




Hello everyone! Let's get started.



# \$whoami

10+ year veteran of IT, Security Operations, Threat Hunting, Incident Response, Threat Research, and Forensics  
GitHub



- <https://github.com/sonofagl1tch>

Career highlight

- Time's person of the year 2006

What am I giving away? A full detonation lab built automatically by cloudformation

- <https://github.com/sonofagl1tch/AWSDetonationLab>



My name is Ryan Nolette and I have been a security practitioner in various capacities for the last 15 or so years.

Today I will be walking you through a practical example of detecting and investigating an attack in the cloud as well as giving away a working detonation lab filled with opensource software that you can use to run preconfigured attacks and learn what logging services have visibility into different attacks. Hopefully this will help you learn how these different log sources work together to provide a full picture of an event.

# Agenda


- Overview
  - Who did what and when?
  - Common Techniques
  - Common Visibility Tools and Their AWS Equivalent
- Increasing visibility until you have accountability
  - Common Logging
    - Authentication
    - Endpoint
    - Network
    - Vulnerabilities
    - Configuration
- End to end example
  - Detonation Lab
  - Logging Pipelines and Services
- Finding What Matters



In this presentation I am going to cover a brief overview of a few AWS logging sources and how they compare to the most common logging sources in a typical enterprise network and an end to end example using the detonation lab that I am giving away

# Who did what and when

- These are the 3 pillars of each stage of scoping the event
- Will be modified for each iteration
- The analysts should be able to start at any of the stages and complete the cycle



The diagram illustrates the concept of 'Visibility' leading to 'Accountability'. A funnel shape contains three circles labeled 'What', 'Who', and 'When'. An arrow points from the funnel down to the word 'Accountability'.

As part of any investigation, I strive to answer 3 questions, who did what and when. This is an iterative process and will be repeated for every step in the investigation. I should be able to start with knowledge of any of the 3 and find the other 2.

For example, I start my investigation with an IP address. I have the who. Next I need to find the time scope of when it was used. That is the when. Then I find that this IP address attempted to ssh into my instance 300 times in 10 minutes. This is the what. With these 3 questions answered I can reliably say that this external attacker tried to brute force ssh access into my instance.

I know this is a very simplistic example but I want to very clearly explain these tenants

# Common Techniques

## What's Their Goal?


- OS hardening
- Config management
- Identity Management
- Process monitoring

Visibility → Accountability

Who What When

To expand on these tenants of investigation, I break them into 2 buckets. Visibility and accountability. If you cannot see something, you can not account for it. You must keep increasing visibility until you can account for each event in your environment with 3 simple questions, who did what and when.

Now I know this end goal is next to impossible but that is part of the iterative process, security is a journey, not an end goal.



## Common Visibility Tools and Their AWS Equivalent

<u>Traditional Tool</u>	<u>AWS equivalent</u>
IDS/IPS	guardDuty
DLP	Macie
EDR	Cloudwatch + osquery, GRR
Netflow	Cloudwatch + VPCFlow
DNS	Cloudwatch + Route53
Access and authentication auditing	CloudTrail
Active Directory	Directory Service
Identity Management	IAM
Single Sign On	AWS SSO
Vulnerability scanner	Inspector
Configuration Management	AWS config
Logging	Cloudwatch + Firehose + Lambda

So how do I get visibility in the cloud? Simply put, the same way you do in your on prem network. Why complicate things? These AWS tools are the servicewise cloud equivalent of their common on prem twinsies.

Who here has an IPS?

Who here has a Vulnerability scanner?

Who here collects netflow or other network traffic logs?

If you want to know the equivalent to other traditional controls that I haven't mentioned here, find me after the presentation and I will help you find what you want. I didn't have enough room here to mention them all.

# Increasing visibility until you have accountability



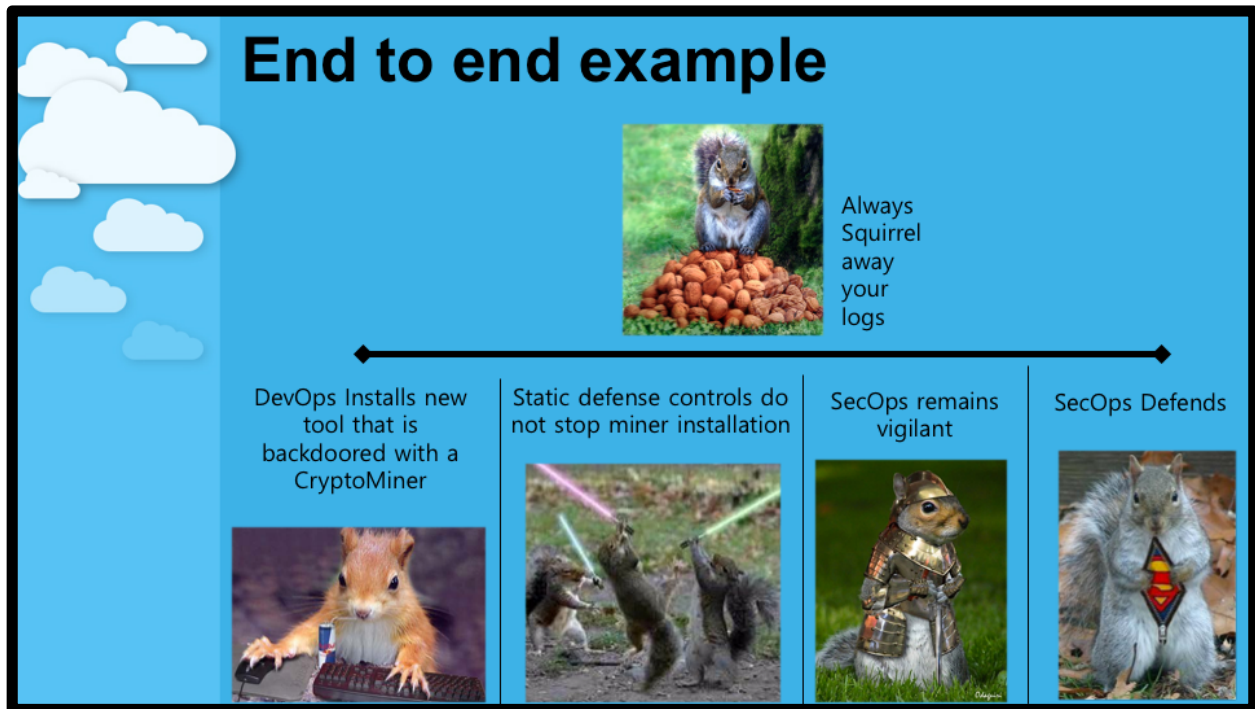
The process of asking who did what and when and increasing logging and controls until you can answer those questions for every scenario you can think of.

<u>OS hardening</u>	<u>Logging</u>	<u>Authentication</u>	<u>Endpoint</u>	<u>Network</u>	<u>Vulnerabilities/Configuration</u>
<ul style="list-style-type: none"><li>• CIS guidelines audit and hardening scripts.</li><li>• Additional logging and hardening scripts created by experience over time.</li></ul>	<ul style="list-style-type: none"><li>• Common logging like auth logs and process creation etc</li></ul>	<ul style="list-style-type: none"><li>• /var/log/secure</li><li>• IAM logs</li><li>• IAM roles</li><li>• IAM policies</li></ul>	<ul style="list-style-type: none"><li>• EDR</li><li>• HIDS</li><li>• Cloudwatch</li><li>• GuardDuty</li></ul>	<ul style="list-style-type: none"><li>• IDS</li><li>• Netstat</li><li>• Tcpdump</li><li>• Vpc flow logs</li><li>• Dns route 53 logs</li></ul>	<ul style="list-style-type: none"><li>• Generic vuln scanner</li><li>• Inspector</li><li>• NVD/CVE usage</li><li>• Aws config</li><li>• OS hardening</li><li>• Applications config</li></ul>



Expanding on our list of equivalent tools I want to show few options for each of the above categories

Take a few ideas, go nuts 😊



Now let's talk about some practical application of what I have said so far.

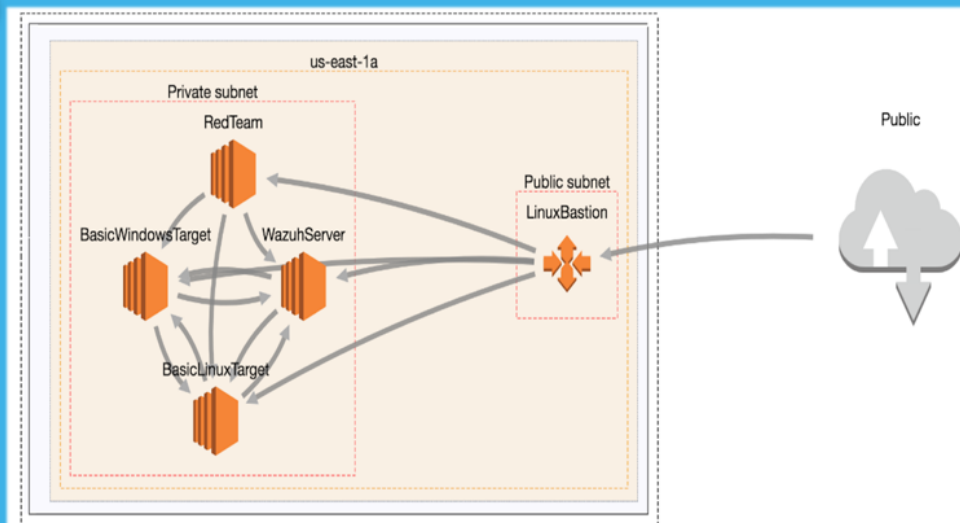
The scenario is that a devops engineer has installed a new tool they needed for monitoring their application but accidentally installs an RPM that includes a cryptominer.

The endpoint doesn't have any whitelisting software on it and the rpm meets all conditions for installing an application from an RPM.

Secops notices a spike in outbound traffic to china and investigates.



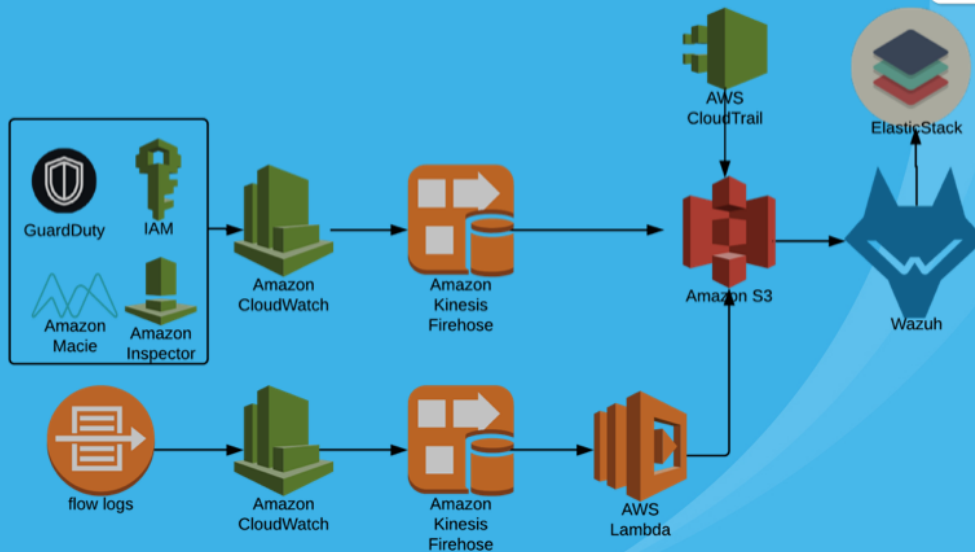
# Detonation Lab Topology



Now before we go into the investigation, let's talk about the lab we are working in.

The topology of the detonation lab is 4 hosts (3 linux and 1 windows) behind a bastion host all within their own VPC.

# Pipelines

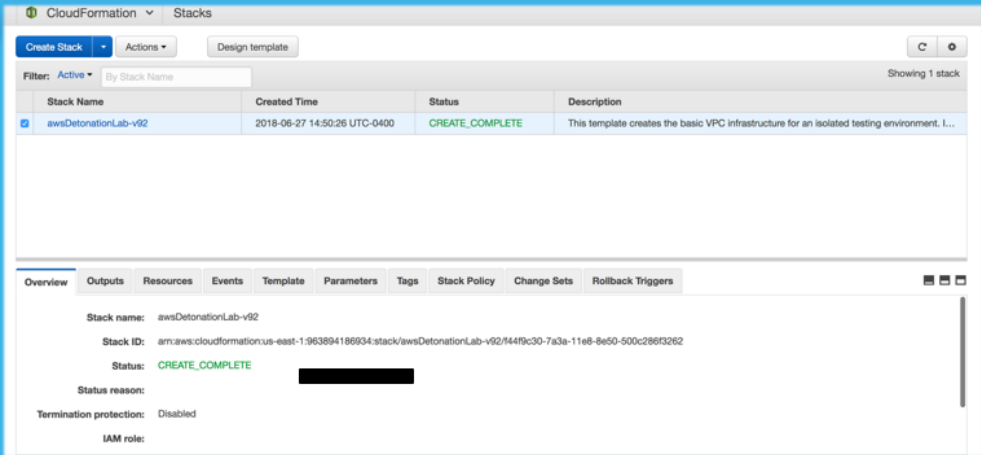


To get logs from various services I am using 3 pipelines. The first pipeline allows for a service to generate logs, trigger an event rule in cloudwatch, sends the logs to kinesis firehose, and writes the logs to an s3 bucket. These logs saved to s3 are then read by a wodle module on the wazuh server and ingested into Kibana.

The second pipeline mimics the first but with the added feature of using lambda to enhance the logs with metadata before writing to s3.

The third pipeline is for the EDR logs which are sent directly to wazuh from the agents.


# CloudFormation



The screenshot displays the AWS CloudFormation console. At the top, there's a navigation bar with 'CloudFormation' and 'Stacks'. Below this, a table lists the stacks. The first stack, 'awsDetonationLab-v92', is highlighted. Its status is 'CREATE\_COMPLETE', and its description is 'This template creates the basic VPC infrastructure for an isolated testing environment. I...'. Below the table, the 'Overview' tab is selected, showing details for the stack: 'Stack name: awsDetonationLab-v92', 'Stack ID: arn:aws:cloudformation:us-east-1:963894186934:stack/awsDetonationLab-v92/f44f9c30-7a3a-11e8-8e50-500c2863262', 'Status: CREATE\_COMPLETE', 'Status reason: [REDACTED]', 'Termination protection: Disabled', and 'IAM role: [REDACTED]'.

CloudFormation allows you to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all regions and accounts.

How did I automate the creation of this lab? Cloudformation. This allows me to build a simple or complex template to stand up systems quick. Think of it similar to chef or ansible except this can configure the majority of all aws services including networking and identity management for the entire account.



# S3

Create bucket

Delete bucket

Empty bucket

28 Buckets 0 Public 1 Regions

awsdetonationlab-v92-s3bucketcloudtrail-v9d4yqs8ytsh	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketguardduty-jkrem5i7jb	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketiam-1y00gbul8vi82	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:32 PM GMT-0400
awsdetonationlab-v92-s3bucketinspector-1xz05by24ypuk	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:32 PM GMT-0400
awsdetonationlab-v92-s3bucketmacie-1ty3blodhx6ff	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketpcflow-167orji11dqu1	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:32 PM GMT-0400

Amazon S3 is object storage built to store and retrieve any amount of data from anywhere

This is my storage.

# VPC

VPC Dashboard

Filter by VPC:  
Select a VPC

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

Egress Only Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

Endpoint Services

NAT Gateways

Peering Connections

Create VPC Actions

Search VPCs and their proper

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options set	Route table	Network ACL
awsDetonationLab-v92	vpc-734da209	available	172.16.0.0/27		dopt-7016980b	rtb-d91eb8a6	acl-167d356c

vpc-734da209 | awsDetonationLab-v92

Summary

CIDR Blocks

Flow Logs

Tags

VPC ID: vpc-734da209 | awsDetonationLab-v92

State: available

IPv4 CIDR: 172.16.0.0/27

IPv6 CIDR:

DHCP options set: dopt-7016980b

Route table: rtb-d91eb8a6

Network ACL: acl-167d356c

Tenancy: Default

DNS resolution: yes

DNS hostnames: yes

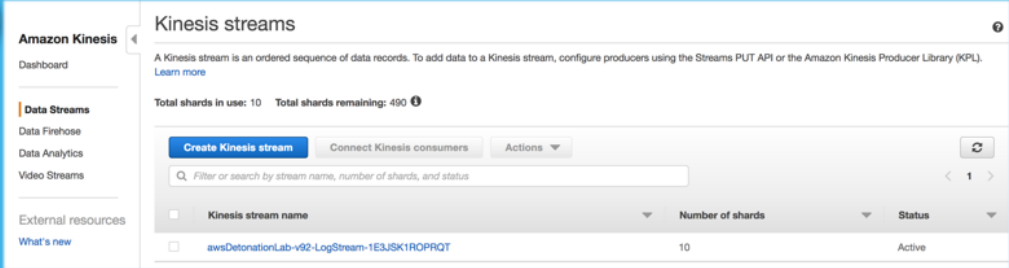
ClassicLink DNS Support: no

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define.

In my lab I am using an isolated VPC to control access to resources for the attacks.

13

# VPC Flow - Kinesis Stream



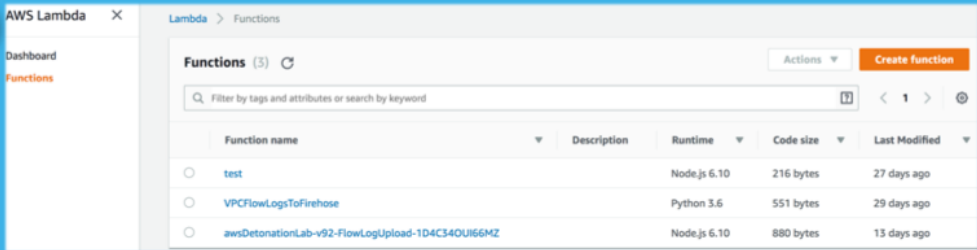
The screenshot displays the Amazon Kinesis console interface. On the left is a navigation sidebar with options: Dashboard, Data Streams (selected), Data Firehose, Data Analytics, Video Streams, External resources, and What's new. The main content area is titled 'Kinesis streams' and includes a description: 'A Kinesis stream is an ordered sequence of data records. To add data to a Kinesis stream, configure producers using the Streams PUT API or the Amazon Kinesis Producer Library (KPL). Learn more'. It also shows 'Total shards in use: 10' and 'Total shards remaining: 490'. Below this are buttons for 'Create Kinesis stream', 'Connect Kinesis consumers', and 'Actions'. A search bar is present with the placeholder 'Filter or search by stream name, number of shards, and status'. A table lists the streams:

<input type="checkbox"/>	Kinesis stream name	Number of shards	Status
<input type="checkbox"/>	awsDetonationLab-v92-LogStream-1E3JSK1R0PRQT	10	Active

Amazon Kinesis Data Streams (KDS) is a massively scalable and durable real-time data streaming service. KDS can continuously capture gigabytes of data per second from hundreds of thousands of sources such as website clickstreams, database event streams, financial transactions, social media feeds, IT logs, and location-tracking events.

This is the netflow monitoring used for the lab.

# Lambda



The screenshot shows the AWS Lambda console interface. On the left is a navigation menu with 'Dashboard' and 'Functions'. The main area is titled 'Functions (3)' and contains a search bar and a table of functions. The table has columns for Function name, Description, Runtime, Code size, and Last Modified. Three functions are listed: 'test' (Node.js 6.10, 216 bytes, 27 days ago), 'VPCFlowLogsToFirehose' (Python 3.6, 551 bytes, 29 days ago), and 'awsDetonationLab-v92-FlowLogUpload-1D4C340U66M2' (Node.js 6.10, 880 bytes, 13 days ago). A 'Create function' button is in the top right.

Function name	Description	Runtime	Code size	Last Modified
test		Node.js 6.10	216 bytes	27 days ago
VPCFlowLogsToFirehose		Python 3.6	551 bytes	29 days ago
awsDetonationLab-v92-FlowLogUpload-1D4C340U66M2		Node.js 6.10	880 bytes	13 days ago

AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume - there is no charge when your code is not running.

I use lambda to run code against logs I am generating to enhance it with additional details.

# Firehose

Kinesis Firehose delivery streams continuously collect, transform, and load streaming data into the destinations that you specify.

[Create delivery stream](#) [Test with demo data](#) [Delete](#)

Filter or search by name

Viewing 1 - 5 of 5 delivery streams

Name	Status	Created	Source	Record transformation	Destination
awsDetonationLab-v92-FirehosedeliverystreamGuardDuty-1ANGKIB81L550	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 <a href="#">awsdetonationlab-v92-s3bucketguardduty-jkem5i7jb</a>
awsDetonationLab-v92-FirehosedeliverystreamIAM-1M0ZMRPN00530	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 <a href="#">awsdetonationlab-v92-s3bucketiam-1y00gbu8v82</a>
awsDetonationLab-v92-FirehosedeliverystreamInspector-SKFM22924VLJ	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 <a href="#">awsdetonationlab-v92-s3bucketinspector-1xz05by24ypuk</a>
awsDetonationLab-v92-FirehosedeliverystreamMacie-3ROVXTDXED19	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 <a href="#">awsdetonationlab-v92-s3bucketmacie-1tv3blodhxdfl</a>

Amazon Kinesis Data Firehose is the easiest way to reliably load streaming data into data stores and analytics tools.

This is how I write the logs to s3



# IAM

Search IAM

Dashboard  
Groups  
**Users**  
Roles  
Policies  
Identity providers  
Account settings  
Credential report  
Encryption keys

User ARN: `arn:aws:iam::963894186934:user/wazuh-user`  
Path: `/`  
Creation time: 2018-05-24

Permissions Groups Security credentials Access Advisor

Add permissions Attached policies: 1 Add inline policy

Policy name Policy type

Attached directly

wazuh-read-cloudTrail Managed policy

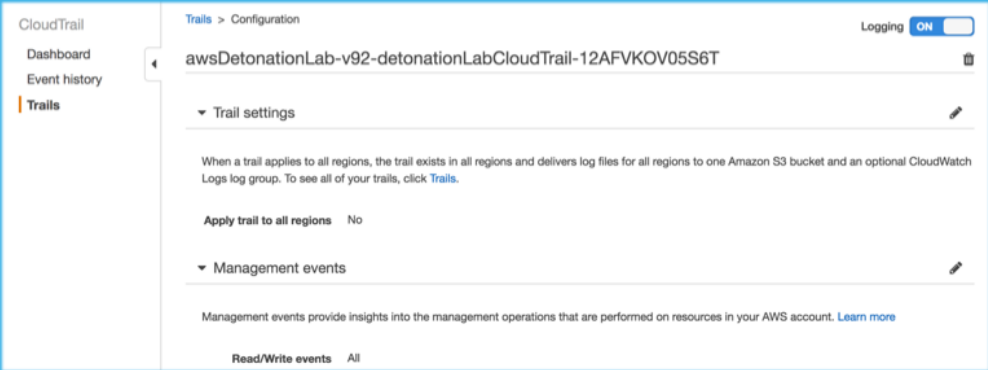
Policy summary {} JSON Edit policy Simulate policy

```
1- {  
2-   "Version": "2012-10-17",  
3-   "Statement": [  
4-     {  
5-       "Sid": "VisualEditor0",  
6-       "Effect": "Allow",  
7-       "Action": [  
8-         "s3:GetObject",  
9-         "s3:ListBucket",  
10-        "s3:DeleteObject"  
11-       ],  
12-       "Resource": [  
13-         "  
14-       ]  
15-     }  
16-   ]  
17- }
```

AWS Identity and Access Management (IAM) enables you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.

IAM controls the access to all amazon resources for the lab without giving any unneeded privileges

# CloudTrail



CloudTrail

Trails > Configuration

Logging ☒

awsDetonationLab-v92-detonationLabCloudTrail-12AFVKOV05S6T

▼ Trail settings

When a trail applies to all regions, the trail exists in all regions and delivers log files for all regions to one Amazon S3 bucket and an optional CloudWatch Logs log group. To see all of your trails, click [Trails](#).

Apply trail to all regions No

▼ Management events

Management events provide insights into the management operations that are performed on resources in your AWS account. [Learn more](#)

Read/Write events All

AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account. With CloudTrail, you can log, continuously monitor, and retain account activity related to actions across your AWS infrastructure.

Cloud trail logs all activity to the aws API. Every action take in the console or from the aws cli generates an API action that cloudtrail will monitor.



# Macie

DASHBOARD

ALERTS

USERS

RESEARCH

SETTINGS

INTEGRATIONS

Categories

All (12)

Basic Alert (12)

Predictive (0)

Anonymized Access (0)

Config Compliance (0)

Credential Loss (0)

Data Compliance (0)

File Hosting (0)

Identity Enumeration (0)

Information Loss (0)

Location Anomaly (0)

Open Permissions (0)

Privilege Escalation (0)

Ransomware (0)

Service Disruption (0)

Suspicious Access (1)

Amazon Macie

assumed-role/molette/rmolet... US East (N. Virginia)

Active (12) Archived (0) All (12)

Amazon Macie is monitoring 0 new S3 objects since the last alert generated 12 days ago. [Learn more](#)

INFO

User or role Access Denied while attempting to List S3 buckets from non-AWS IP

SUSPICIOUS ACCESS BASIC ALERT

12 days ago 1 Results 0 Views

963894186934us-east-1

LOW

Large quantity of S3 buckets deleted

INFORMATION LOSS BASIC ALERT

14 days ago 94 Results 0 Views

963894186934us-east-1

LOW

Change to Cloudtrail logging policy

CONFIG COMPLIANCE BASIC ALERT

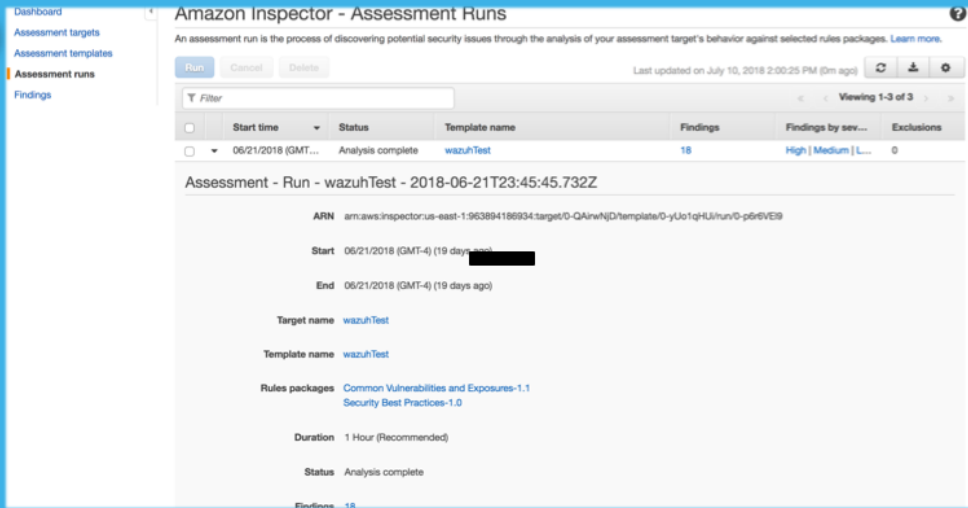
14 days ago 1 Results 0 Views

963894186934us-east-1

Amazon Macie is a security service that uses machine learning to automatically discover, classify, and protect sensitive data in AWS.

I use macie to monitor the data in my s3 buckets

# Inspector



The screenshot displays the Amazon Inspector console interface. On the left, a navigation menu includes 'Dashboard', 'Assessment targets', 'Assessment templates', 'Assessment runs', and 'Findings'. The main panel is titled 'Amazon Inspector - Assessment Runs' and includes a sub-header explaining that an assessment run is the process of discovering potential security issues. Below this, there are buttons for 'Run', 'Cancel', and 'Delete', along with a timestamp 'Last updated on July 10, 2018 2:00:25 PM (5m ago)'. A table lists assessment runs with columns for 'Start time', 'Status', 'Template name', 'Findings', 'Findings by sev...', and 'Exclusions'. One run is visible: 'Assessment - Run - wazuhTest - 2018-06-21T23:45:732Z'. Below the table, detailed information for this run is shown, including its ARN, start and end times, target name ('wazuhTest'), template name ('wazuhTest'), rules packages ('Common Vulnerabilities and Exposures-1.1' and 'Security Best Practices-1.0'), duration ('1 Hour (Recommended)'), and status ('Analysis complete').

Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS.

I use inspector as a vulnerability and compliance scanner for my lab

# CloudWatch Event Rules and Logs

The screenshot displays the AWS CloudWatch console interface. On the left, a navigation menu includes links to Dashboards, Alarms, Billing, Events, Rules, Event Buses, Logs, Metrics, and Favorites. The main content area is divided into two sections: 'Rules' and 'Log Groups'.

**Rules Section:** This section shows a list of event rules. The 'Create rule' button is visible at the top. Below it, a table lists the rules with columns for Status, Name, and Description.

Status	Name	Description
OK	GuardDuty-Alerts	record guardDuty alerts and send to firehose
OK	IAM-Alerts	send IAM alerts to firehose
OK	Macie-Alerts	this is to collect all Macie Alerts
OK	Inspector-Alerts	this will send inspector alerts to firehose

**Log Groups Section:** This section shows a list of log groups. The 'Create Metric Filter' button is visible at the top. Below it, a table lists the log groups with columns for Log Groups, Expire Events After, Metric Filters, and Subscriptions.

Log Groups	Expire Events After	Metric Filters	Subscriptions
/aws/kinesisfirehose/GuardDuty-Alerts	Never Expire	0 filters	None
/aws/kinesisfirehose/IAM-Alerts	Never Expire	0 filters	None
/aws/kinesisfirehose/Macie-Alerts	Never Expire	0 filters	None
/aws/kinesisfirehose/Inspector-Alerts	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v72-FlowLogUpload-19627L8G2ZFC	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v92-FlowLogUpload-1D4C34OU66MZ	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v92-FlowLogUpload-UHWS8FOORQKJ	Never Expire	0 filters	None
awsDetonationLab-v92-BastionMainLogGroup-6WHTFNE7ZQ1L	Never Expire	1 filter	None
awsDetonationLab-v92-FlowLogs-6QZWD4W93YAV	1 day	0 filters	Kinesis (awsDetonationLab-v92-LogStream-1E3J5K1R0PRTQ)
detonationLab-linux	Never Expire	0 filters	None
detonationLab-windows	Never Expire	0 filters	None

Amazon CloudWatch is a monitoring and management service built for developers, system operators, site reliability engineers (SRE), and IT managers.

I am using cloudwatch to watch for service events that can be forwarded to a firehose to be written to s3. I also use cloudwatch logs to hold my vpcflow data

# GuardDuty

GuardDuty
Findings
Settings
Lists
Accounts
What's New
Usage
Partners

## Findings

Showing 181 of 181

Actions

Saved filters / Auto-archive

No saved filters

Add filter criteria

	Finding type	Resource	Li
<input type="checkbox"/>	Recon:EC2/PortProb...	Instance: i-05aa12a	31 ... 62
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-05aa12a	8 h... 1
<input type="checkbox"/>	Recon:EC2/PortProb...	Instance: i-04dc3dc	14 ... 987
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-08215d7	21 ... 1
<input type="checkbox"/>	CryptoCurrency:EC2...	Instance: i-0648216	21 ... 5
<input type="checkbox"/>	Backdoor:EC2/C&C...	Instance: i-0648216	21 ... 2
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-0648216	21 ... 2
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-0648216	21 ... 1
<input type="checkbox"/>	Recon:EC2/Portscan	Instance: i-0648216	21 ... 1
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-04671c8	21 ... 1
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-04dc3dc	3 d... 2
<input type="checkbox"/>	UnauthorizedAccess:...	Instance: i-04dc3dc	3 d... 31

Useful?

Close

### CryptoCurrency:EC2/BitcoinTool.BIDNS

Finding ID: 26b242d38f22f514179531eb5b4383dd

EC2 instance i-0648216eb9976da1 is querying a domain name that is associated with Bitcoin-related activity.

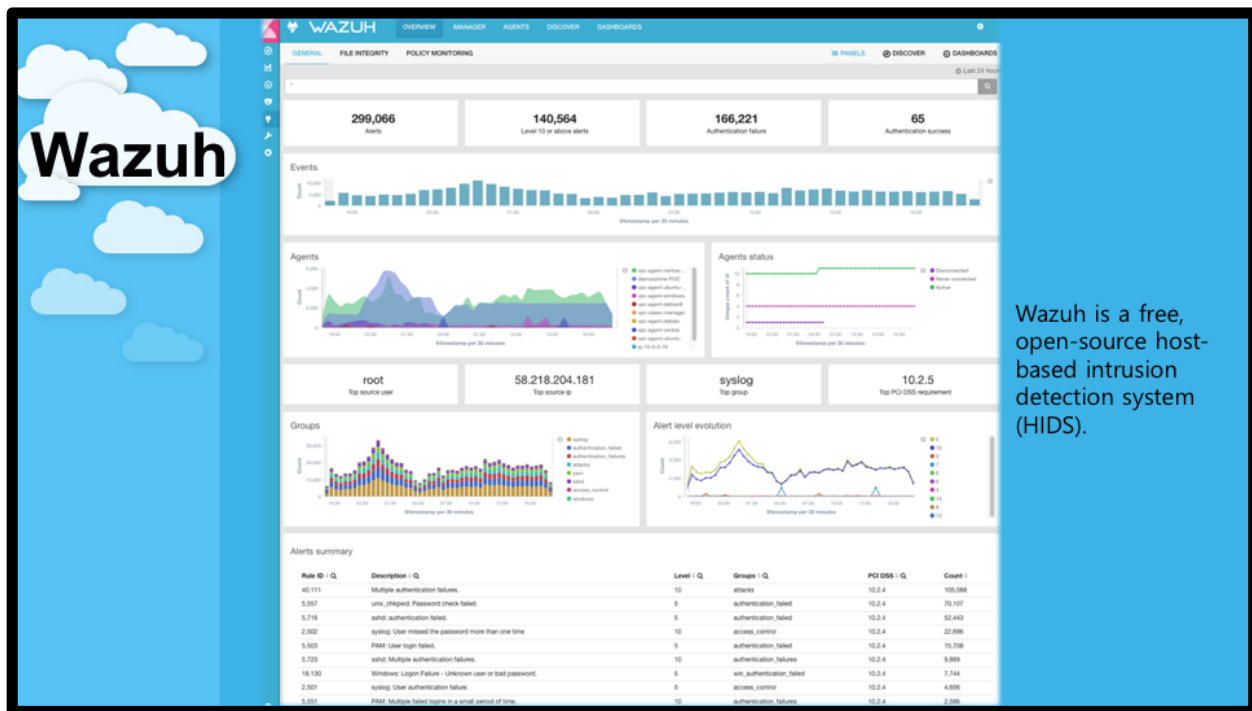
Severity	Region	Count
Medium	us-east-1	5
Account ID	Resource ID	Threat list name
	i-0648216eb9976da1	ProofPoint
Created at	Updated at	
07-10-2018 15:24:39 (20 hour...	07-10-2018 15:40:08 (19 hour...	

Resource affected

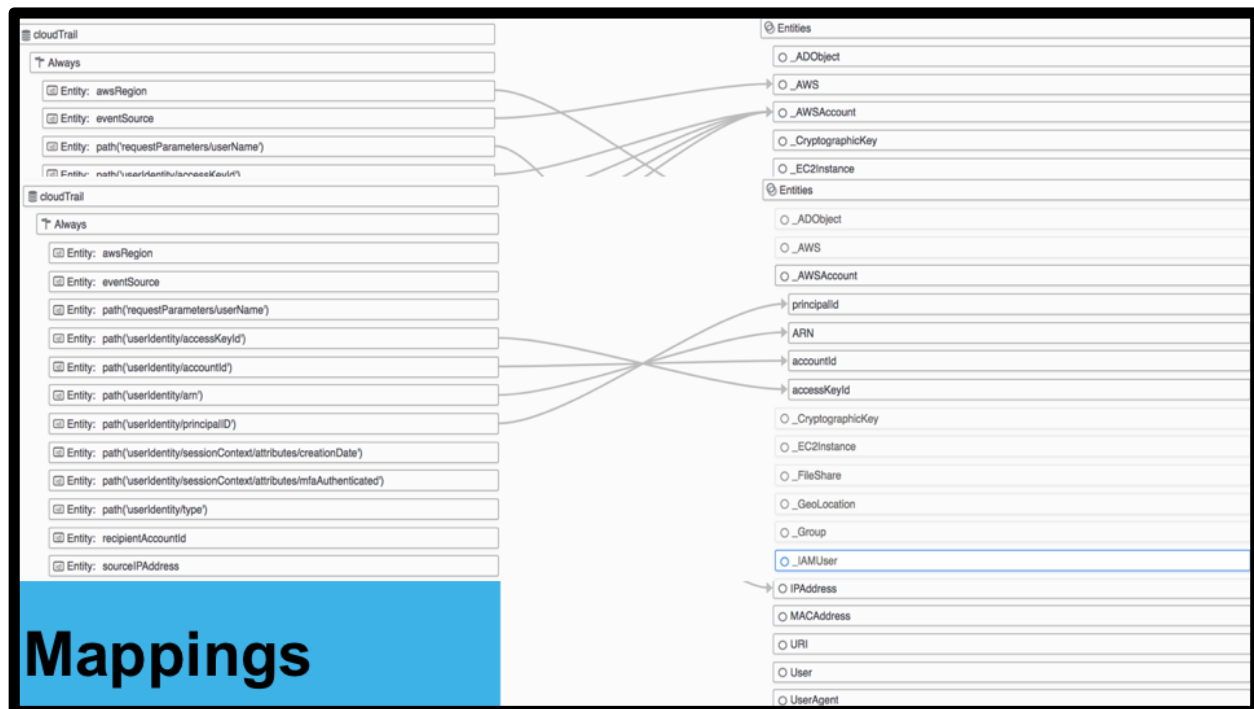
Resource role	Resource type
TARGET	Instance
Instance ID	Instance type
i-0648216eb9976da1	m4.large
Instance state	Availability zone
running	us-east-1a
Image ID	Image description
ami-428aa838	Amazon Linux 2 LTS Candidate AMI 2017.12.0.2...
Launch time	
06-27-2018 14:54:17	
Instance profile	
arn:aws:iam::[redacted]:instance-profile/cloudwatch-writeLogs	
ID: AIPATQ44ZZE64MH83ASZ	
Tags	

Amazon GuardDuty is a threat detection service that continuously monitors for malicious or unauthorized behavior to help you protect your AWS accounts and workloads.

I use Guadduty to alert me on unusual and malicious behaviors like bruteforce attempts, scanning, and credential abuse.



I used wazuh as my HIDS for the lab. It is based on OSSEC but has a large number of additional features. I'd like to give a big shoutout to the wazuh team and my contact Marta for working with me to improve the integration of aws service logs into the product. What a great example of the opensource community.



Now that we have all the data getting ingested we need to figure out the best way to correlate the data and start using it to detect and investigate activity

Here I have graphically mapped the datasource cloudtrail to entities and features in a model



# Simple Dashboards

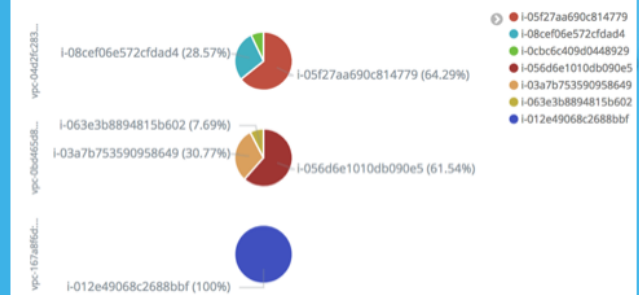
VPCFlow-Top10ExternalDestinationIP-Table

Top 10 External Destination IP	Count
54.239.31.225	4,406
54.239.25.71	4,282
54.239.25.60	4,150
54.239.30.177	2,478
54.239.30.195	2,465
45.127.112.2	1,709
54.239.29.61	1,469
66.241.101.63	477
209.141.60.238	291

GuardDuty-MostCommonAccountID-Table

Account ID	Count
9638 [REDACTED]	40

GuardDuty-BreakdownOfAlertsPerInstancePerVPC-Pie



CloudTrail-EventNames-pie



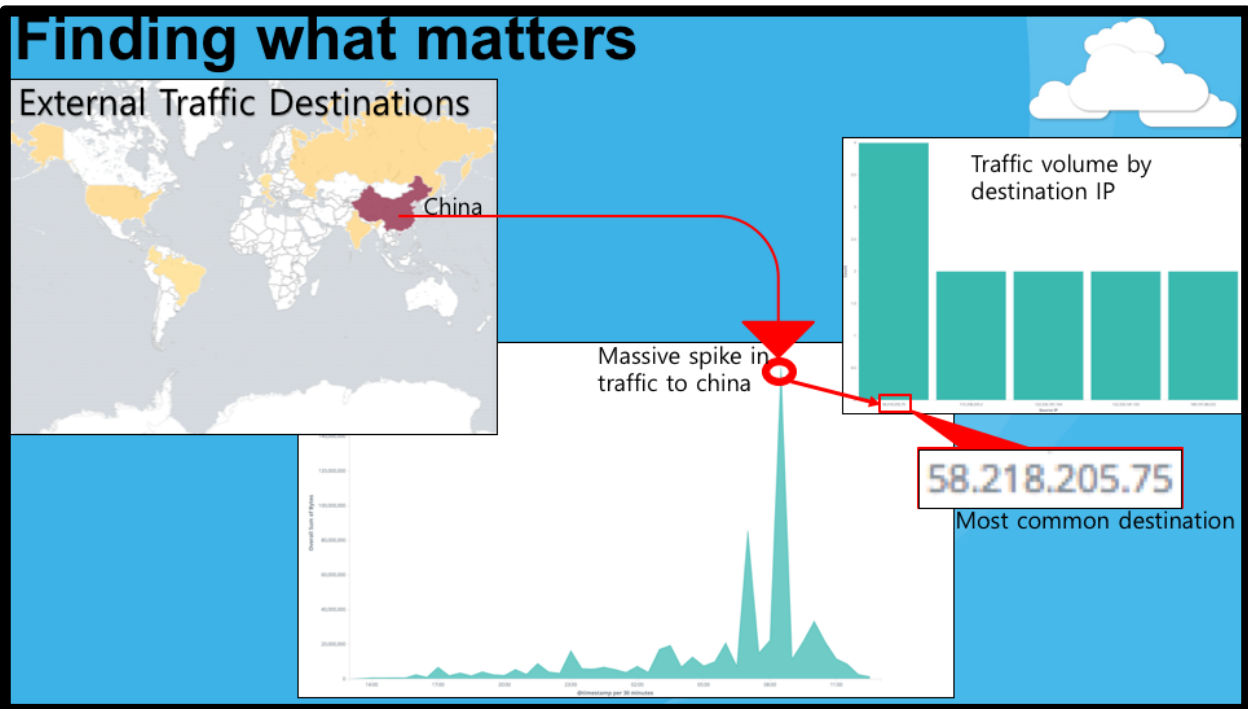
Let's get started with a few simple dashboards. All of these dashboards and more are included in the detonationlab Kibana instance for usage.

Top destination address.

Top talkers by interface.

Most common account ID in guardduty alerts.

Most common cloutrail events



Using some more advanced dashboards have a world heat map for vpc traffic. I noticed that china is now a common destination so I check out a histogram showing volume of traffic and notice that the traffic has suddenly spiked. Pivoting into destination IPs I noticed that a single destination is responsible for the majority of the traffic spike.



From this IP address I check my logs to see if there are any other events involving this destination and I find multiple guardduty alerts for traffic to known C&C server, an ssh brute force attack against one of my instances, and a few other attacks. Let's look at my involved instance for the culprit.

# Finding what matters

Most common destination

58.218.205.75

File added to the system.

File responsible for outbound connection

New file '/bin/pip3.7' added to the file system.

VT Results

53 engines detected this file

Here I can see a record of the netstat table being changed to involve the known destination IP on that host. It looks like the binary pip3.7 is responsible for the traffic. I do a quick lookup on the hash for the binary on virustotal and find this this binary is actually a known crypto miner.

I've now done a full stack analysis of a suspicious network event all the way down to the binary responsible for the traffic. What next? Well how did they get in in the first place?

I can go back through all new file alerts and find the user and binary responsible for creating this one. I find that apache was the user writing the file to disk and can assume I have a vulnerable webserver. But how?

# How did they get in?

Severity	Date	Finding	Target	Template	Rules Package
High	Yesterday at...	Instance i-063e3b8894815b602 is vulnerable to C...	everything	everything	Common Vulnerabilities and Exposures-1.1

Finding for assessment target 'everything' and template 'everything'

ARN: `arn:aws:inspector:us-east-1:963894186934:target/0-ITOUJioL/template/0-S7AJH5Cj/huv/0-JGN1eNRF/finding/0-jkF0xbLT`

Run name: `5f44c7b-3421-133f-1833-5ef4328e05a6_885e45c5-3f8a-9b6d-d70f-194994c58260`

Target name: `everything`

Template name: `everything`

Start: Yesterday at 10:55 PM (GMT-4) (12 hours ago)

End: Yesterday at 11:56 PM (GMT-4) (11 hours ago)

Status: Analysis complete

Rules package: Common Vulnerabilities and Exposures-1.1

AWS agent ID: `i-063e3b8894815b602`

Finding: Instance i-063e3b8894815b602 is vulnerable to CVE-2018-10897

Severity: High

Description: A directory traversal issue was found in reposync, a part of yum-utils, where reposync fails to sanitize paths in remote repository configuration files. If an attacker controls a repository, they may be able to copy files outside of the destination directory on the targeted system via path traversal. If reposync is running with heightened privileges on a targeted system, this flaw could potentially result in system compromise via the overwriting of critical system files. Version 1.1.31 and older are believed to be affected.

Recommendation: Use your Operating System's update feature to update package yum-plugin-priorities-0:1.1.31-45.amzn2.0.1, yum-utils-0:1.1.31-45.amzn2.0.1. For more information see <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-10897>

Show Details

Using inspector I am able to scan my instance to for common vulnerabilities and found that my server was vulnerable to a known CVE that allows for files to be written to disk as the webserver user

# Recap

## Visibility

## Accountability

Traditional Tool	AWS equivalent
IDS/IPS	guardDuty
DLP	Macie
EDR	Cloudwatch + osquery, GRR
Netflow	Cloudwatch + VPCFlow
DNS	Cloudwatch + Route53
Access and authentication auditing	CloudTrail
Active Directory	Directory Service
Identity Management	IAM
Single Sign On	AWS SSO
Vulnerability scanner	Inspector
Configuration Management	AWS config
Logging	Cloudwatch + Firehose + Lambda

For a quick recap, I am able to view activity in my cloud environment very similarly to how I would in my on prem environment. I use these cloud native tools to increase visibility until I am able to find accountability for all actions. Using just tools available from AWS and opensource I can monitor my environment effectively



Last but not least, let's take a look at squirrely attacker personal lifecycle.

They start working hard but not seeing financial returns they need to live how they want to

Then they discover with a few investments they can start making money easily with cryptominers

Next they use those ill gotten gains to buy the only lambo in easter Europe

Which leads to them getting arrested for tax evasion

Which leads to them flipping to act as a witness and end up in the witness protection program

Only to start the cycle again



Thank you all very much for your time today.

And as always remember my motto, flag it, tag it, and bag it.