



Who done it:

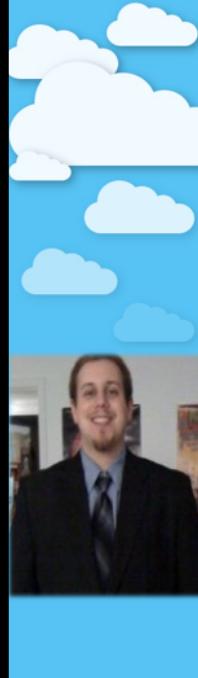
Gaining visibility and
accountability in the cloud

By Ryan Nolette



Squirrel Edition

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 clouformation

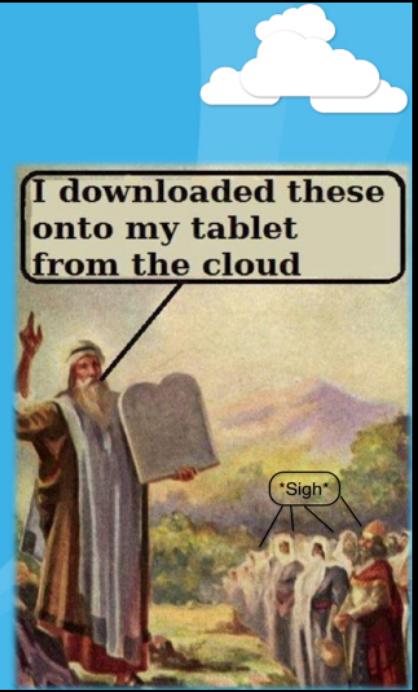
– <https://github.com/sonofagl1tch/AWSDetonationLab>

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

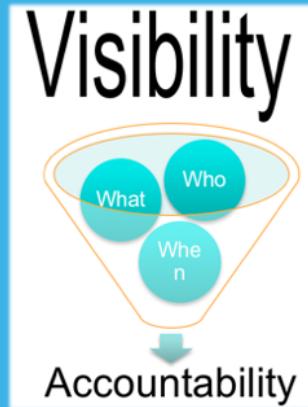
- Overview
 - Who did what and when?
 - Common Techniques
 - Common Visibility Tools and Their AWS Equivalent
- Increasing visibility until you have accountability
 - Common Logging
 - Authentication
 - Endpoint
 - Network
 - Vulnerabilities
 - Configuration
- End to end example
 - Detonation Lab
 -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



As part of any investigation, I strive to answer 3 questions, who did what and when. This is a 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?



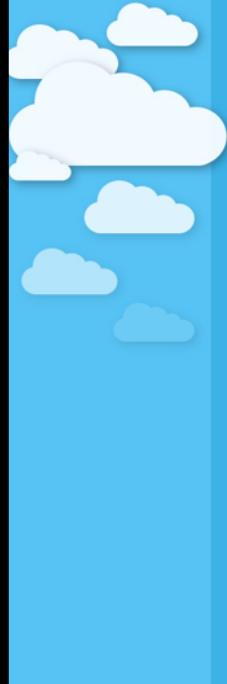
- OS hardening
- Config management
- Identity Management
- Process monitoring

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

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

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.

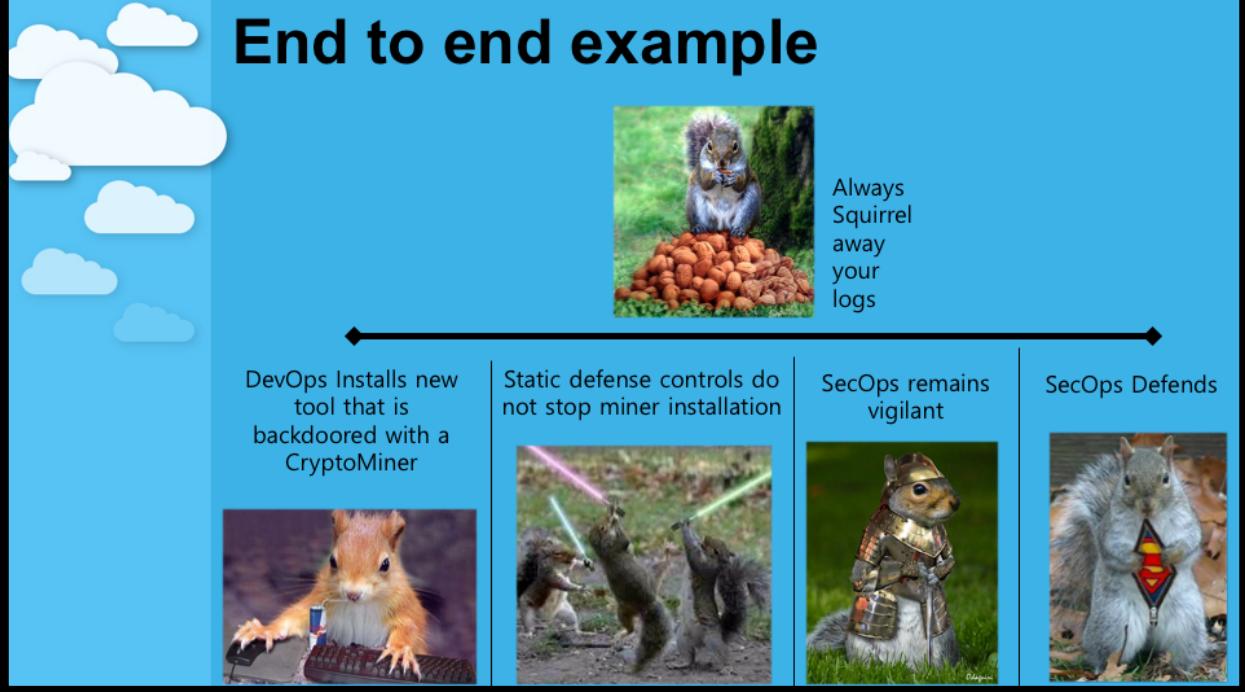
<u>OS hardening</u>	<u>Logging</u>	<u>Authentication</u>	<u>Endpoint</u>	<u>Network</u>	<u>Vulnerabilities/Configuration</u>
<ul style="list-style-type: none">CIS guidelines audit and hardening scripts.Additional logging and hardening scripts created by experience over time.	<ul style="list-style-type: none">Common logging like auth logs and process creation etc	<ul style="list-style-type: none">/var/log/secureIAM logsIAM rolesIAM policies	<ul style="list-style-type: none">EDRHIDSCloudwatchGuardDuty	<ul style="list-style-type: none">IDSNetstatTcpdumpVpc flow logsDns route 53 logs	<ul style="list-style-type: none">Generic vuln scannerInspectorNVD/CVE usageAws configOS hardeningApplications config



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 😊

End to end example



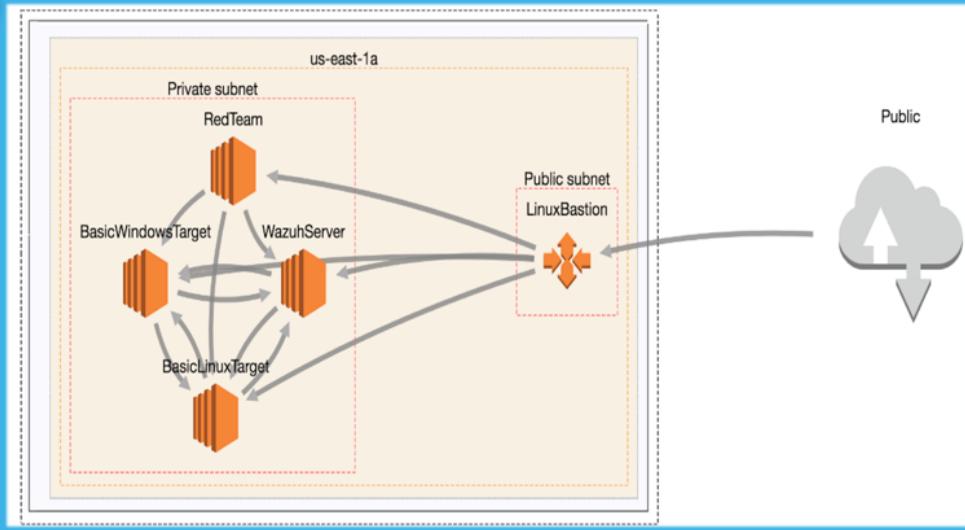
Now let's talk about some practical application of what I have said so far.

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.

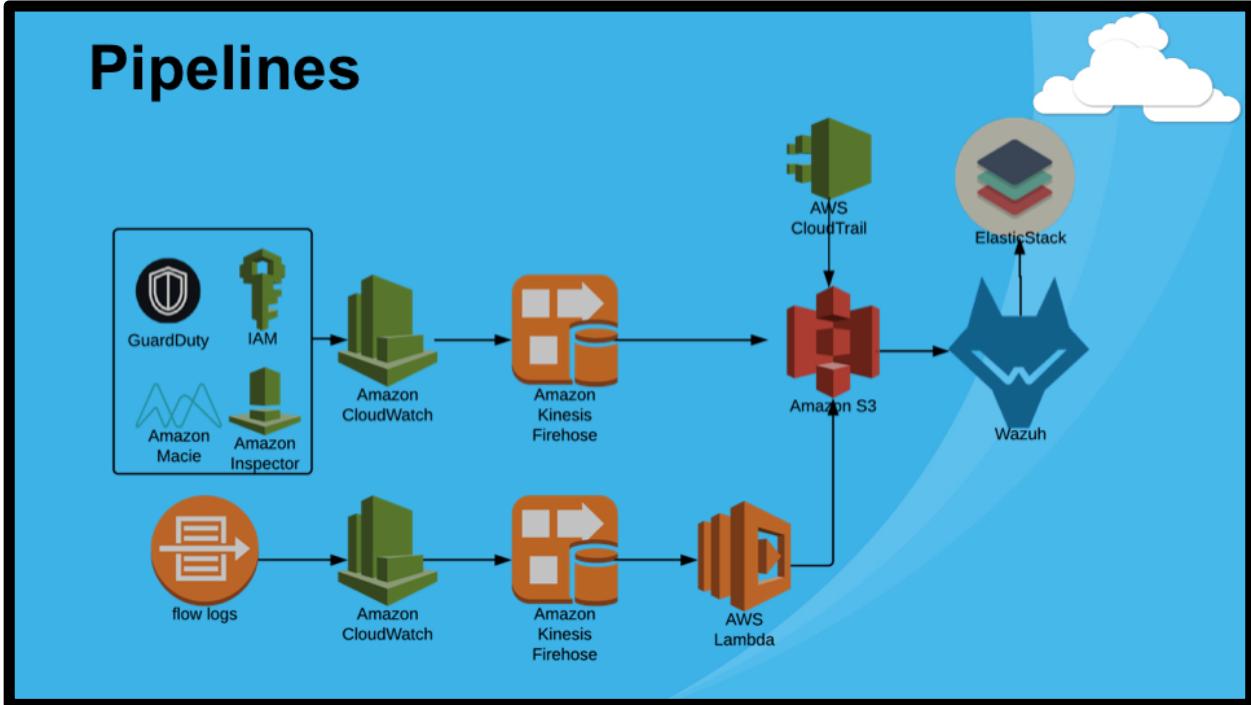
Detonation Lab Topology



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.

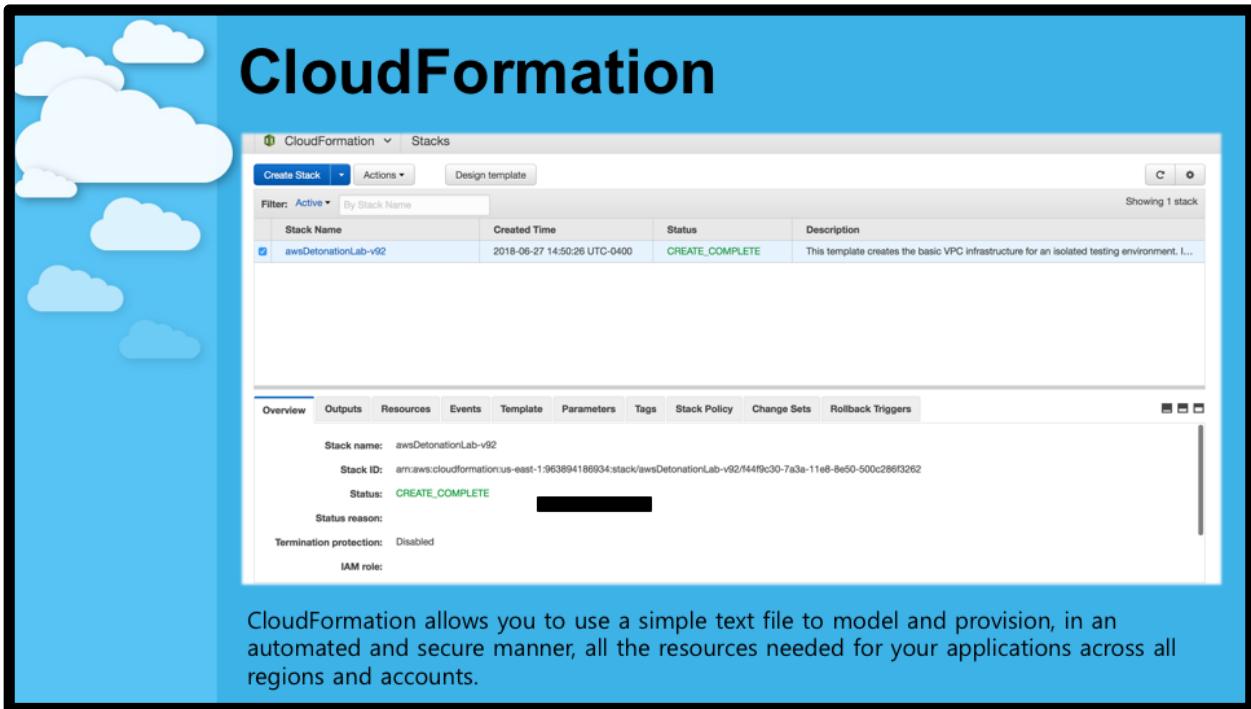
Pipelines



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 pipelines 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.



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.

The screenshot shows the Amazon S3 console interface. The top navigation bar has a search bar labeled "Search for buckets", a "Create bucket" button, and three other buttons: "Delete bucket", "Empty bucket", "28 Buckets", "0 Public", and "1 Regions". Below the navigation is a table listing six buckets:

Bucket Name	ACL	Region	Last Modified
awsdetonationlab-v92-s3bucketcloudtrail-v9d4yqs8ytsh	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketguardduty-jkremr5l7jb	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketiam-1y00gbui8vi82	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-1tv3blodhx6fl	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:33 PM GMT-0400
awsdetonationlab-v92-s3bucketvpcflow-167orj11dq1	Not public *	US East (N. Virginia)	Jun 27, 2018 2:50:32 PM GMT-0400

Below the table, a large blue area contains the text: "Amazon S3 is object storage built to store and retrieve any amount of data from anywhere".

This is my storage.

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with various navigation options: Your VPCs, Subnets, Route Tables, Internet Gateways, Egress Only Internet Gateways, DHCP Options Sets, Elastic IPs, Endpoints, Endpoint Services, NAT Gateways, and Peering Connections. The main area displays a table of VPCs. A single row is selected, showing details for 'awsDetonationLab-v92'. The table columns include Name, VPC ID, State, IPv4 CIDR, IPv6 CIDR, DHCP options set, Route table, and Network ACL. Below the table, there are tabs for Summary, CIDR Blocks, Flow Logs, and Tags. The Summary tab is active, showing the VPC ID (vpc-734da209), Name (awsDetonationLab-v92), State (available), IPv4 CIDR (172.16.0.0/27), IPv6 CIDR (not specified), DHCP options set (dopt-7016980b), and Route table (rtb-d91eb8a6). It also shows the Network ACL (acl-167f356c) and tenancy settings (Default, yes, 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.

The screenshot shows the Amazon Kinesis Data Streams console. The main title is "VPC Flow - Kinesis Stream". On the left, there's a sidebar with "Amazon Kinesis" at the top, followed by "Dashboard", "Data Streams" (which is selected and highlighted in orange), "Data Firehose", "Data Analytics", and "Video Streams". Below that are "External resources" and "What's new". The main content area is titled "Kinesis streams" and contains a brief 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)." There's a "Learn more" link. Below this, it says "Total shards in use: 10 Total shards remaining: 490". A "Create Kinesis stream" button is available. A search bar allows filtering by "stream name, number of shards, and status". A table lists the existing stream: "awsDetonationLab-v92-LogStream-1E3JSK1ROPRQRT" with 10 shards and an "Active" status. The table has columns for "Kinesis stream name", "Number of shards", and "Status".

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.

The screenshot shows the AWS Lambda Functions dashboard. The left sidebar has a blue background with white clouds and displays 'Dashboard' and 'Functions'. The main area has a light blue background with the word 'Lambda' in large black letters. The title 'Lambda' is at the top. Below it is a search bar with placeholder text 'Filter by tags and attributes or search by keyword'. A table lists three functions:

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-1D4C340UI66MZ		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.

The screenshot shows the Amazon Kinesis Data Firehose console. The left sidebar has sections for Dashboard, Data Streams (with Data Firehose selected), Data Analytics, and Video Streams. Below that is External resources and What's new. The main area title is "Firehose". A sub-header says "Kinesis Firehose delivery streams continuously collect, transform, and load streaming data into the destinations that you specify." It includes a "Create delivery stream" button, a "Test with demo data" button, and a "Delete" button. There is a search bar labeled "Filter or search by name". A table lists five delivery streams:

Name	Status	Created	Source	Record transformation	Destination
awsDetonationLab-v92-FirehosedeliverystreamGuardDumbANGKIB81L550	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 awsdetonationlab-v92-s3bucketguardduty-jkremr5f7jbj
awsDetonationLab-v92-FirehosedeliverystreamIAM-1M0ZMRPN00530	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 awsdetonationlab-v92-s3bucketiam-1y00gbu@v182
awsDetonationLab-v92-FirehosedeliverystreamInspect-SKFM22924VLJ	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 awsdetonationlab-v92-s3bucketinspector-1xz05by24ypuk
awsDetonationLab-v92-FirehosedeliverystreamMacie-3ROVXTDXE019	Active	2018-06-27T14:51-0400	Direct PUT and...	Disabled	Amazon S3 awsdetonationlab-v92-s3bucketmacie-1hv3blodhx8fl

Below the table, a note states: "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

The screenshot shows the AWS IAM console interface. On the left, there's a sidebar with navigation links: Dashboard, Groups, **Users**, Roles, Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main area is titled 'IAM' and shows a user named 'wazuh-user'. The user details include:

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

The 'Permissions' tab is selected, showing 'Attached policies: 1'. The attached policy is 'wazuh-read-cloudTrail', which is a Managed policy. Below this, there's a 'Policy summary' section with a 'JSON' button and an 'Edit policy' button. The JSON code for the policy is displayed:

```
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-         "arn:aws:s3:::wazuh-*/*"
14-       ]
15-     }
16-   ]
17- }
```

A descriptive text at the bottom of the page states: "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

The screenshot shows the AWS CloudTrail configuration interface. On the left, there's a sidebar with 'CloudTrail' at the top, followed by 'Dashboard', 'Event history', and 'Trails' (which is highlighted). The main content area has a header 'Trails > Configuration' and a trail name 'awsDetonationLab-v92-detonationLabCloudTrail-12AFVKOV05S6T'. A 'Logging' switch is set to 'ON'. Below the trail name, there are sections for 'Trail settings' (with a note about logs being delivered to S3 and CloudWatch Logs), 'Management events' (with a note about providing insights into management operations), and 'Read/Write events' (set to 'All'). At the bottom of the configuration page, there's a descriptive text block:

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.

The screenshot shows the Amazon Macie interface. On the left is a sidebar with icons for Dashboard, Alerts, Users, Research, Settings, and Integrations. The main area has a blue header with the Macie logo. Below the header is a search bar and a filter section for Active (12), Archived (0), and All (12) alerts. A message states: "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

LOW

Large quantity of 53 buckets deleted

INFORMATION LOSS BASIC ALERT

14 days ago 94 Results 0 Views

LOW

Change to Cloudtrail logging policy

CONFIG COMPLIANCE BASIC ALERT

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

The screenshot shows the Amazon Inspector console with a blue header and sidebar featuring white clouds. The sidebar has navigation links: Dashboard, Assessment targets, Assessment templates, Assessment runs (which is selected and highlighted in orange), and Findings.

The main content area is titled "Amazon Inspector - Assessment Runs". It displays a table of assessment runs. The table has columns: Start time, Status, Template name, Findings, Findings by sev..., and Exclusions. One row is visible:

Start time	Status	Template name	Findings	Findings by sev...	Exclusions
06/21/2018 (GMT...)	Analysis complete	wazuhTest	18	High Medium Low	0

Below the table, a detailed view of the "wazuhTest" assessment run is shown:

- ARN: arn:aws:inspector:us-east-1:963894186934:target/0-QA/rwNjD/template/0-yUo1qjH/run/0-p6t6VEI9
- Start: 06/21/2018 (GMT-4) (19 days ago)
- End: 06/21/2018 (GMT-4) (19 days ago)
- Target name: wazuhTest
- Template name: wazuhTest
- Rules packages:
 - Common Vulnerabilities and Exposures-1.1
 - Security Best Practices-1.0
- Duration: 1 Hour (Recommended)
- Status: Analysis complete
- Findings: 18

Text at the bottom of the screenshot states: "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

Rules

Rules route events from your AWS resources for processing by selected targets. You can create, edit, and delete rules.

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

CloudWatch Logs

Filter: Log Group Name Prefix:

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-196Z7ILBGZ2ZFC	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v92-FlowLogUpload-1D4C34OLU66MZ	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v92-FlowLogUpload-UHW58P0CRQKJ	Never Expire	0 filters	None
/aws/lambda/awsDetonationLab-v92-BastionHostLogGroup-SWHTFNE7ZQ1L	Never Expire	1 filter	None
awsDetonationLab-v92-FlowLog-6Q2WD4W93YAV	1 day	0 filters	Kinesis (awsDetonationLab-v92-LogStream-1E3JSK1R0PRQ1)
detonationLab-linux	Never Expire	0 filters	None
detonationLab-windows	Never Expire	0 filters	None

Viewing 1 to 5 of 5 Rules

Log Groups 1-11

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

The screenshot shows the Amazon GuardDuty console. On the left, a sidebar menu includes 'Findings' (selected), 'Settings', 'Lists', 'Accounts', 'What's New', 'Usage', and 'Partners'. The main area is titled 'Findings' with a subtitle 'Showing 181 of 181'. It displays a table of findings with columns: Finding type, Resource, and Count. A specific finding is selected, showing detailed information on the right. The finding ID is 26b242d38f22f514179531eb5b4383dd. The finding details include:

Severity	Region	Count
Medium	us-east-1	5
Account ID	I-06482168eb9976da1	Threat list name
Created at	07-10-2018 15:24:39 (20 hours...)	ProofPoint
Updated at	07-10-2018 15:40:08 (19 hours...)	

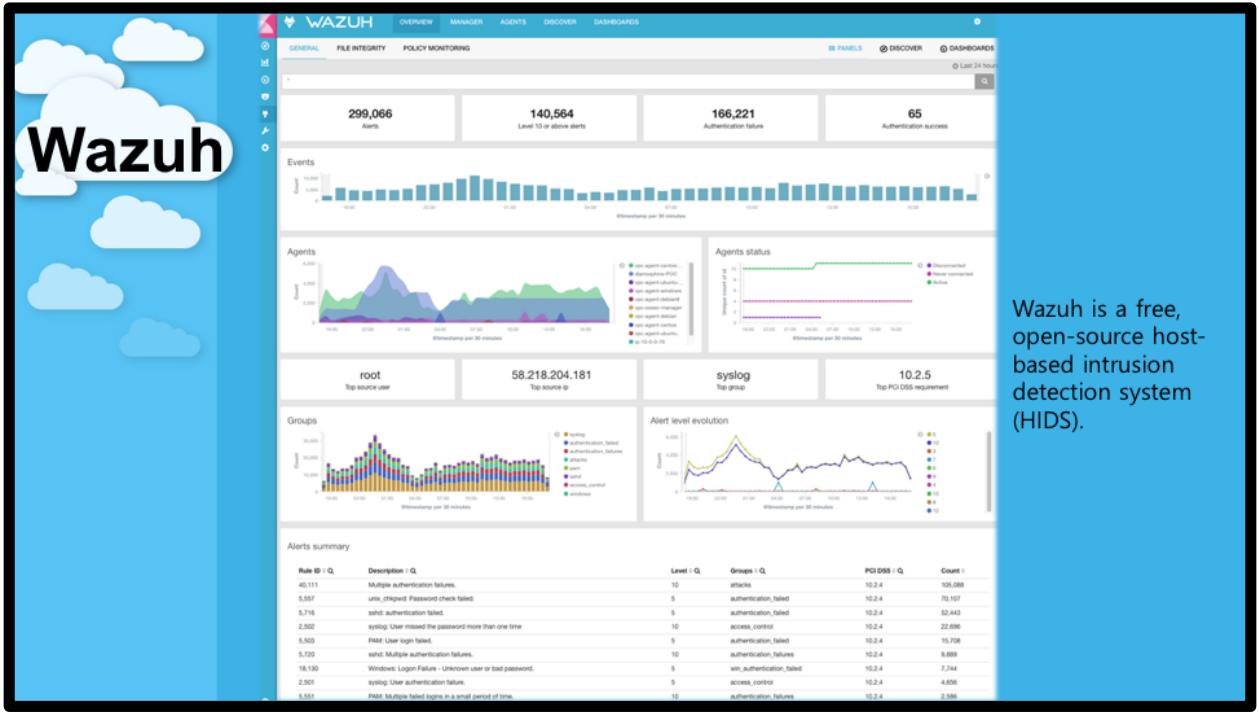
Resource affected

Resource role	Resource type
TARGET	Instance
Instance ID	I-06482168eb9976da1
Instance state	running
Image ID	ami-42baa838
Launch time	06-27-2018 14:54:17
Instance profile	Arn: arn:aws:iam:█████████████████████:instance-profile/cloudwatch-writeLogs
ID	AIPN7044ZZZB4MHR3AS2

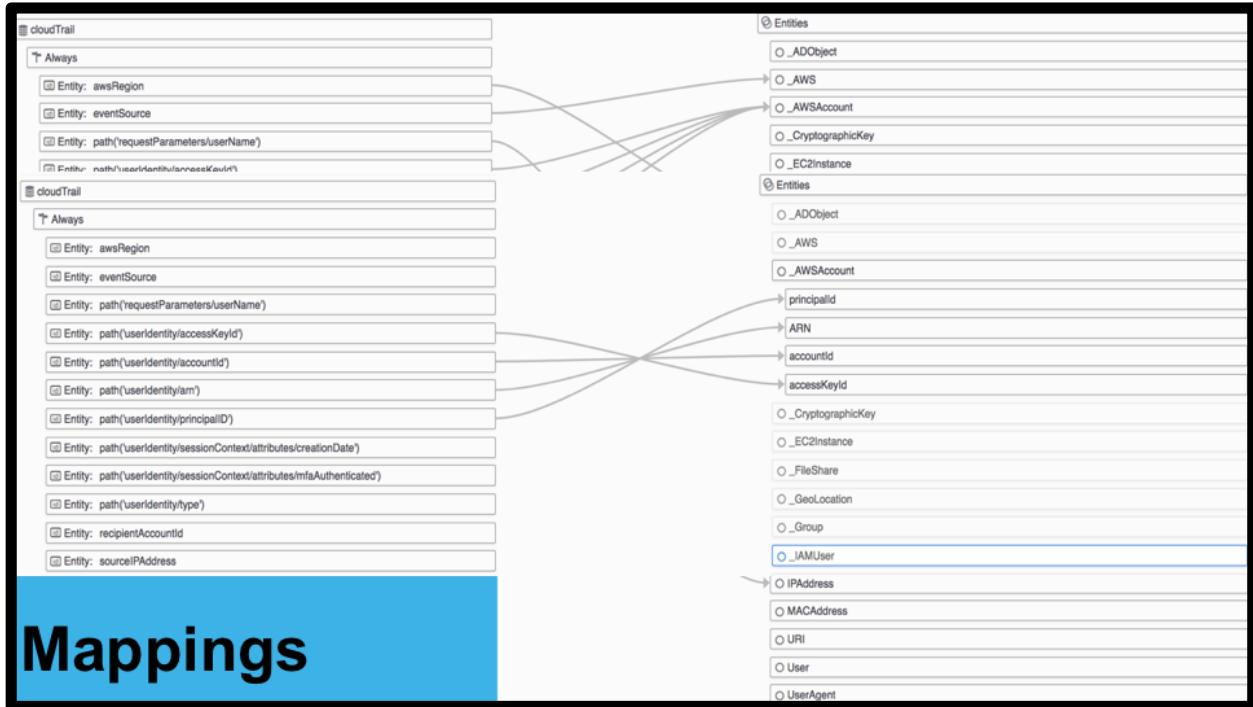
Tags

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
[REDACTED]	1



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

Finding what matters



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.

Finding what matters

58.218.205.75
Most common destination

loggingSourcesFor-58.218.205.75
AWS Logging Sources: vpc, guardduty

Logs with Destination

Selected Fields: t_data.aws.event.title, @timestamp, t_id, t_index, _score, t_type, t_agentId, t_agent.name

Time: July 24th 2018, 15:50:02.103 - August 23rd 2018, 15:50:02

Time	data.aws.event.title	rule.groups	rule.description	preddecoder.hostname
August 21st 2018, 14:25:34.000	i-056d6e1810db090e5 is performing RDP brute force attacks against 172.16.0.23.	amazon	Guard Duty Finding with a high level: i-056d6e1810db090e5 is performing RDP brute force attacks against 172.16.0.23. Brute force attacks are used to gain unauthorized access to your instance by guessing the RDP password.	ip-172-16-0-21
August 21st 2018, 14:25:34.589	Bitcoin-related domain name queried by EC2 instance i-056d6e1810db090e5.	amazon	Guard Duty Finding with a medium level: EC2 instance i-056d6e1810db090e5 is querying a domain name that is associated with Bitcoin-related activity.	ip-172-16-0-21
August 21st 2018, 14:25:34.589	Command and Control server domain name queried by EC2 instance i-056d6e1810db090e5.	amazon	Guard Duty Finding with a high level: EC2 instance i-056d6e1810db090e5 is querying a domain name that is associated with a known Command & Control server.	ip-172-16-0-21
August 21st 2018, 11:07:47.547	Bitcoin-related domain name queried by EC2 instance i-056d6e1810db090e5.	amazon	Guard Duty Finding with a medium level: EC2 instance i-056d6e1810db090e5 is querying a domain name that is associated with Bitcoin-related activity.	ip-172-16-0-21
August 21st 2018, 10:57:48.155	Command and Control server domain name queried by EC2 instance i-056d6e1810db090e5.	amazon	Guard Duty Finding with a high level: EC2 instance i-056d6e1810db090e5 is querying a domain name with a known Command & Control server.	ip-172-16-0-21
August 21st 2018, 09:27:43.800	i-056d6e1810db090e5 is performing RDP brute force attacks against 172.16.0.23.	amazon	Guard Duty Finding with a high level: i-056d6e1810db090e5 is performing RDP brute force attacks against 172.16.0.23. Brute force attacks are used to gain unauthorized access to your instance by guessing the RDP password.	ip-172-16-0-21
August 21st 2018, 09:27:43.800	Outbound portscans from EC2 instance i-056d6e1810db090e5.	amazon	Guard Duty Finding with a medium level: EC2 instance i-056d6e1810db090e5 is performing outbound port scans against remote host 172.16.0.22.	ip-172-16-0-21
August 21st 2018, 09:27:43.790	i-056d6e1810db090e5 is performing SSH brute force attacks against 172.16.0.22.	amazon	Guard Duty Finding with a high level: i-056d6e1810db090e5 is performing SSH brute force attacks against 172.16.0.22. Brute force attacks are used to gain unauthorized access to your instance by guessing the SSH password.	ip-172-16-0-21

Ec2 Instance involved

Hostname of findings

All Guardduty Findings for instance

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

The screenshot displays a dashboard for monitoring network traffic and file system changes. On the left, a table shows 'Most common destination' traffic, with the IP **58.218.205.75** highlighted. On the right, a log entry indicates a file was added to the system, and another log entry shows a new file ('/bin/pip3.7') was added to the file system. A red arrow points from the 'File added to the system' log to the 'New file' log. A third red arrow points from the 'New file' log to a 'VT Results' section where it says 53 engines detected the file. A fourth red arrow points from the 'VT Results' section to the 'File added to the system' log.

Timestamp	id	index	score	type	agent_id	agent_name	decoder_name	full_log
August 24th 2018, 16:16:44.349	12e08fb8-powhuL089	wazuh-alerts-3.x-2018.08.24	-	ossec	802	linuxVtctm2		
	agent_ip		172.16.8.75					
	agent_name							
	decoder_name							
	full_log							
	ossec: output: 'netstat outbound connections':							
	Active Internet connections (w/o servers)							
	Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name							
	tcp 0 1 172.16.8.75:34272 58.218.205.75:80 SYN_SENT 6421 pip3.7							
	tcp 53 0 172.16.8.75:52689 54.239.30.195:443 ESTABLISHED 3492/wsgent							
	tcp 0 0 172.16.8.75:22 172.16.8.75:22 ESTABLISHED 2945/sshd: ec2-user							
	udp 0 0 172.16.8.75:59387 172.16.8.75:1514 ESTABLISHED 6597/ossec-agentd							
	previous_output							
	ossec: output: 'netstat outbound connections':							
	Active Internet connections (w/o servers)							
	Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name							
	tcp 0 1 172.16.8.75:34272 58.218.205.75:80 SYN_SENT 6421 pip3.7							
	tcp 53 0 172.16.8.75:52689 54.239.30.195:443 ESTABLISHED 3492/wsgent							
	tcp 0 0 172.16.8.75:22 172.16.8.75:22 ESTABLISHED 2945/sshd: ec2-user							
	udp 0 0 172.16.8.75:59387 172.16.8.75:1514 ESTABLISHED 6597/ossec-agentd							
	rule_description							
	Listened ports status (checkin/rport) changed (new port opened or closed).							
	rule_firerates							
	rule_gdr							
	rule_gpp13							
	rule_groups							
	rule_id							
	rule_level							
	rule_mail							
	rule_pcl_dss							

Timestamp	id	index	score	type	agent_id	agent_name	decoder_name	full_log
August 28th 2018, 16:06:59.483	004947syscheck	wazuh-alerts-3.x-2018.08.28	-	syscheck	800	ip-172-16-8-21.ec2.internal		
	location							
	manager_name							
	path							
	rule_decoder_name							
	rule_id							
	rule_level							
	rule_mail							
	rule_pcl_dss							
	syscheck.event							
	syscheck.gid_after							
	syscheck.gid_before							
	syscheck.gname_after							
	syscheck.inode_after							
	syscheck.mds_after							
	syscheck.mine_after							
	syscheck.path							
	syscheck.perm_after							
	syscheck.sha1_after							
	syscheck.size_after							
	syscheck.uid_after							
	syscheck.uname_after							

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?

The screenshot shows the AWS Inspector findings interface. At the top, there's a table with columns: Severity, Date, Finding, Target, Template, and Rules Package. A single row is selected, showing 'High' severity, 'Yesterday at...', 'Instance i-063e3b8894815b602 is vulnerable to CVE-2018-10897', 'everything', 'everything', and 'Common Vulnerabilities and Exposures-1.1'. Below the table, the page title is 'Finding for assessment target 'everything' and template 'everything''. The main content area contains the following details:

- ARN:** arn:aws:inspector:us-east-1:963894186934:target/0-ITQJioL/template/0-S7AH5c/run/0-JGN1eNPF/finding/0-jR0XhsLT
- Run name:** 5f44c7fb-3421-133f-1833-5ef4328e05e6_885e45c5-3f6a-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



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

Flag it, Tag it, and Bag it.



Thank you all very much for your time today.

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