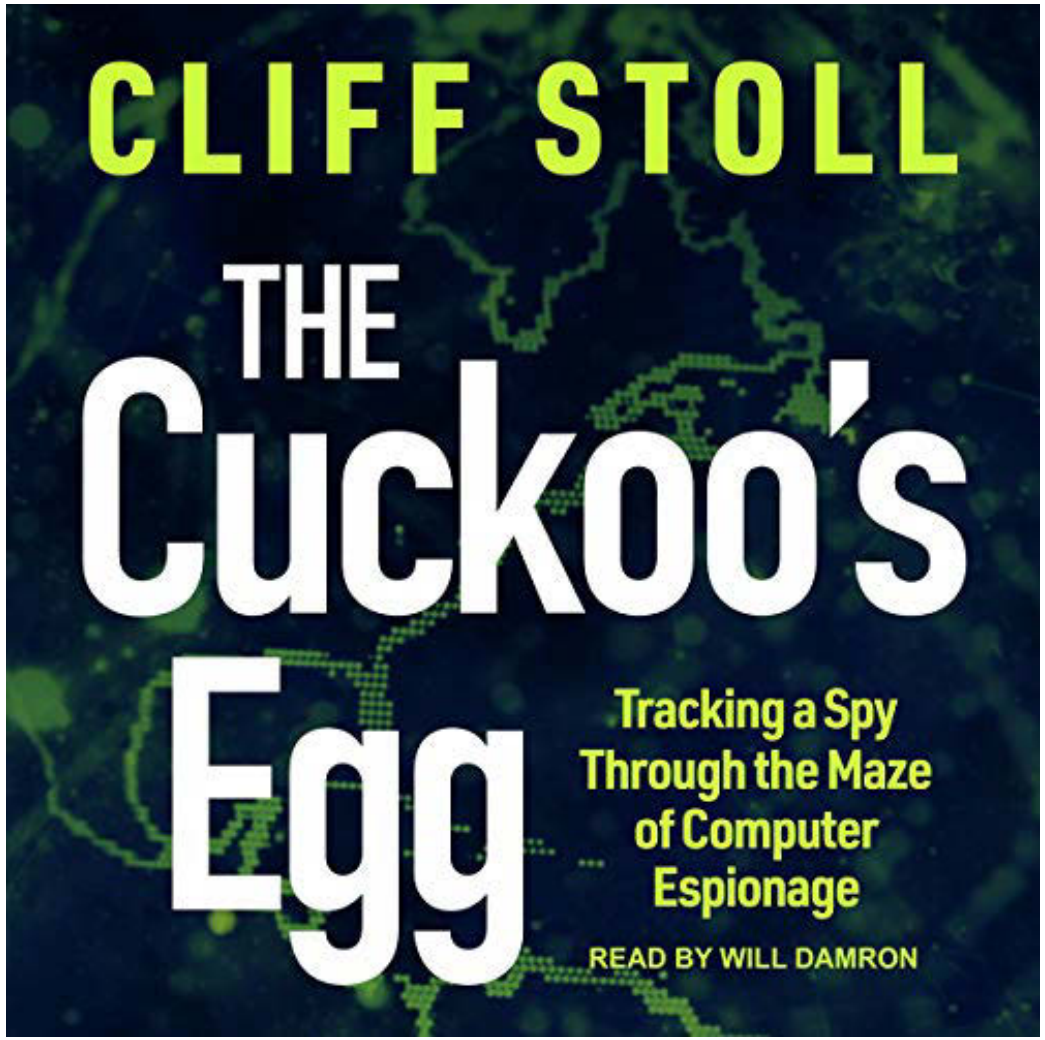
UNITED STATES ARMY


FREEDOM


I'M PROBABLY JUST GONNA
KEEP HIM ON THE LINE FOREVER

Markus Hess


КГБ СССР


CLIFF STOLL
THE
Cuckoo's
Egg
Tracking a Spy
Through the Maze
of Computer
Espionage
READ BY WILL DAMRON

Deception

Deception

“Deception is an attempt to manipulate the beliefs of others in order to influence their behaviour"

“Deception is simply leading someone to believe something that is not true, typically in order to gain some personal advantage"

“Deception techniques exploit cognitive biases of adversaries. To be able to do that, defenders need to learn as much as possible about the cognitive state of an adversary”

“The subtle point is that the deception planner should not seek to influence what the adversary believes, but what the planner wants the adversary to do”

Deception

“Deception is an attempt to manipulate the beliefs of others in order to influence their behaviour"

“Deception is simply leading someone to believe something that is not true, typically in order to gain some personal advantage"

“Deception techniques exploit cognitive biases of adversaries. To be able to do that, defenders need to learn as much as possible about the cognitive state of an adversary”

“The subtle point is that the deception planner should not seek to influence what the adversary believes, but what the planner wants the adversary to do”

Deception

Deception is an attempt to manipulate the beliefs of others for influencing their behavior, typically gaining some personal advantage.

Deception (Cyber)

A fake digital resource that mimics some characteristics of real resources

Cyber deception is a proactive security and defense tactic which hinges on deceiving bad actors and malicious attacks

A security resource whose value lies in being probed, attacked, or compromised

Cyber deception is a broad term for a wide variety of techniques that trick attackers into engaging with dummy digital resources, which don't serve authorized enterprise users. The sole purpose of these decoys - which can include servers, services, networks, files, user accounts and email accounts - is to reveal attacks in progress

Deception (Cyber)

A fake digital resource that mimics some characteristics of real resources

Cyber deception is a proactive security and defense tactic which hinges on deceiving bad actors and malicious attacks

A security resource whose value lies in being probed, attacked, or compromised

Cyber deception is a broad term for a wide variety of techniques that trick attackers into engaging with dummy digital resources, which don't serve authorized enterprise users. The sole purpose of these decoys - which can include servers, services, networks, files, user accounts and email accounts - is to reveal attacks in progress

Deception (Cyber)

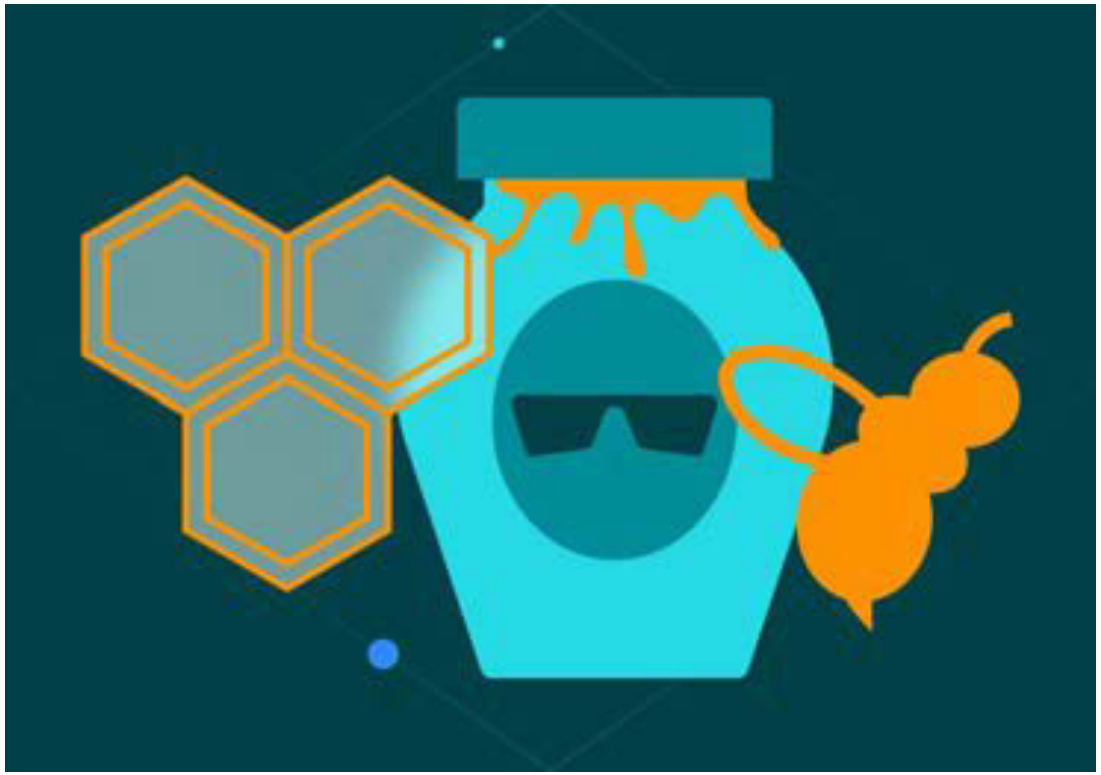
A strategic cybersecurity measure that leverages low-risk decoy digital assets to manipulate and mislead adversaries, ensuring genuine resources remain uncompromised without impact legitimate users.

Objectives:

- Enhancing threat detection
- Gathering intelligence on adversary tactics
- Delaying attacker progress
- Deterring against malicious activities

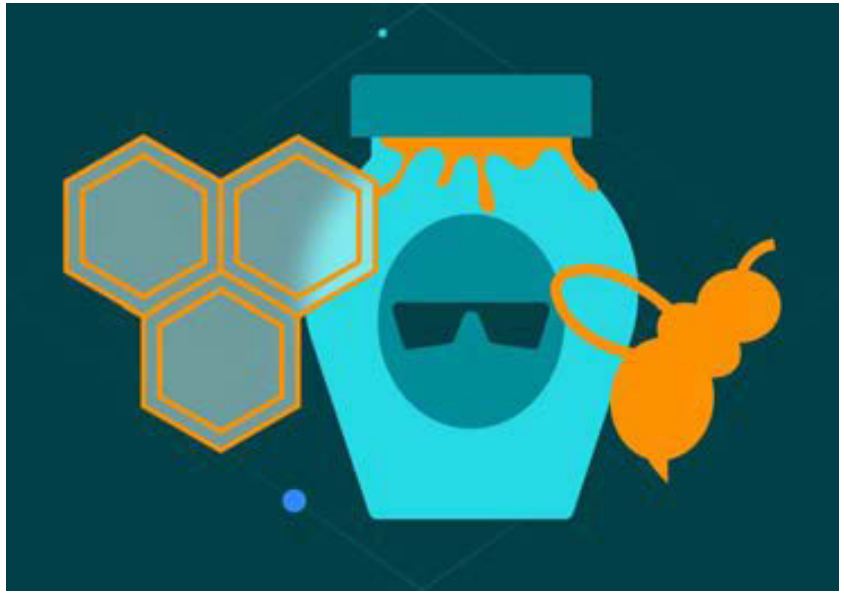

Honey-X

Honeypots



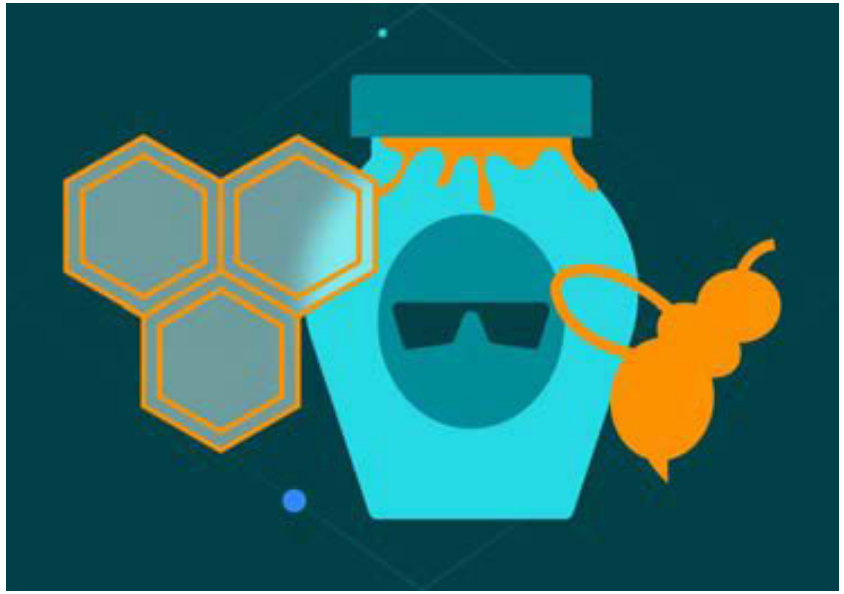
Two types of Honeypots

AWS Honeypots



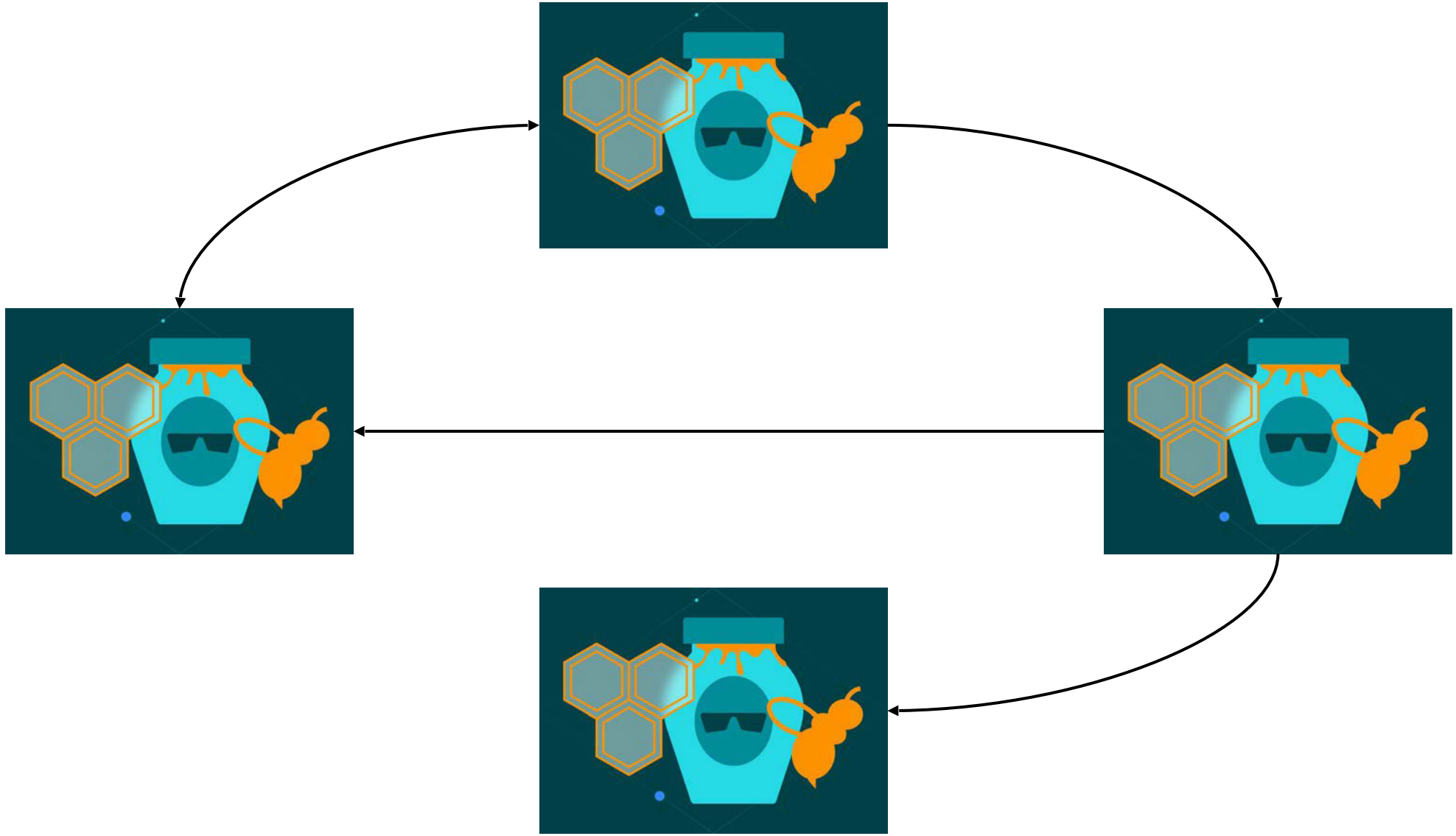
Disadvantages:

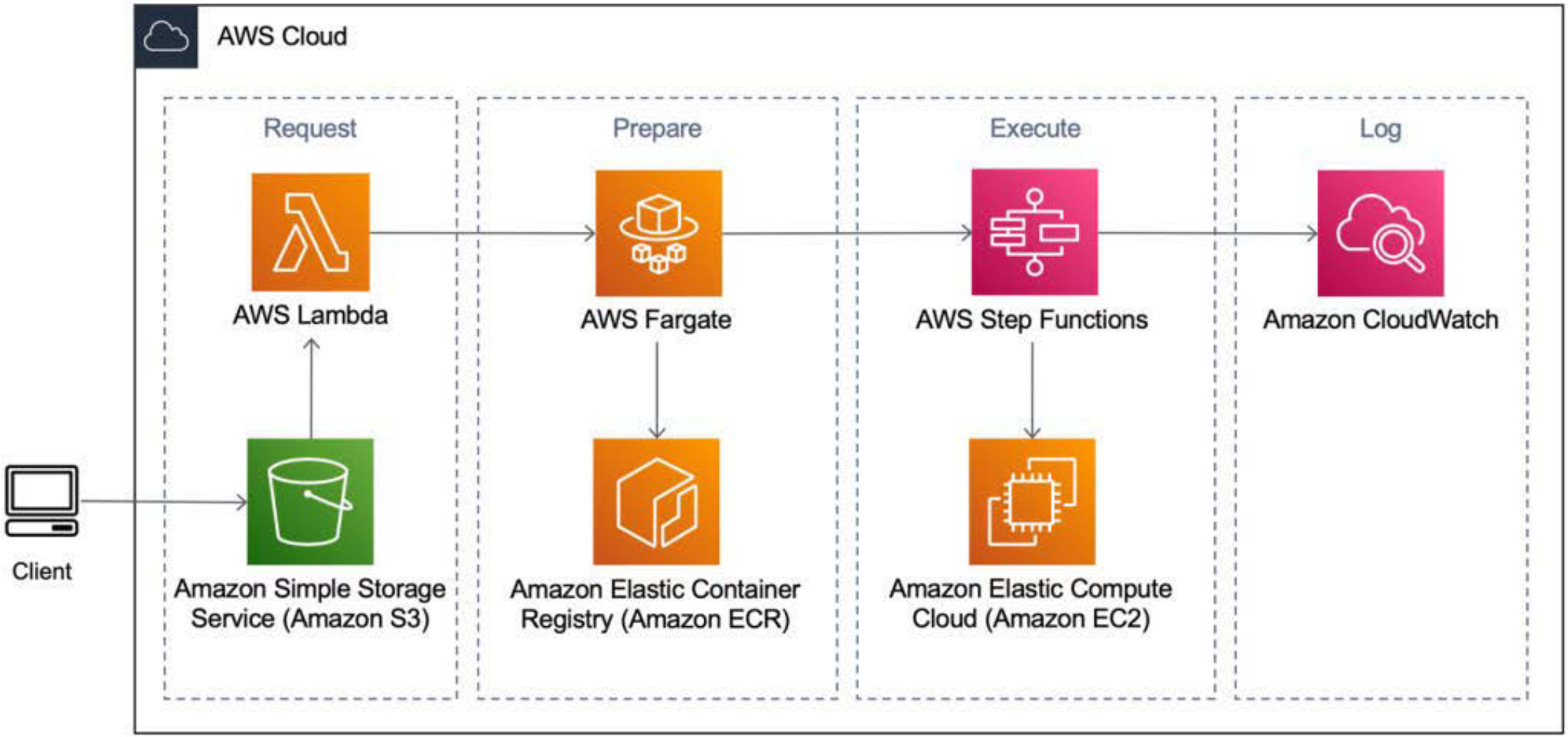
- Not scalable
- High maintenance
- Lacks plausible network traffic
- Can be abused

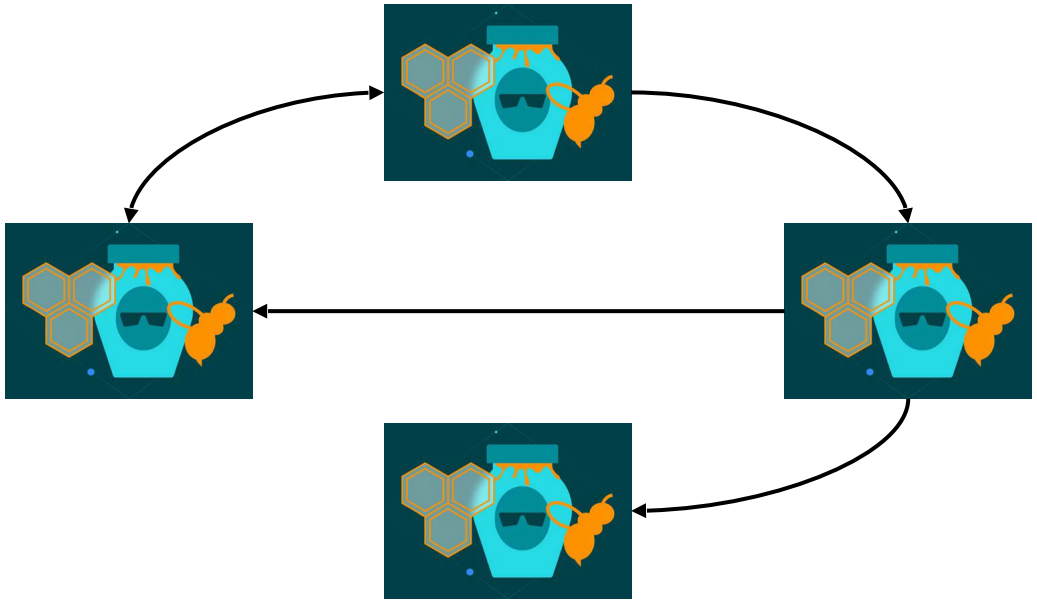


Disadvantages:

- Not scalable
- High maintenance
- Lacks plausible network traffic
- Can be abused



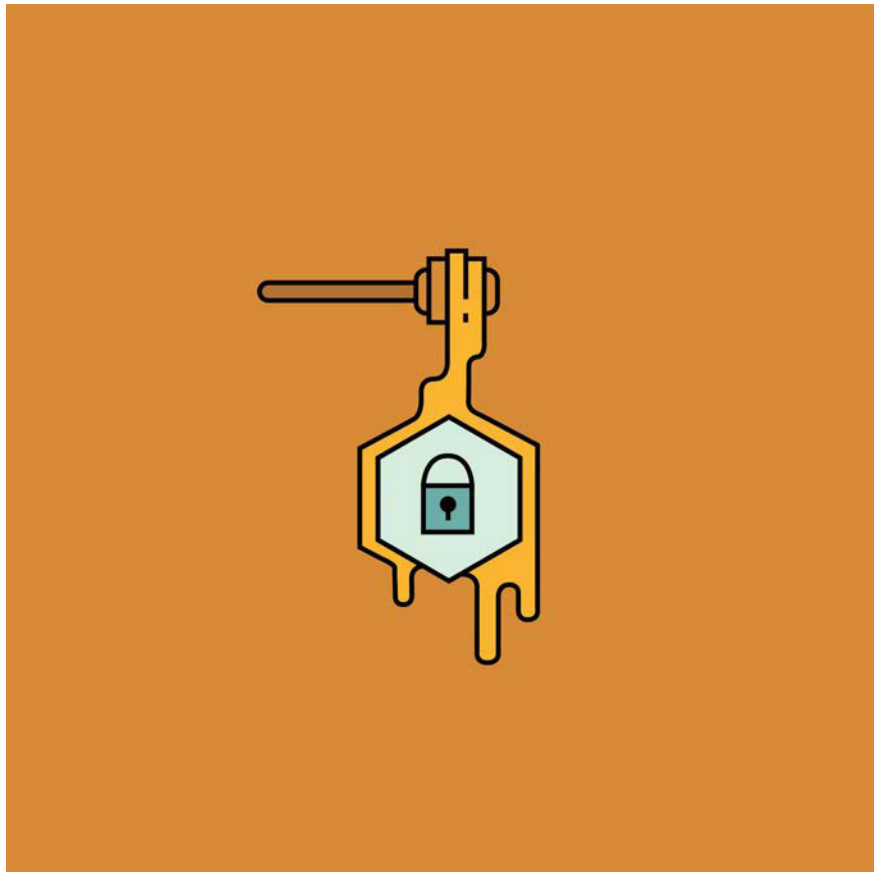

AWS Cloud
Request
Prepare
Execute
Log
AWS Lambda
AWS Fargate
AWS Step Functions
Amazon CloudWatch
Client
Amazon Simple Storage Service (Amazon S3)
Amazon Elastic Container Registry (Amazon ECR)
Amazon Elastic Compute Cloud (Amazon EC2)

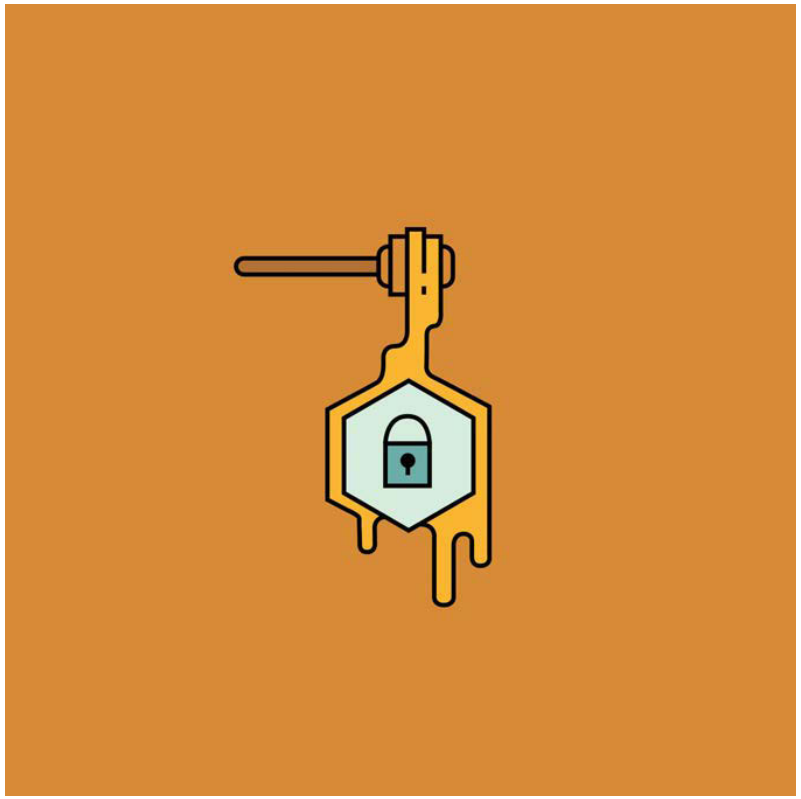


Disadvantages:

- Not scalable
- High maintenance
- ~~Lacks plausible network traffic~~
- ~~Can be abused~~

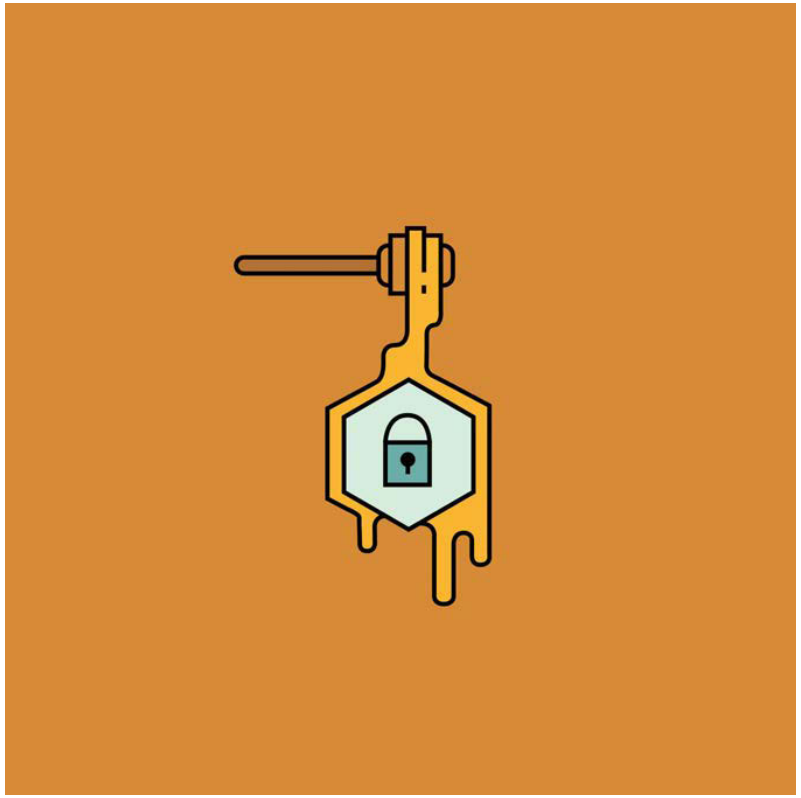
Honeytokens





Disadvantages:

- ~~Not scalable~~
- ~~High maintenance~~
- ~~Lacks plausible network traffic~~
- ~~Can be abused~~



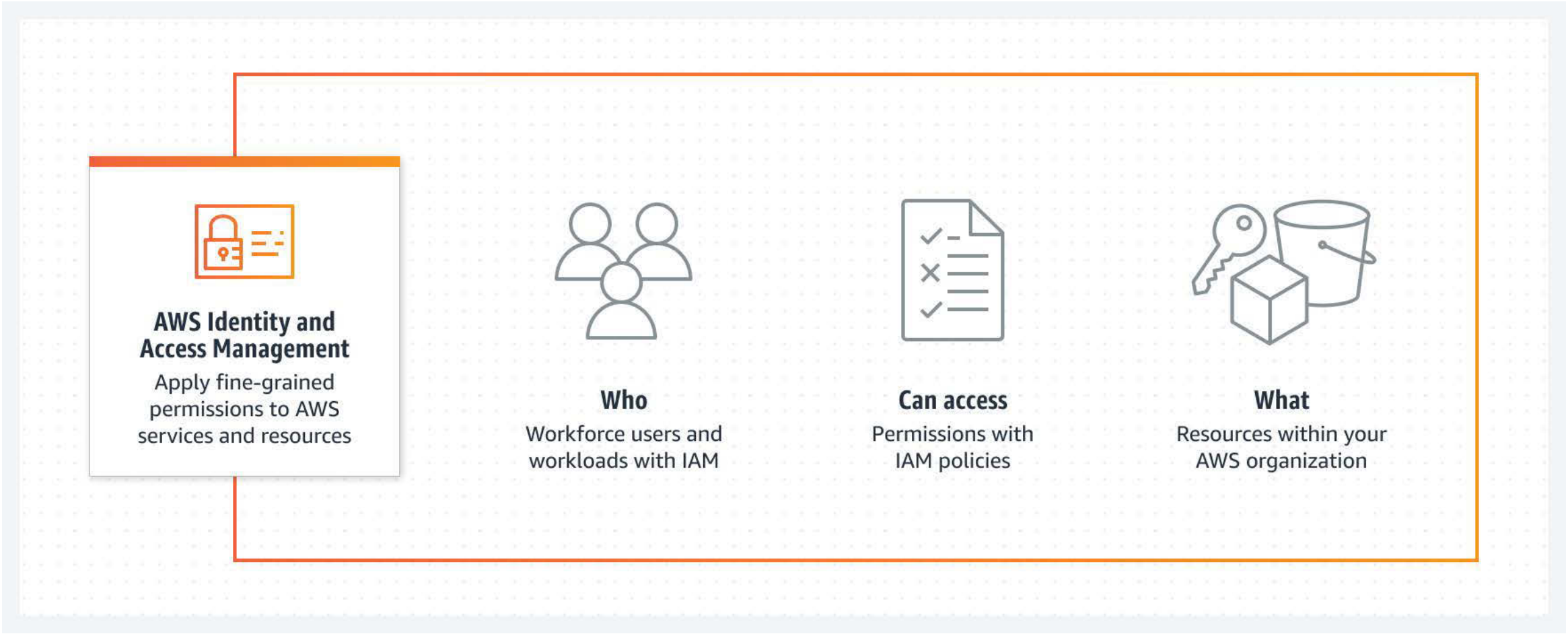
Advantages:

- Simplicity
- Ease
- Quiet
- Polymorphism
- Omnipresence

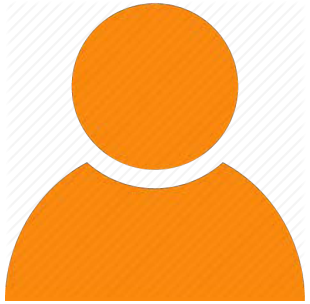
IAM

Identity and Access Management

101


AWS Identity and Access Management
Apply fine-grained permissions to AWS services and resources
Who
Workforce users and workloads with IAM
Can access
Permissions with IAM policies
What
Resources within your AWS organization

Users / Identities



AWS Tenant Account Owner User

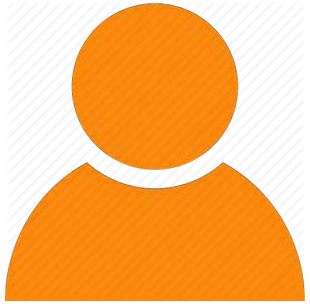
AWS IAM User

Federated User

Groups

Roles

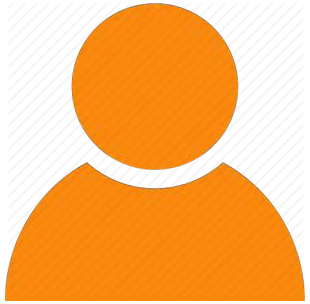
Types of Role Assumption



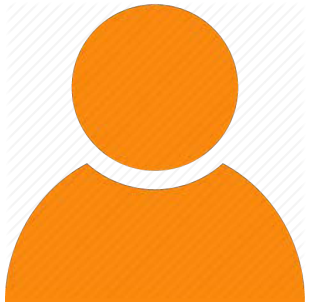
User

Service

The 3 Entities Assigned Permissions



Authentication



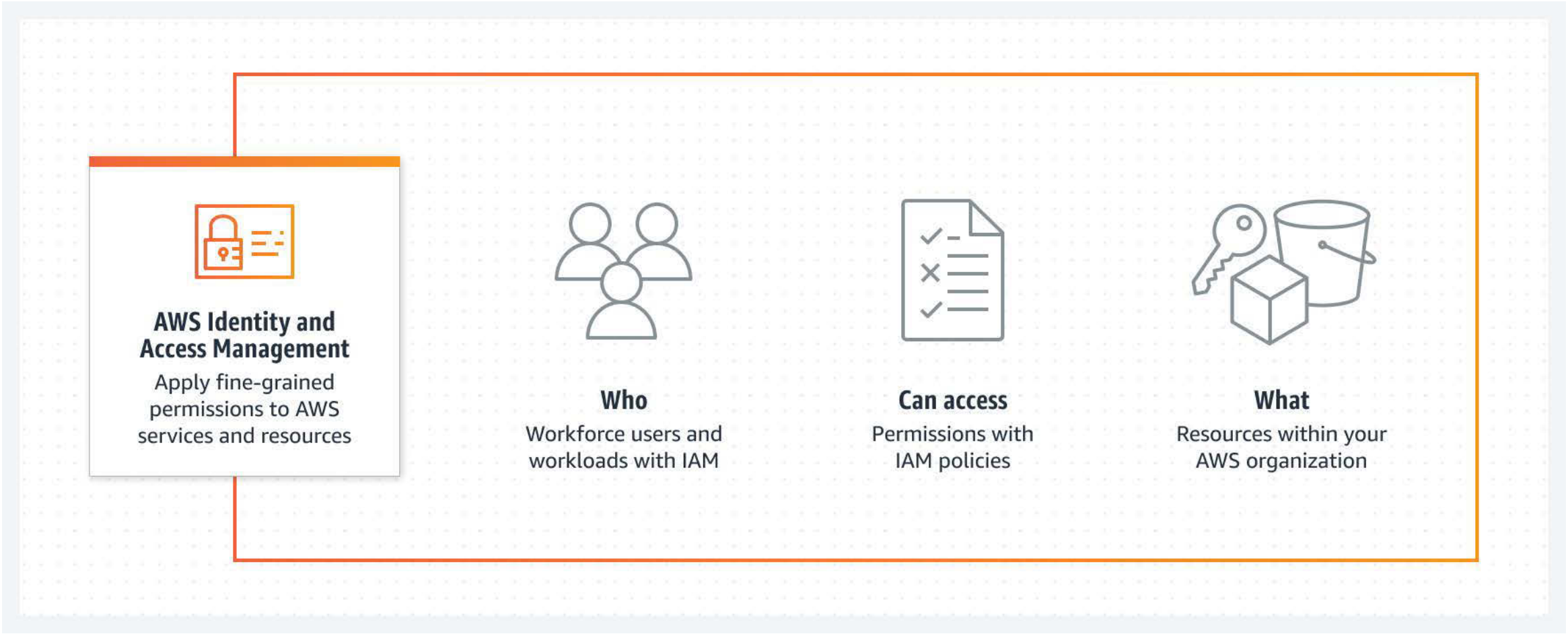
User Authentication

- Console password
- Access keys

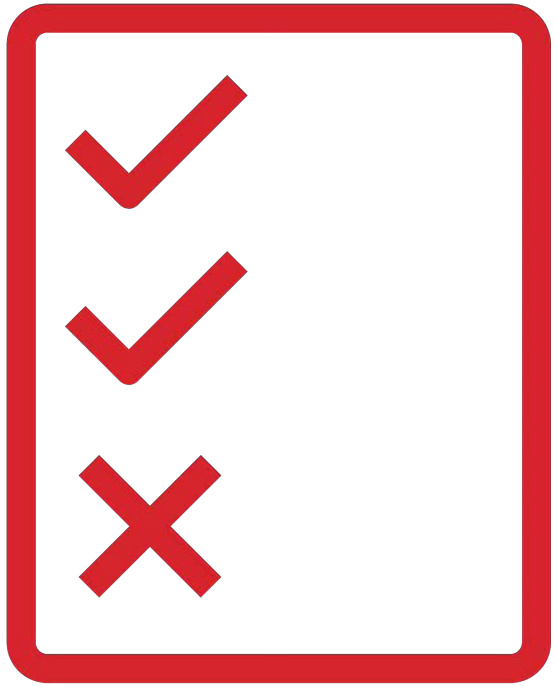
```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Role Authentication

```
aws sts get-session-token --duration-seconds 900
{
    "Credentials": {
        "SecretAccessKey": "rv1xIuMa6Frz1h5ojNW8BsguSGPiqkT7GNUEZZoL",
        "SessionToken":
"FQoGZXIvYXdzEOz//////////wEaDFONA2EY59z3Fe83tiKrAdJBHPpt5dcAOEYIv0XjtOOMfPDD
TgjRu8PVtMQrGswr3AqmIlPx2q9H8p6W67e4ypves23jBpHwC1SYDqZI6o0T+B4RILFAsNQfd8oOU
kzsNZX5bJN0zPUXxwRQmW33c+Ysu01tBsNX+GeNpF+jVEzDbC8TiQ7N7/YvzlKYd1hYBsOlmCYS0N
Dn6s3+oH0QvoGfip71C5maxZNfNCVKf0I4jRC39Hm3JwF8tyitr6zwBQ==",
        "Expiration": "2019-08-10T18:40:49Z",
        "AccessKeyId": "ASIAXWLG7NXMYTB2ON7Z"
    }
}
```


AWS Identity and Access Management
Apply fine-grained permissions to AWS services and resources
Who
Workforce users and workloads with IAM
Can access
Permissions with IAM policies
What
Resources within your AWS organization

IAM Policies

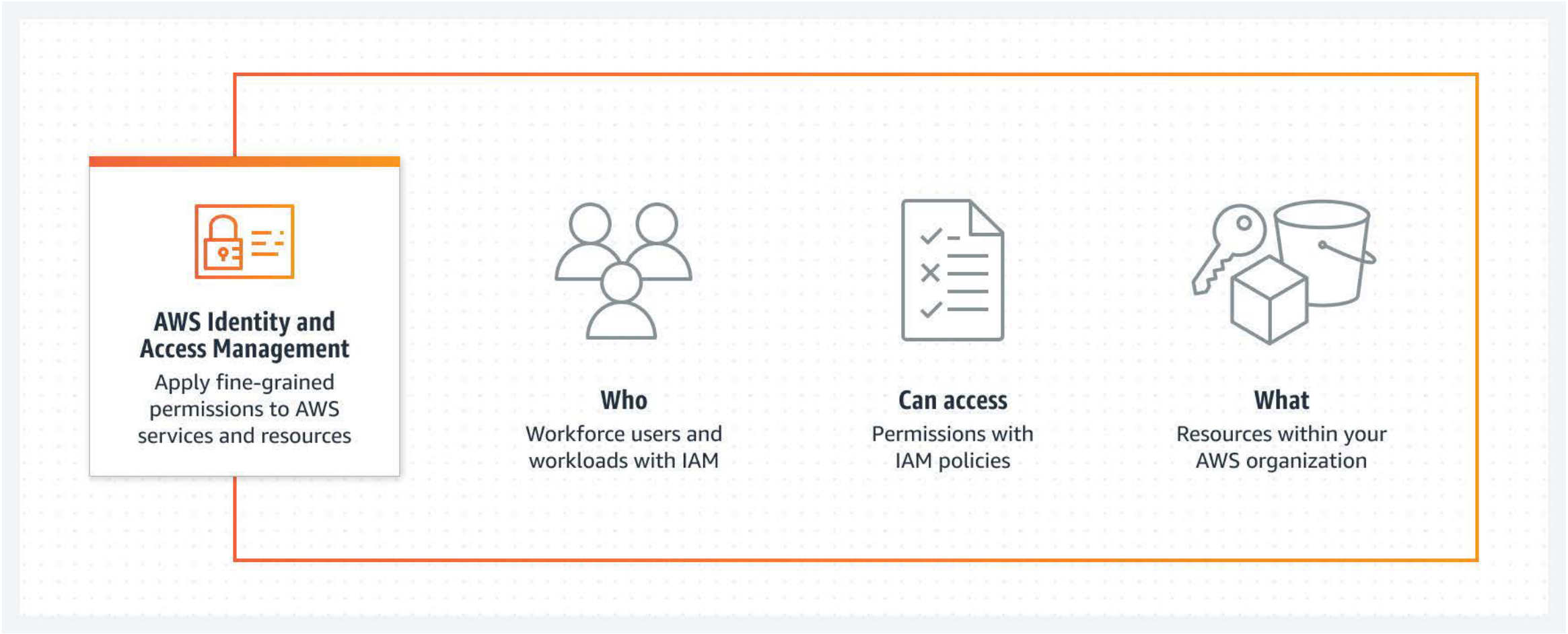


IAM Policies

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

IAM Policies

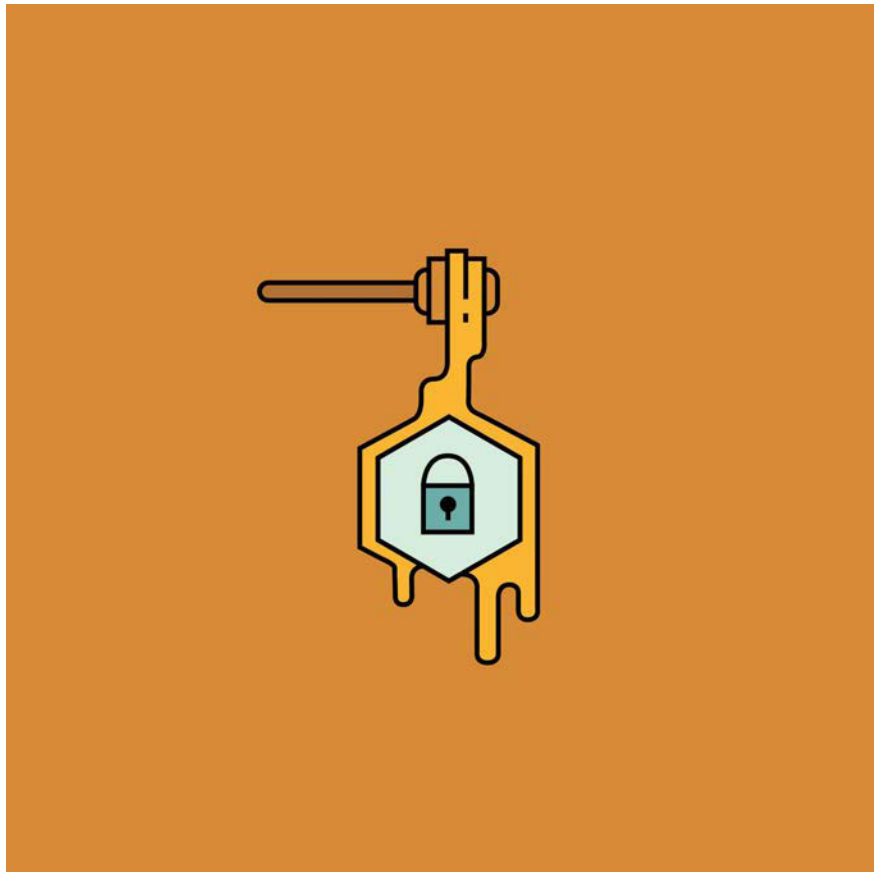
```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```


AWS Identity and Access Management
Apply fine-grained permissions to AWS services and resources
Who
Workforce users and workloads with IAM
Can access
Permissions with IAM policies
What
Resources within your AWS organization

“A resource is an object that exists within a service”


I'M BACK
WHAT DID I MISS?
memegenerator.net

Back to Honeytokens



Honeytokens in AWS

AWS IAM Resources
(Polymorphism)


THAT'S NICE
BUT HOW CAN I USE IT?
imgflip.com

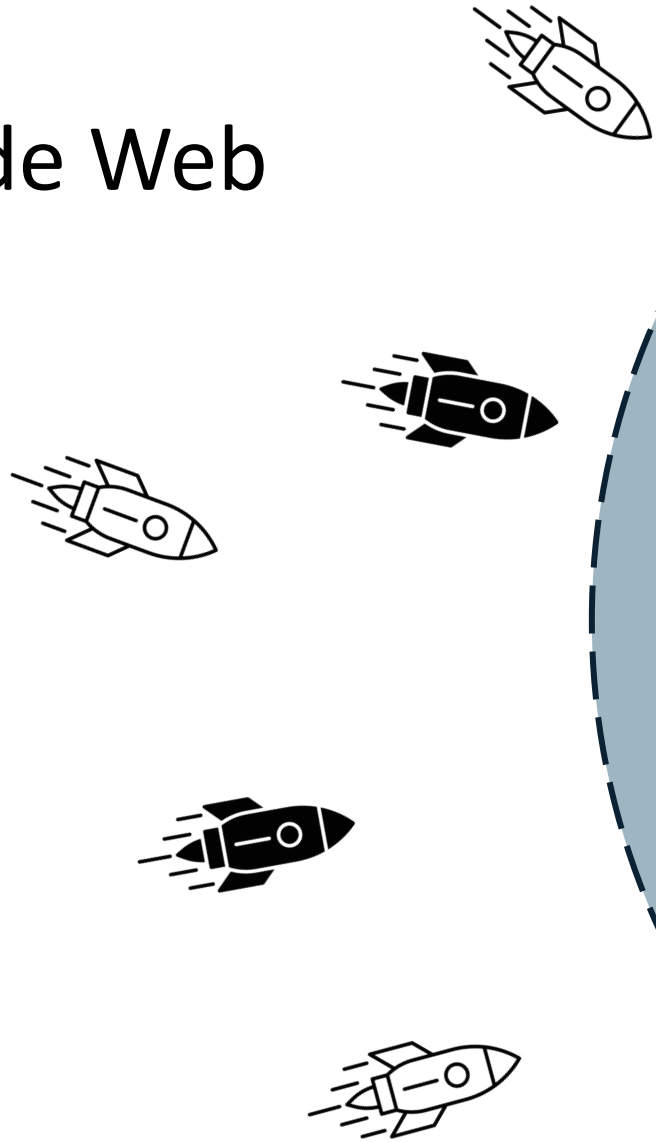
Perimeter Honeytokens

Lateral Movement Honeytokens

Perimeter Honeytokens

Collect intelligence & delay

World Wide Web



Perimeter Honeytokens in AWS:
Access Keys

Perimeter Honeytokens in AWS:
Access Keys

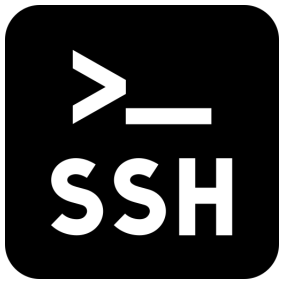
GitHub

On-Premise

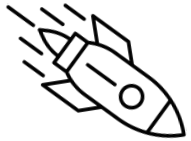
slack

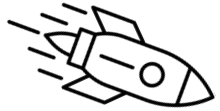
Word

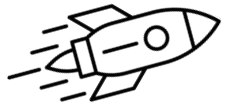
Perimeter Honeytokens and Honeypots in AWS



Effectiveness in Intelligence







2 minutes to exploit keys exposed on GitHub

3 minutes to access HTTP honeypot

4 minutes to access SSH honeypot

1 hour to access S3 buckets

Based on "**2023 Honeypotting in the Cloud Report**" published by the Orca research pod


I'M KIND.
NOT
NAIVE.
Don't confuse the two.

Lateral Movement Honeytokens

Detect & Deter

Lateral Movement Honeytokens in AWS:
Access Keys

Lateral Movement Honeytokens in AWS:
Access Keys

S3 Bucket

Amazon
EC2

Amazon
DynamoDB

Lateral Movement Honeytokens in AWS: Access Triggers

S3 Bucket

AWS Lambda

Alerting Mechanism

Amazon
CloudTrail

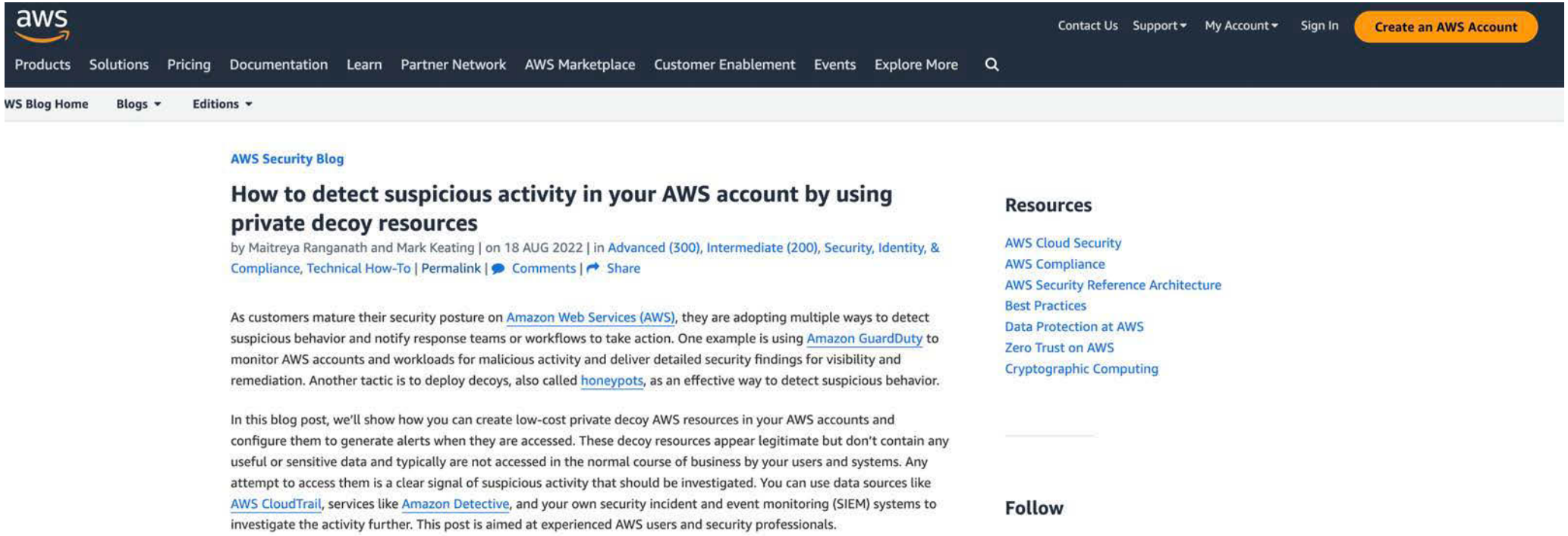
"AWS CloudTrail is an AWS service that helps you enable operational and risk auditing, governance, and compliance of your AWS account. Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail." (AWS)

Alerting Mechanism

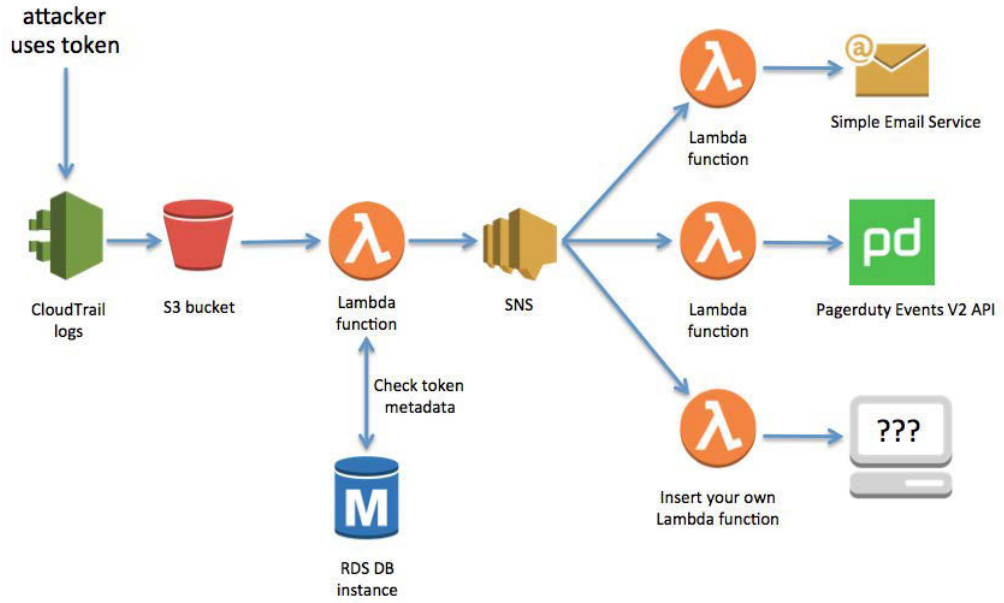
“AWS CloudTrail is an AWS service that helps you enable operational and risk auditing, governance, and compliance of your AWS account. Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail.” (AWS)

Tools for Deception in AWS

AWS Security Blog blogpost



SPACECRAB



Breach Detection at Scale with AWS Honey Tokens

Daniel Bourke
Sr. Security Analyst, Atlassian

Daniel Grzelak
Head of Security, Atlassian

Honey tokens, by which we mean credentials or database records or DNS entries that set off alarms if you look at them funny, are extremely helpful for securing your enterprise.

We'll go over some infrastructure we've built to help deploy a specific type of honey token (AWS credentials) at scale (i.e. in a reasonably automatable fashion), as well as some things we learned by *accidentally* leaking a bunch of AWS credentials all over the internet.

I. What is a honeytoken?

We're using 'honey token' in this paper as a stand-in for anything you can lock down and fire alerts from. This can be nearly anything, depending on your context and capabilities: In a database, a record that won't get returned in normal business queries, but will get returned by an unwary attacker running `'SELECT * FROM IMPORTANT_TABLE;'` can be a honey token, as long as you alert if that record is ever queried. If you control a DNS server, you can set up alarms on certain subdomains being resolved, and sprinkle links to them in your documentation, where your employees will never see it but a curious interloper will spider it. Alternatively you might put some bogus internal email addresses in your CMS and if they ever start getting spam, you know someone's been peeking at your stuff. All of these are relatively easy to create for one-off or low-scale deployments, and you should consider doing so (or using a freely-available third-party service to do it for you).

II. AWS keys as honeytokens

AWS keys make extremely good honey tokens, because they're very interesting to attackers (because if you find someone's AWS keys, you may have just found several thousand dollars worth of cryptocurrency mining hardware in someone else's cloud); and because you, the defender, can really easily secure AWS keys, and alert if anyone tries to use them. They also have the convenient properly of being found in an enormous variety of locations, from developers' desktops to server environment variables to three months deep in your chat

While we were waiting for that product to come to market, we wrote SPACECRAB. SPACECRAB is something you can deploy in an hour or so, which will provide you with an API endpoint you can use to create, update and dispose of AWS credentials, and a plethora (two) of alerting options (it's email or PagerDuty, but you can write your own as well).

Let's talk about an entirely hypothetical deployment scenario, where you've got a fleet of workstations, some kind of workstation management system that those workstations are plugged in to, an empty AWS account, and an insatiable thirst to secure the enterprise. You can leverage these assets in the following fashion:

1. Go to https://bitbucket.org/asecurityteam/spacecrab and clone the repo to your local machine.

2. Follow the instructions in the repo until you have a new SPACECRAB instance installed in your AWS account.

3. Write a script in the appropriate language for your workstations, that talks to your API gateway with your API token you've just made (Step 2 covers a lot of things), and stores the results somewhere on the workstation's file system, in a place an attacker would look. This might be ~/Downloads/accessKeys.csv, or somewhere on the windows desktop, or really anywhere useful. It's entirely up to you and your expectations around attacker behaviour.

4. Sit back and wait to get paged when one of those tokens is immediately used by an inquisitive bear or panda or something else touching your stuff.

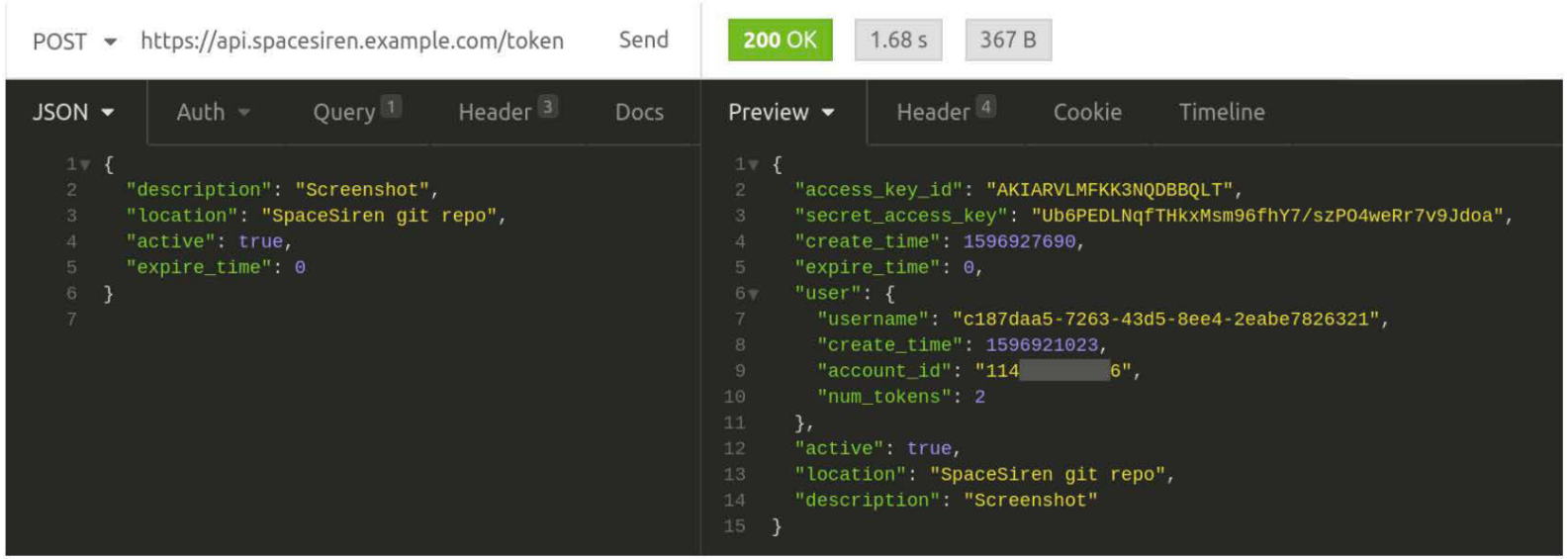
Having gone to all this trouble, you can now also add that script to your cloud service deployment pipeline, ensuring there's a set of extremely juicy looking variables waiting for the next person to get remote code execution on your service. Or add it to all your private repositories with a commit hook, or... you'll find somewhere to put them.

III. How does it work?

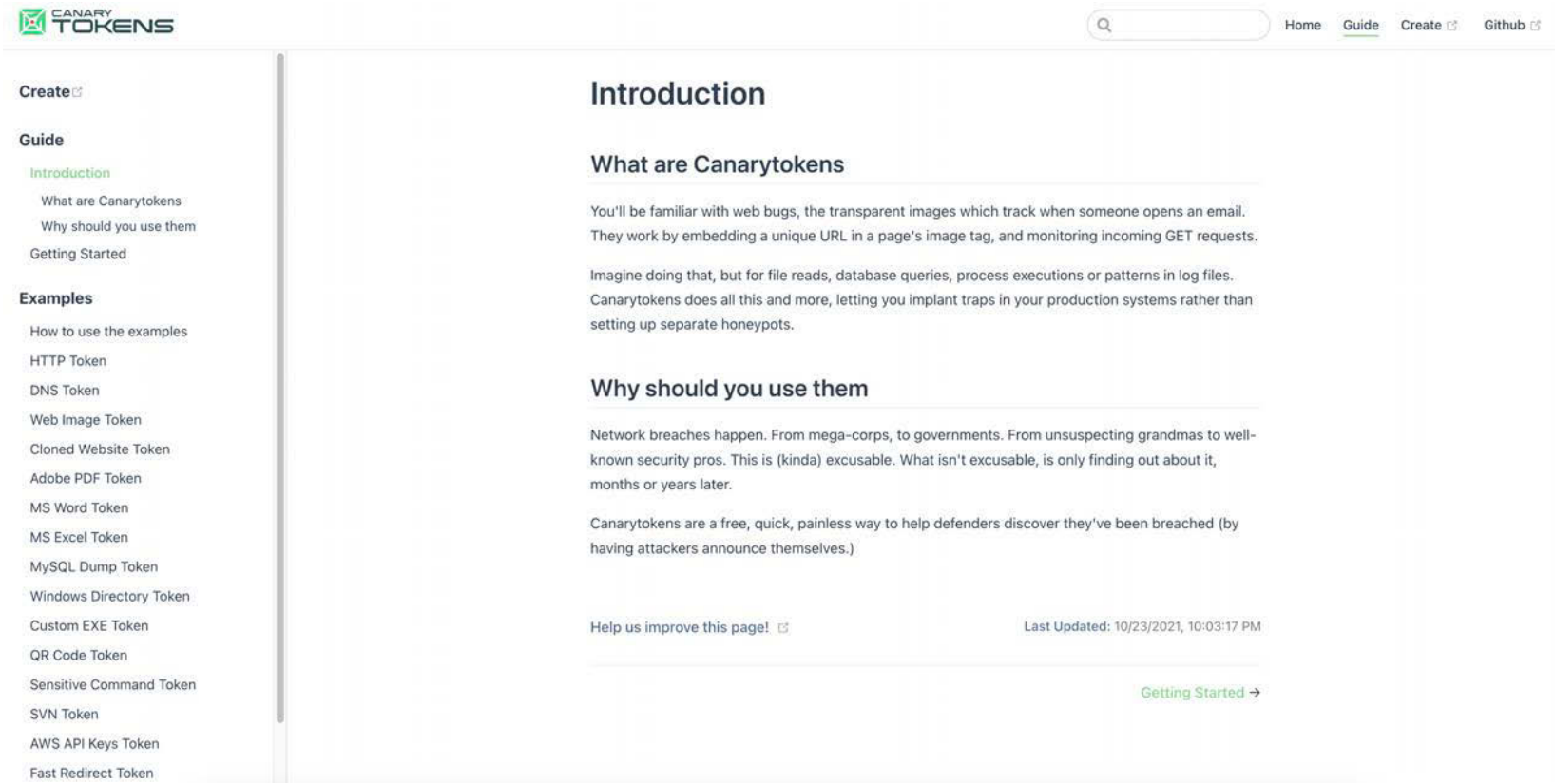
SpaceSiren

How It Works

- SpaceSiren provides an API to create no-permission AWS IAM users and access keys for those users.
- You sprinkle the access keys wherever you like, for example in proprietary code or private data stores.
- If one of those sources gets breached, an attacker is likely to use the stolen key to see what they can do with it.
- You will receive an alert that someone attempted to use the key.



Canarytokens



GPT-4

Google Bard

ANTHROPIC
CLAUDE 2

Thank you!