**Number:** 200

**Question:**

Deliverable (Artifact) = Green | Action within an Event (Process) = Blue | Variables (Variable for Analysis) = Orange | Technology (Artifact) = Purple

List out the IAM users that accessed an AWS service (successfully or unsuccessfully) in Frothly's AWS environment? Answer guidance: Comma separated without spaces, in alphabetical order. (Example: ajackson,mjones,tmiller)

**Hint:**

Use aws:cloudtrail as the sourcetype. Look at the values within the user\_type field.

**Answer**:

bstoll,btun,splunk\_access,**[web\_admin]** 🡨 The Who?

**Query:**

index=botsv3 earliest=0 sourcetype=aws:cloudtrail user\_type=IAMUser| top limit=0 userName

**Analysis:**

**Initial Analysis Question?** What is Normal vs. Abnormal access to the AWS environment? Be aware of Human Intelligence (Customer) vs. Systems Intelligence (SIEM [Splunk] Data or Vendor Data [AWS Documentation]) vs. Threat Intelligence (3rd Party Data [IBM X-Force Exchange] or Security Operations Center Data [Reliaquest])

Step 1 🡪 Check Systems Intelligence | Step 2 🡪 Check Threat Intelligence | Step 3 🡪 Verify Human Intelligence 🡺 COLLECTIVELY equal to Cyber Analysis Intelligence

Normal Event vs. Abnormal Event could be analyzed by investigating the errorCode field.

Abnormal Event could be mapped to MITRE ATTACK Framework. This would “Classify” the event.

What other attributes/fields can we use to inform Normal vs. Abnormal Events? These columns could come by way of Technologies (Could be a Hypothesis?)

Two technology candidates in the AWS event data: sourceIPAddress (**where** the connection came from), userAgent (**what** was used to make the connection), eventType (**how** the connection was made), and eventTime (**when** did the connection take place)

GENERALIZED QUESTIONS IN AN ANALYSIS: Who?, What?, Where?, How?, and When?

**Sample output and evidence of significance:**

Summary statistics on **errorCode** field:

**A screenshot of a cell phone

Description automatically generated**

Sample of Client.InstanceLimitExceeded: (43.344% [280] of total count)

A screenshot of a social media post

Description automatically generated

Sample of Client.UnauthorizedOperation: (35.758% [231] of total count)

A screenshot of a social media post

Description automatically generated

Sample of Client.Unsupported: (16.099% [104] of total count)

A screenshot of a social media post

Description automatically generated

Sample of success: (2.632% [17] of total count)

A screenshot of a social media post

Description automatically generated

Sample of AccessDenied (1.084% [7] of total count)

A screenshot of a social media post

Description automatically generated

Sample of Server.InsufficientInstanceCapacity (1.084% [7] of total count)

**A screenshot of a cell phone

Description automatically generated**

**Reference:** (sourcetype=aws:cloudtrail) <https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-event-reference-record-contents.html>

**Comments and Analysis:**

The following are fields of significance from the AWS CloudTrail log:

errorCode: AccessDenied 🡪 AWS defines this as: The AWS service error if the request returns an error.

eventType: AwsApiCall 🡪 AWS defines eventType as: Identifies the type of event that generated the event record. This can be the one of the following values. AwsApiCall is defined as: An API was called.

eventName: GetUser 🡪 AWS defines eventName as: The requested action, which is one of the actions in the API for that service.

sourceIPAddress: 139.198.18.205

userAgent: [Boto3/1.7.42 Python/2.7.12 Linux/4.4.0-131-generic Botocore/1.10.42]

eventTime: timestamp; ex. 2018-08-20T09:26:40Z

Threat Intelligence:

A majority of the errorCode events included a sourceIPAddress from 139.198.18.205. This IP address consisted of 99.56% of the client requests where “UnauthorizedOperation” was the errorCode reported in the events. Threat Intelligence query was executed using IBM X-Force Exchange 🡪 <https://exchange.xforce.ibmcloud.com/ip/139.198.18.205>

IBM X-Force Exchange categorized 139.198.18.205 as “Unsuspicious”. The Location as identified as “China”. The Registrant Name is “Yunify Technologies Inc.”. The Registrant Organization is “YUNIFY-NET”. The Registrar Name is APNIC. The Email is [ipas@cnnic.cn](mailto:ipas@cnnic.cn).

Running a url check with VirusTotal on the cnnic.cn domain results in “Clean” from all contributors of the Community.

An additional search was conducted in the RIPE database: <https://apps.db.ripe.net/db-web-ui/query?searchtext=139.198.18.205> . This IP is not registered in the RIPE database. Results in “NON-RIPE-NCC-MANAGED-ADDRESS-BLOCK”

The APNIC (Asia Pacific) Database was queried here: <https://wq.apnic.net/static/search.html> . The results of this database show:

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

The significance of the APNIC search is that this IP is not managed by an ISP and is not empowered to investigate complaints of network abuse. Therefore, if a host on this network (139.198.0.0 – 139.198.255.255) was compromised for malicious use, then there is not authority to investigate the complaint.

A simple Google query was executed searching for the e-mail: “mazhiqiang@yunify.com”. Results from the Google search found a blog post here: <http://bannedhackersips.blogspot.com/2018_02_21_archive.html>

This blog post indicated that on 2-21-2018, The IP 139.198.188.137 was banned by Fail2Ban after 5 attempts against SSH. This IP is on the same network and has the same e-mail contact, netname, etc. as 139.198.18.205 (included in the AWS logs).

Additional Google searches for “mazhiqiang@yunify.com” report a brute force SSH attack on their network: <http://bruteforcers.net/633040> . The details of the attack indicate that the same e-mail contact and org-name (“Yunify Technologies (HK) Limited”).

**Narrative:**

Based on the evidence collected, it is reasonable to conclude that any IP from the 139.198.0.0 – 139.198.255.255 network range is potentially compromised and can be used for remote attacks on hosts globally. Hosts on this network are not reportable and there is no manageable authority to report abuse.

When observing the errorCode data in the query results for the web\_admin IAMUser it is evident that remote execution of python code from a Linux OS instance was being executed from 139.198.18.205 and demonstrated reconnaissance behavior. The web\_admin user account was being used, along with its AWS API key (accessKeyId) to remotely access AWS resources such as an EC2 instance and S3 bucket. The reconnaissance activities included Unauthorized Operations on an EC2 instance such as Unauthorized AWSApiCalls. 231 Unauthorized Operations within a short period of time indicate brute force activity. Additionally, there were 280 counts where the web\_admin user account attempted to exceed the EC2 instance limit further indicating a brute force attack. The web\_admin user account logged 7 AccessDenied events that include attempts to “GetUser”, “ListBuckets”, “ListAccessKeys”, “CreateAccessKey”, “DeleteAccessKey”, and “CreateUser”. The web\_admin user account recorded 17 success events for “GetCallerIdentity” and “GetSessionToken”. These behaviors exhibit reconnaissance on what permissions the web-admin user account can and cannot do on AWS using the AWS Security Token Service (AWS STS) API calls.

**MITRE ATTACK MAPPING:**

The errorCode events logged from activity by the web\_admin IAMUser from 139.198.18.105 are “Abnormal”. The “AWS Matrix” is located here: <https://attack.mitre.org/matrices/enterprise/cloud/aws/>

Within the AWS Matrix, the behavior logged by these events is closely aligned to the “Credential Access” category. Specifically, the “ ” is the Technique observed. Here is the link with the information to the “Cloud Instance Metadata API” Technique details: <https://attack.mitre.org/techniques/T1522/>

The recommended mitigation includes: “**Filter Network Traffic**” and is described as “*Limit access to the Instance Metadata API using a host-based firewall such as iptables. A properly configured Web Application Firewall (WAF) may help prevent external adversaries from exploiting Server-side Request Forgery (SSRF) attacks that allow access to the Cloud Instance Metadata API.”*

**Recommendation:**

The web\_admin account has been compromised. Change the password for the web\_admin account or disable the account until all vectors of access from the account have been investigated. This may cause an interruption in services if this account is being used for any service reason. Investigate all uses of the web\_admin account to see what it has access to and it being used for. All services and hosts using web\_admin should be investigated further for vulnerabilities and exploits. Follow recommended MITRE Framework Mitigation strategy to “Filter Network Traffic”.

**Cyber Analysis Methodology (CAM) Summary:**

**Event Type**: Credential Access: Cloud Instance Metadata API (using MITRE Framework)

**Iteration #1**: (Initial Artifact Gathering)

**Question(s):**

List out the IAM users that accessed an AWS service (successfully or unsuccessfully) in Frothly's AWS environment?

**Technologies:**

aws:cloudtrail

**Actions:**

Action 1: Query events for all IAMUsers logged in aws:cloudtrail:

index=botsv3 earliest=0 sourcetype=aws:cloudtrail user\_type=IAMUser| top limit=0 userName

Result of query 🡪 bstoll,btun,splunk\_access,web\_admin

Action 2: Query each IAMUser activity for anomalous data in errorCode field

index=botsv3 earliest=0 sourcetype=aws:cloudtrail user\_type=IAMUser userName=web\_admin errorCode="Client.Unsupported"

Result of queries 🡪 web\_admin user has logged events with anomalous data in errorCode field

**Artifact:**

**IAMUser web\_admin**

**Iteration #2** (Artifact Analysis)

**Question(s):**

What errorCode event anomalies can be observed from Client.InstanceLimitExceeded?

What errorCode event anomalies can be observed from Client.UnauthorizedOperation?

What errorCode event anomalies can be observed from Client.Unsupported?

What errorCode event anomalies can be observed from success?

What errorCode event anomalies can be observed from AccessDenied?

What errorCode event anomalies can be observed from Server.InsufficientInstanceCapacity?

**Technologies:**

aws:cloudtrail 🡪 EC2, S3, STS

**Actions:** **139.198.18.105**

Action 1: Query each errorCode for IAMUser Username=web\_admin and identify all eventName, eventType, sourceIPAddress, userAgent, and eventTime field results. Look for anomalies in all of these fields.

Aggregated statistics from Splunk show the common sourceIPAddress of 98.607% for all errorCode events for IAMUser web\_admin.

**Artifact:**

**IP ADDRESS: 139.198.18.105**

**Iteration #3:** (Artifact Analysis)

**Question(s):**

What threat intelligence exists for 139.198.18.105

**Technologies:**

IBM X-Force Exchange, VirusTotal, APNIC database, Google Search

**Actions:**

Action 1: Query IBM X-Force Exchange

<https://exchange.xforce.ibmcloud.com/ip/139.198.18.205>

“Unsuspicious”. cnnic.cn domain is identified

Action 2: Query cnnic.cn domaon on VirusTotal

The cnnic.cn domain results in “Clean” from all contributors of the Community.

Action 3: Query APNIC database for IP 139.198.18.205

“IP is not managed by an ISP and is not empowered to investigate complaints of network abuse”. e-mail: “mazhiqiang@yunify.com” was identified.

Action 4: Google Query search the e-mail address “mazhiqiang@yunify.com”.

2 Google query results recorded SSH brute force from resources on network range 139.198.0.0 – 139.198.255.255.

**Artifact:**

**“mazhiqiang@yunify.com”**

**Document Findings/Narrative: END OF CAM**