FORCEPOINT

Forcepoint CASB and AWS Security Hub

Integration Guide

Rabih Abou Fakher
Mattia Maggioli
09 April 2020
Public

Version	Date	Author	Notes
0.1	19 December 2019	Rabih Abou Fakher	First draft
0.2	19 December 2019	Mattia Maggioli	Review
0.3	23 December 2019	Rabih Abou Fakher	Update
0.4	07 January 2020	Mattia Maggioli	Review
0.5	07 January 2020	Rabih Abou Fakher	Update
0.6	09 January 2020	Mattia Maggioli	Review
0.7	13 January 2020	Jon Knepher / Audra Simons	Review
0.8	18 January 2020	Rabih Abou Fakher	Update
0.9	18 February 2020	Rabih Abou Fakher	Updated with ARN changes
1.0	02 March 2020	Rabih Abou Fakher	Added docker implementation
1.1	23 March 2020	Neelima Rai	Added troubleshooting chapter
1.2	09 April 2020	Mattia Maggioli	Updated references to ASFF format

Summary

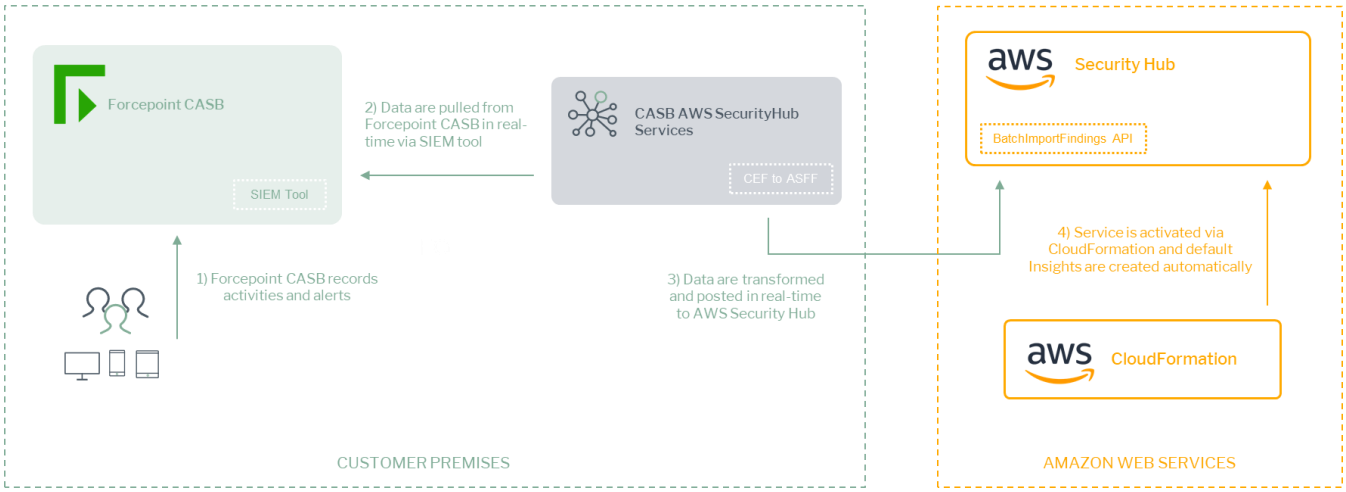
This guide provides step by step instructions to integrate Forcepoint CASB with AWS Security Hub and to export pertinent log data from the CASB SIEM Tool to AWS according to user-configured filters.

The code and instructions provided enable system administrators to automatically:

- Export log events from Forcepoint CASB SIEM Tool into AWS Security Hub in real-time
- Ingest logs as “Findings” inside AWS Security Hub and group them into “Insights” using pre-defined examples created programmatically

This interoperability allows centralization of CASB SIEM Tool logs and events and allows for easy curation of data using “Insights” to group “Findings” by a number of fields (e.g. Severity, Action).

A description of the workflow between the components involved in this POC is depicted in the diagram below:



Caveats

These implementation instructions are tested with the following product versions

- Forcepoint CASB SIEM Tool - Linux version as of 2019-04-15
- AWS Security Hub – ASFF format as of 2020-04-13

Implementation options

Two implementation options are provided in this document

- **Docker** – leverages a docker image where the integration component is already installed with all necessary dependencies: the user only has to edit one configuration file and run the container on

an existing docker setup

- **Traditional** – requires manual deployment of the integration component inside a clean host machine (recommended) or an existing one, provided all requirements are satisfied.

The docker image for exporting risk level information has been tested working with the following requirements

- Docker 19.03.5
- The docker host machine should meet the minimum hardware requirements of 2GB RAM, 50GB free storage, 64bit operating system

while the traditional version of the risk exporter can be deployed on either of the following operating systems

- CentOS 7.x or Ubuntu 18.04 (64bit versions only) with at least 2GB RAM, 50GB free storage

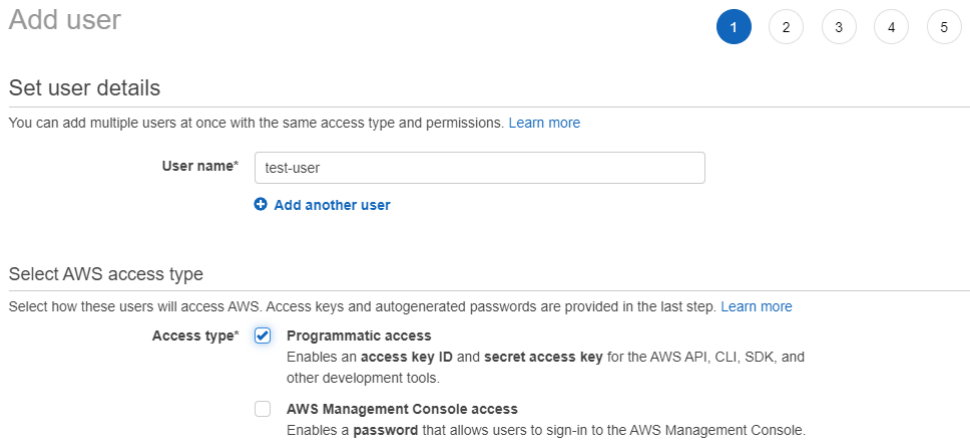
and requires the following dependencies

- Java 1.8
- Python 3.6
- Pipenv
- Unzip

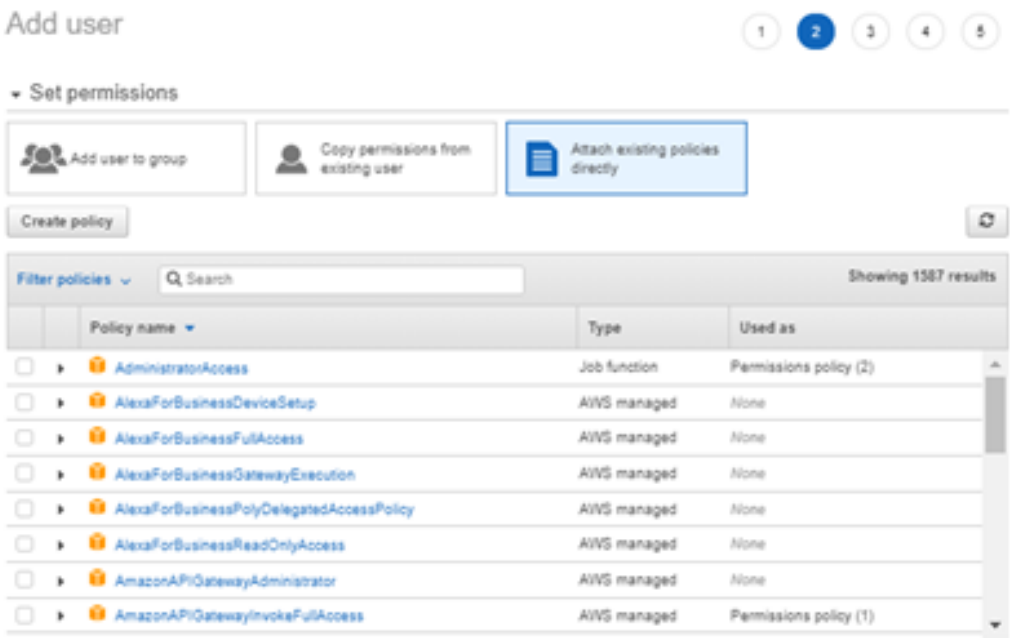
Register a user in AWS and retrieve credentials

To submit logs into AWS Security Hub, retrieve and configure AWS settings as described in this process. If AWS Security Hub is not already active, it will be activated automatically by the installation script.

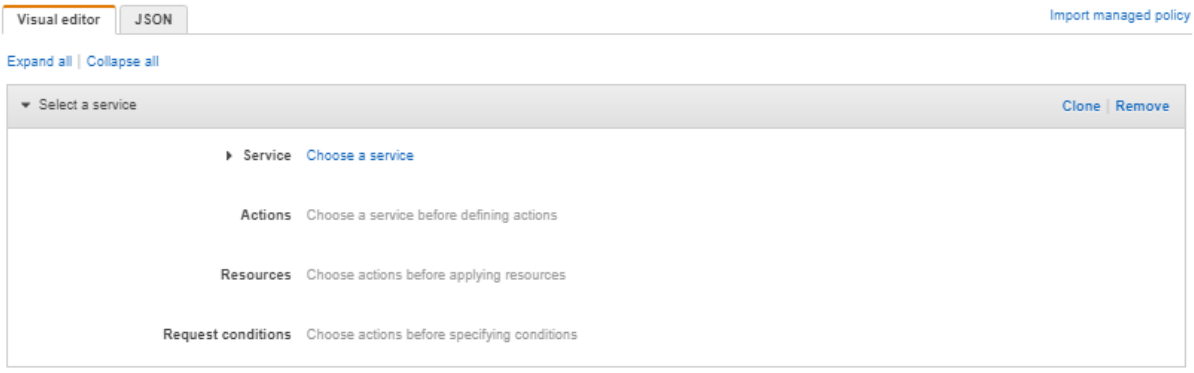
1. Log in to the AWS management console
2. Click on your username in the top right corner and select **My Account**, look for **Account Id** at the top of the page and store the ID in a safe location as it is required for configuring the service in the next steps of this guide
3. Navigate to the AWS management console
4. Search for **IAM** and open it
5. Open the **Users** section and click **Add User** in the top left
6. Enter a name for the new user and select **Programmatic access** in the **Access type** section



7. Select **Attach existing policies directly** and click **Create policy**

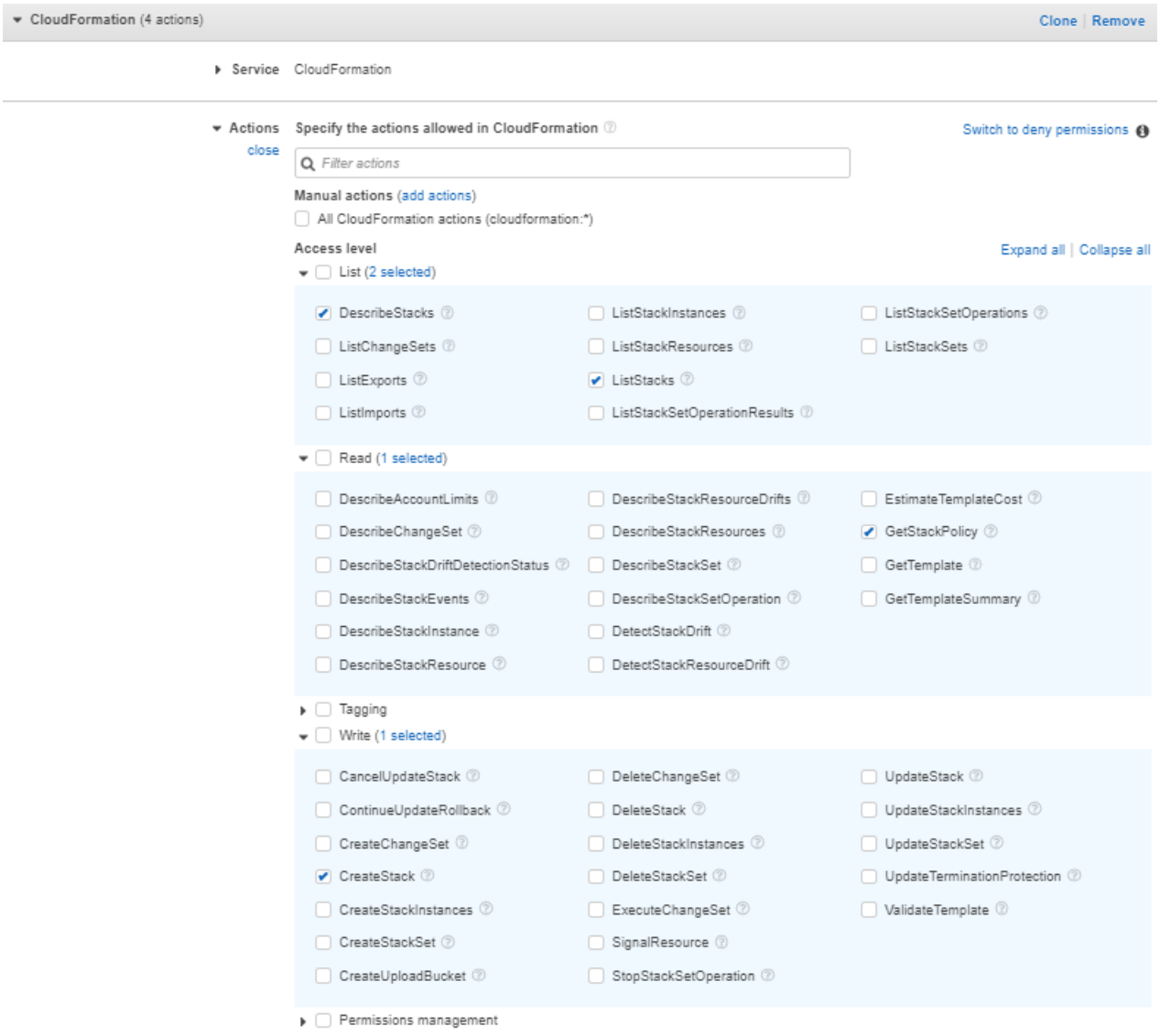


8. On the new page that opens select **Choose a service**

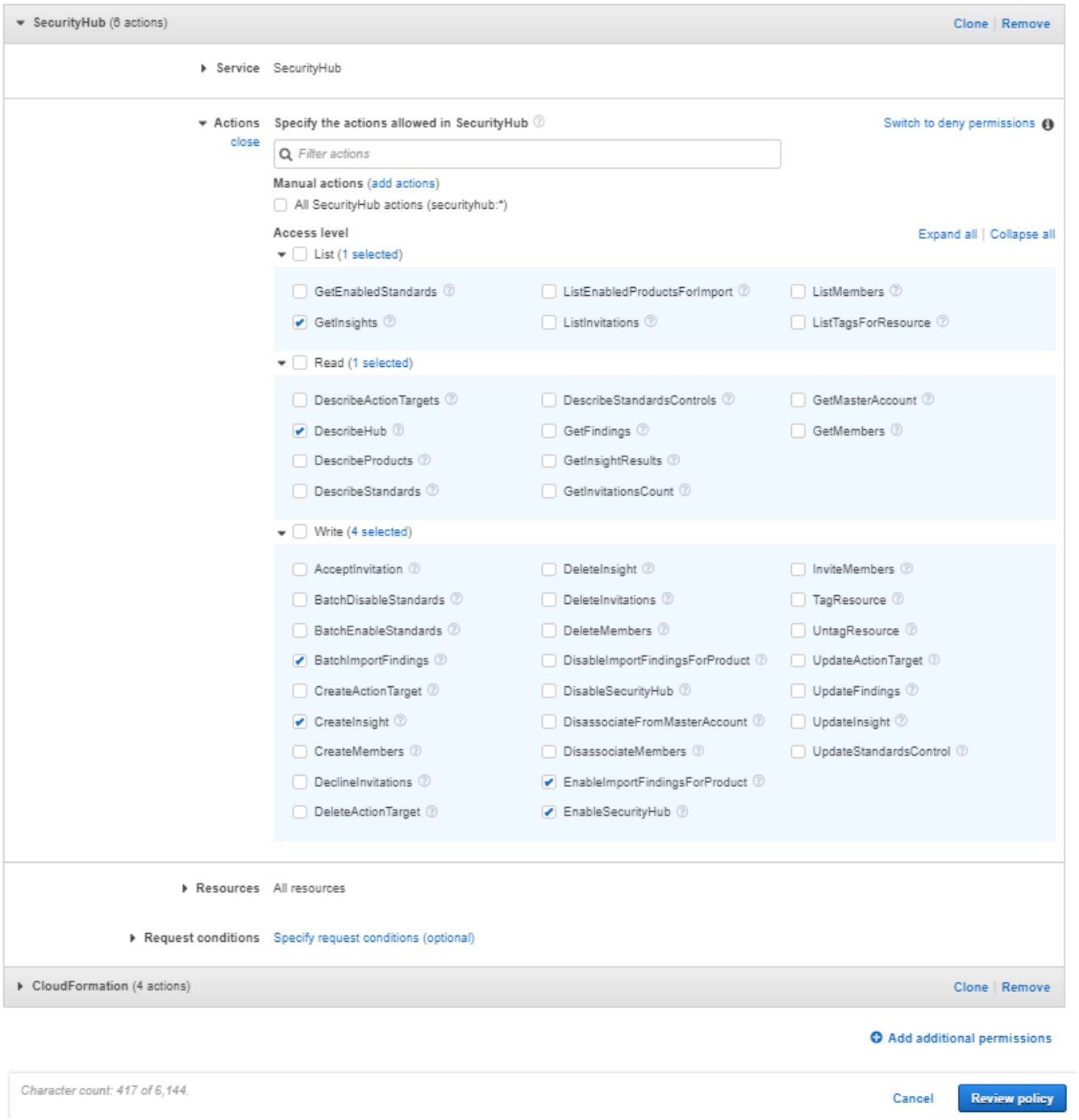


9. Type “CloudFormation” and tick the minimum necessary permissions needed for our setup: **ListStacks, DescribeStacks, GetStackPolicy, CreateStack**.

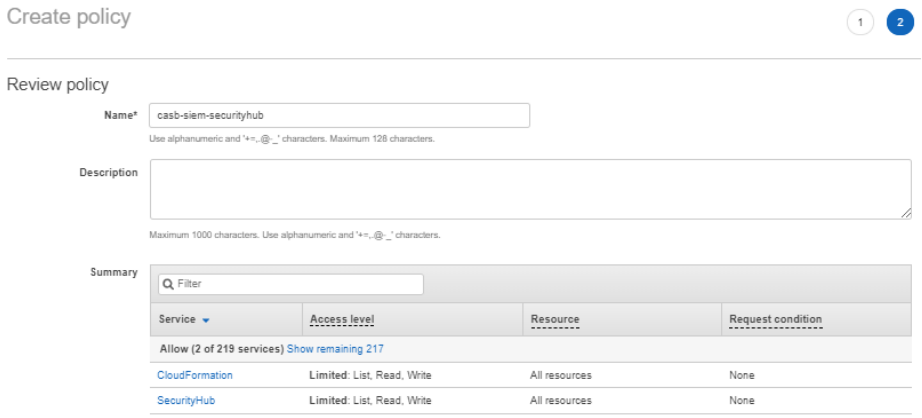
   Under **Resources** tick **All resources**.



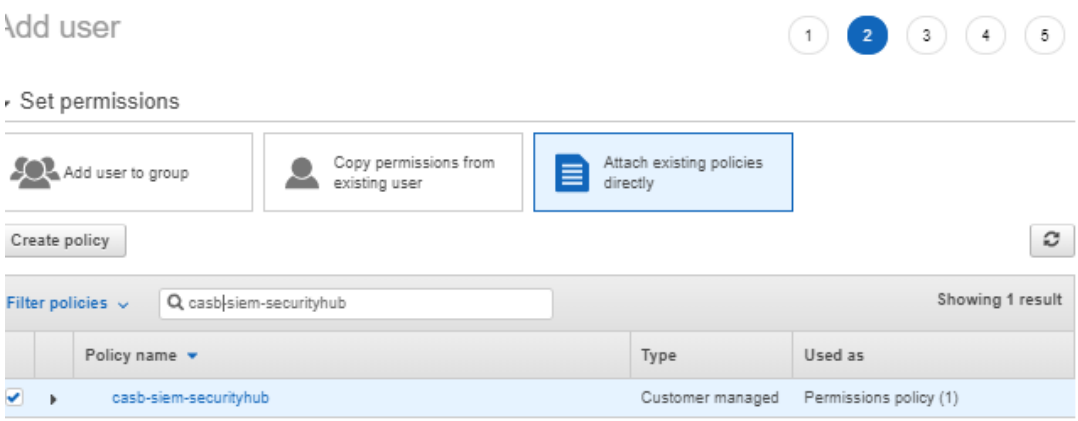
10. Click **Add additional permissions** and select **Choose a service**
11. Type “SecurityHub” and tick: **GetInsights**, **DescribeHub**, **EnableSecurityHub**, **BatchImportFindings**, **CreateInsight, EnableImportFindingsForProduct** . Select **All resources** for **Resources**.



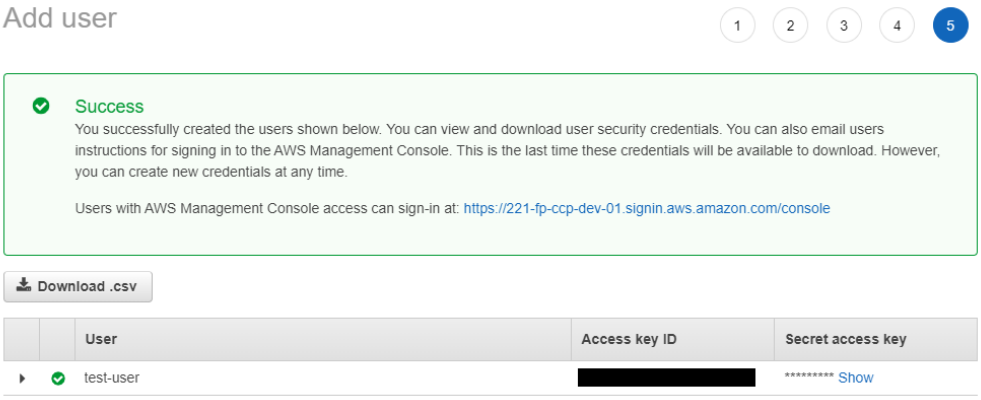
12. Click **Review policy**
13. Type a policy name in the **Name** field and then click **Create policy**



14. Back on the **Add user** page, click the refresh icon and type the new policy name



15. Select the policy and click **Next**
16. Add tags if required in your organization (tags are not required by this integration)
17. Review the details and then click **Create user**
18. In the next screen you will be presented with your new user along with your **Access key ID and Secret access key,** save these or the CSV file in a secure location, this is the only time the Secret access key will be available.



Implementation - Traditional

The solution for the traditional implementation described in this chapter below requires the following files available at this link: https://frcpnt.com/casb-securityhub-latest

- fp-casb-exporter-aws-v1.tar.gz

The file **fp-casb-exporter-aws-v1.tar.gz** contains all files necessary to setup and run Forcepoint CASB connector to AWS Security Hub which automatically monitor, process and upload logs to AWS.

We suggest deploying this service on a clean CentOS 7.x or Ubuntu 18.04 machine with at least 2GB RAM, 20GB free storage and the system needs to be 64 bit, the instructions provided in this document are based on a machine running Ubuntu 18.04 which will be referenced as **Log Proxy** in the rest of this document.

Setup the environment - Traditional

1. Log into the **Log Proxy** machine and unpack the integration package using the command

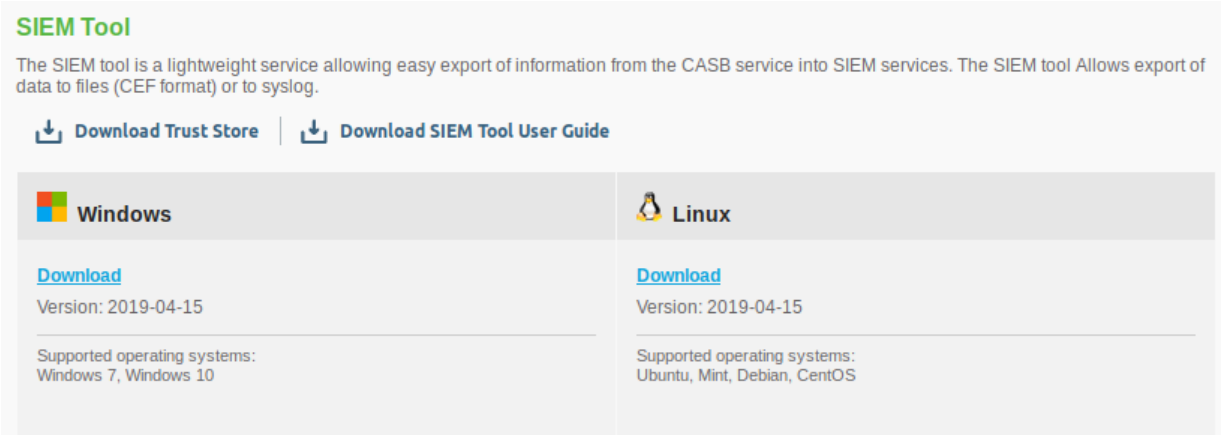
   tar -zxvf ./fp-casb-exporter-aws-v1.tar.gz

2. Move into the **fp-casb-exporter-aws** folder and run the following commands to install the necessary dependencies needed by our integration package, run with a user that has an administrative privileges.

   cd ./deploy
   ./install.sh

Setup CASB SIEM Tool – Traditional

1. Obtain the Linux version of the SIEM tool from Forcepoint CASB management portal: go to **Settings** > **Tools and Agents** (last icon on the left sidebar) > **SIEM Tool** and download both the Linux version and the **TrustStore** file into the **Log Proxy** machine



2. Place both files in the same location
3. Extract the provided SIEM tool archive using the command

   unzip ./SIEM-Tool-Linux-.zip*

Credentials used by the CASB SIEM Tool are generated using the **TrustStore** file, this only needs to be done one time.

4. Run the following command to generate a credentials file that will be used by the CASB SIEM Tool to export data from CASB. Change the red parts with the actual credentials of a CASB account with administrator access, and enter a file name for the credentials file that will be generated:

./SIEMClient.sh --set.credentials --username <user> --password <password> --credentials.file <file>

5. Create a directory where the SIEM tool will store the exported logs

   mkdir casb-siem-files && sudo chmod ugo+rw $_

Setup CASB AWS Security Hub Services - Traditional

1. Edit the file **cfg.json** with all settings as required, more information about the possible values can be found in the Appendix section of this document.

```
{
    "siemPath": "",
    "firstRunSendHistoricalData": true,
    "deleteSiemFiles": true,
    "severityFilterInclude": ["Info", "Low", "Medium", "High", "Critical"],
    "actionFilterInclude": ["Block", "Monitor"],
    "productFilterInclude": ["SaaS Security Gateway", "CASB Incidents", "CASB Admin audit log", "Cloud Service Monitoring"],
    "awsAccountId": "",
    "awsAccessKeyId": "",
    "awsSecretAccessKey": "",
    "regionName": ""
}
```

We recommend reviewing a selection of CASB logs offline, in order to identify the values which better identify the events that are to be exported into AWS Security Hub.

2. Move to the **fp-casb-exporter-aws/deploy** folder and edit **casb-siem-setup.sh** with all settings required, more information about the possible values can be found in the Appendix section of this document

casb-siem-setup.sh

```
_SIEM_HOME_DIR=""
_CREDENTIALS_FILE=""
_HOST="my.skyfence.com"
_PORT=443
_OUTPUT_DIR=""
_TRUST_STORE_PATH=""
```

3. Once all files are edited, install **CASB AWS Security Hub Services** using the commands below, run with a user that has an administrative privileges.

cd ./deploy
./setup.sh

Depending on the number of logs exported from CASB, logs matching all filters will be visible after a few minutes into AWS Security Hub.

AWS Security Hub does not store events older than 90 days, so only CASB logs within this timeframe will be processed and sent into AWS Security Hub by our service.

Systemd processes are configured to start **CASB AWS Security Hub Services** at boot of the log proxy machine.

Implementation – Docker

The docker implementation described in this chapter requires one preparatory step before the docker container can be run: the creation of a credentials file which is used by CASB SIEM Tool (already installed within the container) to connect securely to CASB.

Log into the docker host and download both files from Forcepoint CASB management portal

1. Go to **Settings** > **Tools and Agents** > **SIEM Tool** and click **Download Trust Store**, then click **Download** on the Linux version of the CASB SIEM Tool

2. Place both files in the same location
3. Extract the files from the SIEM tool archive into a local folder inside the docker host

Credentials used by the CASB SIEM Tool are generated using the **TrustStore** file, this only needs to be done one time.

4. Run the following command to generate a credentials file that will be used by the CASB SIEM Tool to export data from CASB. Change the red parts with the actual credentials of a CASB account with administrator access, and enter a file name for the credentials file that will be generated:

./SIEMClient.sh --set.credentials --username <user> --password <password> --credentials.file <file>

In the examples provided in the following pages the <file> name is ***casb-credentials-store***

5. Create a directory where the SIEM tool will store the exported logs, use the command

mkdir casb-siem-files && sudo chmod ugo+rw $_

6. Login into docker repository, you'll be asked to enter your username and password:

   docker login docker.frcpnt.com
   Username**: fp-integrations**
   Password**: t1knmAkn19s**

7. Use this command to download the image

   docker pull docker.frcpnt.com/fp-casb-siem-importer

8. Run the container with either one of the following commands, depending on your scenario

- if **casb-credentials-store** file is located locally then run the following command, replacing the red parts with the actual CASB FQDN and full path of **casb-credentials-store** file

*docker run --detach *
*--name fp-casb-siem-importer *
*--env CASB_HOST='my.skyfence.com' *
*--volume <casb-credentials-store-location>:/usr/fp-casb-siem-importer/casb-credentials-store *
*--volume CasbActivityFilesVolume:/usr/fp-casb-siem-importer/casb-activity-files *
docker.frcpnt.com/fp-casb-siem-importer

- if **casb-credentials-store** file is accessed by a URL, then run the below, replacing the red part with the CASB FQDN and with the URL of **casb-credentials-store** file hosted remotely

*docker run --detach *
*--name fp-casb-siem-importer *
*--env CASB_HOST='my.skyfence.com' *
*--env CONFIG_FILE_URL_LOCATION=<config-file-url> *
*--volume CasbActivityFilesVolume:/usr/fp-casb-siem-importer/casb-activity-files *
docker.frcpnt.com/fp-casb-siem-importer

Setup CASB AWS Security Hub Services - Docker

1. Login into docker repository, you'll be asked to enter your username and password:

   docker login docker.frcpnt.com

   Username**: fp-integrations**

Password**: t1knmAkn19s**

2. Run the below to download the image

   docker pull docker.frcpnt.com/fp-casb-exporter-aws

3. Create **cfg.json** file with all settings as required, more information about the possible values can be found in the Appendix section of this document.

```
{
   "siemPath": "/usr/fp-casb-siem-importer/casb-activity-files",
   "firstRunSendHistoricalData": true,
   "deleteSiemFiles": true,
   "severityFilterInclude": ["Info", "Low", "Medium", "High", "Critical"],
   "actionFilterInclude": ["Block", "Monitor"],
   "productFilterInclude": ["SaaS Security Gateway", "CASB Incidents", "CASB Admin audit log",
"Cloud Service Monitoring"],
   "awsAccountId": "",
   "awsAccessKeyId": "",
   "awsSecretAccessKey": "",
   "regionName": ""
}
```

4. Run the container with either one of the following commands, depending on your scenario

   - if **cfg.json** file is located locally then run the following command, replacing the red part with the full path of the cfg.json file

```
docker run --detach \
 --name fp-casb-exporter-aws \
 --volume <ful-path-of-cfg.json>:/usr/fp-casb-exporter-aws/cfg.json \
 --volume CasbActivityFilesVolume:/usr/fp-casb-siem-importer/casb-activity-files \
 docker.frcpnt.com/fp-casb-exporter-aws
```

   - if **cfg.json** file is accessed by a URL, then run the below, replacing the red part with the URL of the cfg.json file to download

```
docker run --detach \
 --name fp-casb-exporter-aws \
 --env CONFIG_FILE_URL_LOCATION=<config-file-url> \
 --volume CasbActivityFilesVolume:/usr/fp-casb-siem-importer/casb-activity-files \
 docker.frcpnt.com/fp-casb-exporter-aws
```

Depending on the number of logs exported from CASB, logs matching all filters will be visible after a few

minutes into AWS Security Hub.

AWS Security Hub does not store events older than 90 days, so only CASB logs within this timeframe will be processed and sent into AWS Security Hub by our service.

Appendixes

Configuration parameters

The following table provides a description of the parameters in the **cfg.json** file

Parameter	Description	Requires to be changed
siemPath	Path of the directory used by the SIEM Tool to store its output, as defined at step 3.5	YES
firstRunSendHistoricalData	If set to true, process all historical SIEM files in case of CASB SIEM Tool logs present before this setup (leave as true if this is an initial setup)	YES
deleteSiemFiles	If set to true, delete SIEM files from their original directory after processing. Should be set to false if the SIEM output is also used by other services.	NO
severityFilterInclude	CASB logs where Severity matches any of the values in this array will be exported to AWS. e.g. Leave only ["High", "Critical"] to export only CASB logs with either value in the Severity field.	YES
actionFilterInclude	CASB logs where Action matches any of the values in this array will be exported to AWS e.g. Leave only ["Block"] to export only CASB logs with this value in the Action field.	YES
productFilterInclude	CASB logs where Product matches any of the values in this array will be exported to AWS. e.g. Leave only ["SaaS Security Gateway", "CASB Incidents"] to export only CASB logs with either value in the Product field.	YES
awsAccountId	Customer's AWS account ID	YES
awsAccessKeyId	Identifier for the AWS access key	YES
awsSecretAccessKey	Secret key for the AWS access key	YES

regionName	AWS region name where Security Hub will receive CASB logs e.g. eu-west-2	YES

The following table provides a description of the parameters in the **casb-siem-setup.sh** script:

Parameter	Description	Requires to be changed
_SIEM_HOME_DIR	Directory where the SIEM Tool is deployed to	YES
_CREDENTIALS_FILE	Full path including the file name for the credentials file created at step 3.4	YES
_HOST	CASB host e.g. my.skyfence.com	YES
_PORT	Port used to connect to CASB, typically 443	NO
_OUTPUT_DIR	Path of the directory used by the SIEM Tool to store its output, as defined at step 3.5	YES
_TRUST_STORE_PATH	Full path including file name for the TrustStore file that was setup for the SIEM Tool	YES

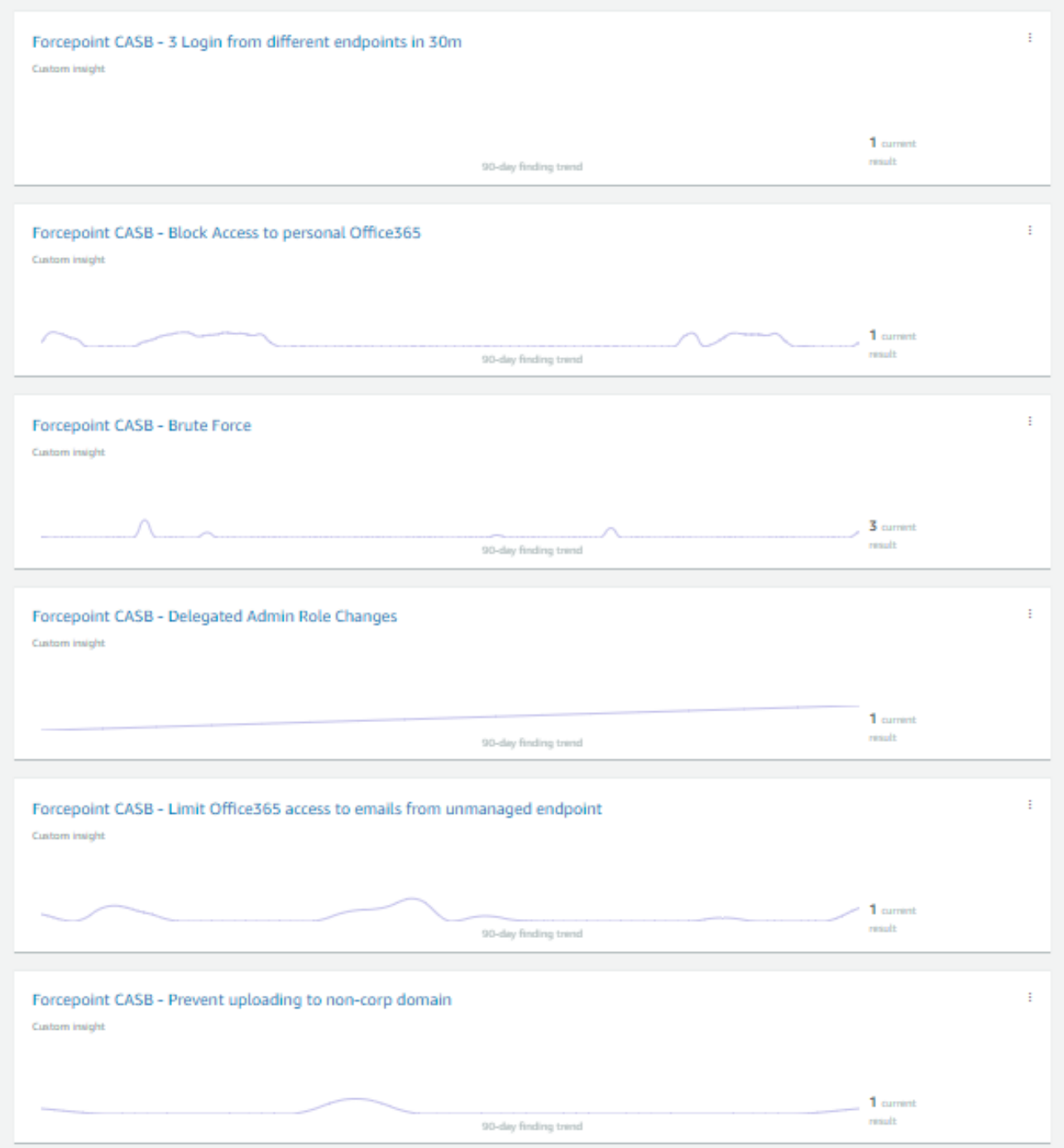
Logs generated by CASB AWS Security Hub Services

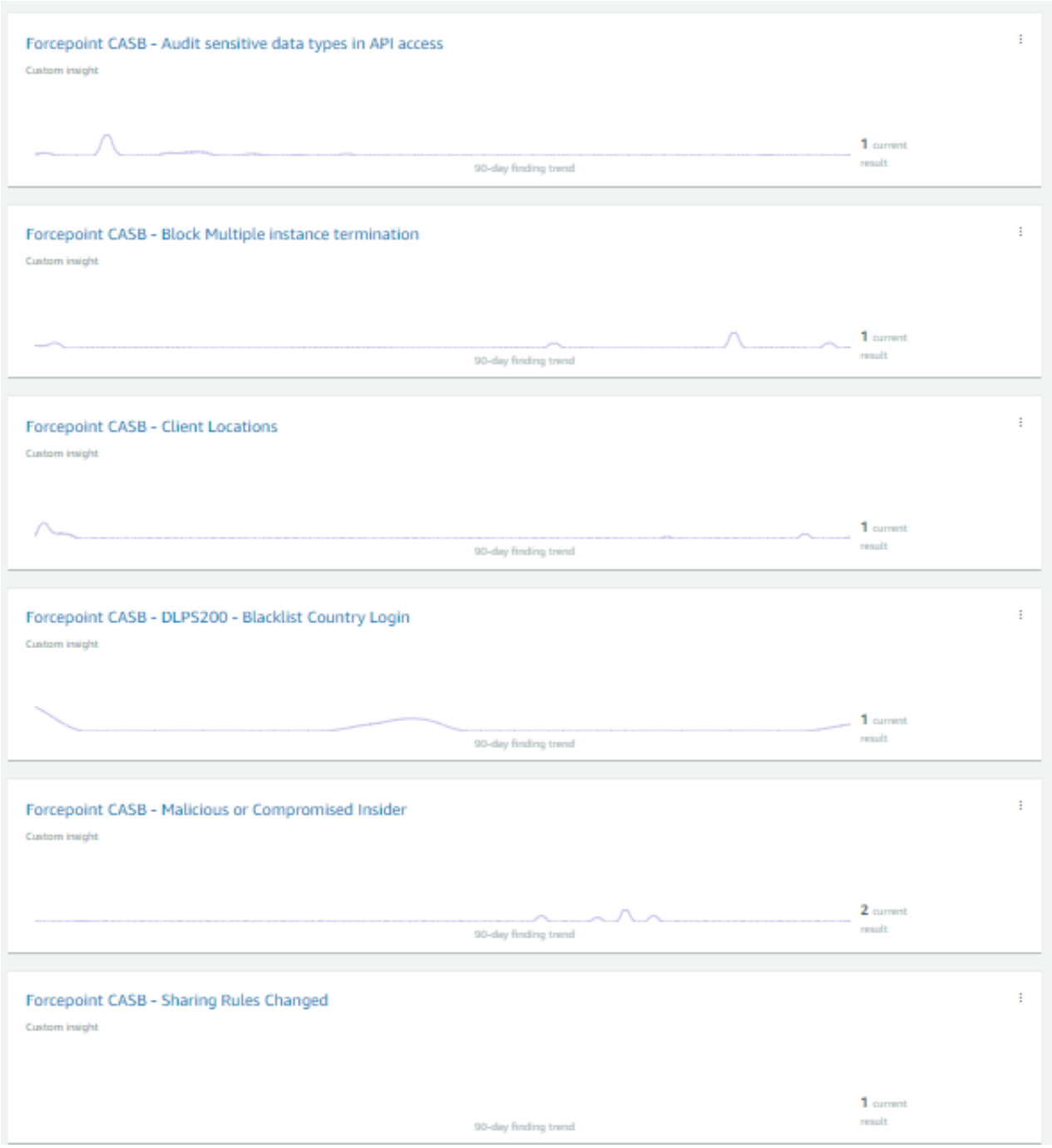
The logs generated by **CASB AWS Security Hub Services** are handled by a log rotate that creates a maximum of 60 MB of logs.

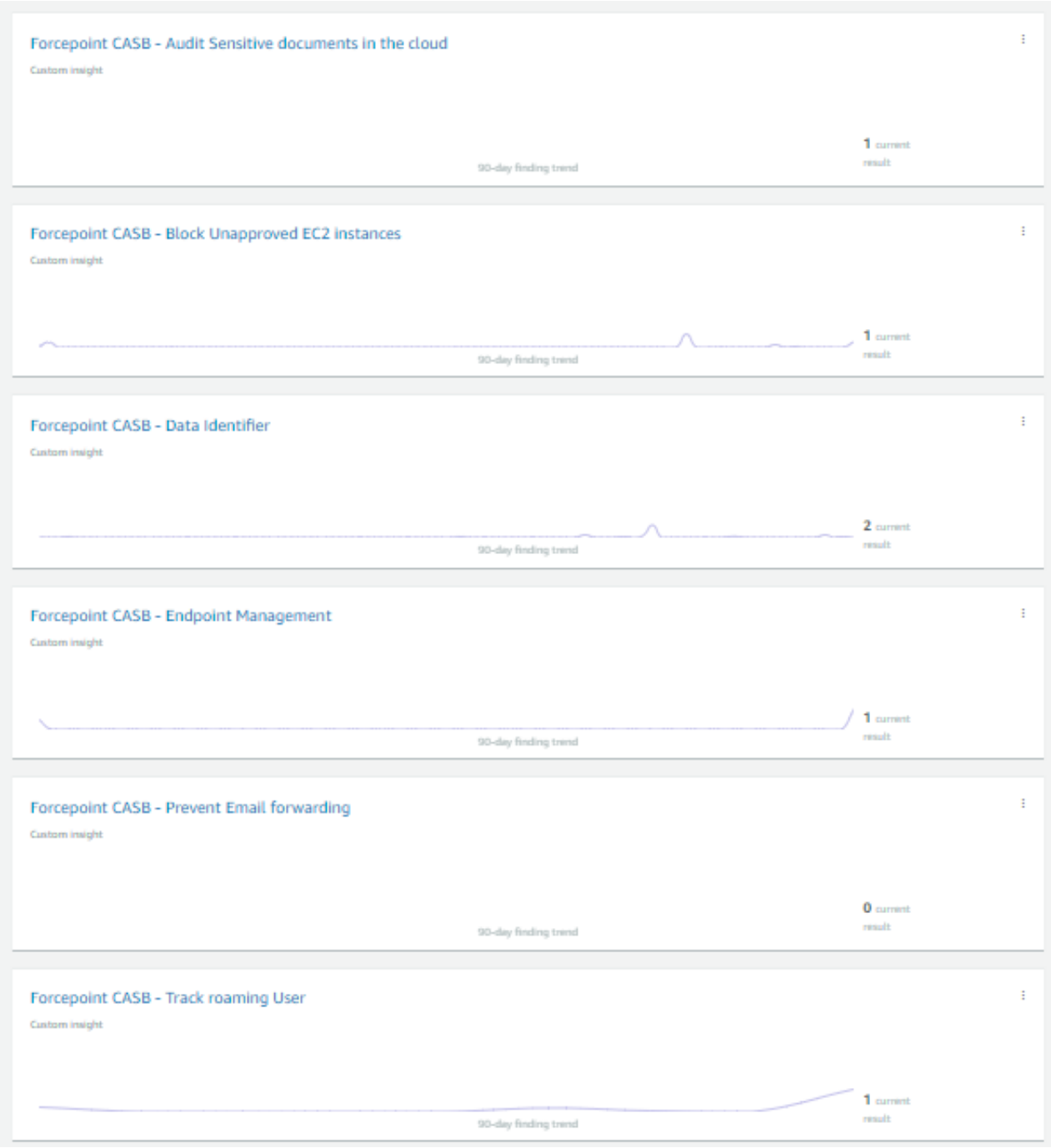
Logs are stored into **fp-casb-exporter-aws/logs** .

Examples of Insights created into AWS Security Hub

Once the integration is set up, a number of Insights are automatically created inside AWS Security Hub: these can be deleted if not necessary or used as examples to create more specific ones.




Forcepoint CASB - Audit sensitive data types in API access
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Block Multiple instance termination
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Client Locations
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - DLPS200 - Blacklist Country Login
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Malicious or Compromised Insider
Custom insight
2 current result
90-day finding trend
Forcepoint CASB - Sharing Rules Changed
Custom insight
1 current result
90-day finding trend


Forcepoint CASB - Audit Sensitive documents in the cloud
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Block Unapproved EC2 instances
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Data Identifier
Custom insight
2 current result
90-day finding trend
Forcepoint CASB - Endpoint Management
Custom insight
1 current result
90-day finding trend
Forcepoint CASB - Prevent Email forwarding
Custom insight
0 current result
90-day finding trend
Forcepoint CASB - Track roaming User
Custom insight
1 current result
90-day finding trend

Troubleshooting

Follow these steps to identify issues impacting the normal operation of the integration described in this document.

Traditional Implementation

Validate the prerequisites

Make sure the prerequisites described in the **Summary** chapter are all satisfied

- Check the version of Forcepoint CASB SIEM Tool in use is listed as compatible

  Forcepoint CASB SIEM Tool - Linux version (2019-04-15)

- The integration component must be installed in a clean CentOS 7.x or Ubuntu 18.04 machine with at least 2GB RAM, 50GB free storage and the system needs to be 64-bit
- User must have sudo permissions
- The _TRUST_STORE_PATH value used in **casb-siem-setup.sh** can be replaced by: */etc/pki/java/cacerts* for CentOS or */etc/ssl/certs/java/cacerts* for Ubuntu
- It is very important to have the correct details in the config file **cfg.json** (for instance, aws region name, account id etc should be correct)
- User must have an admin CASB account for this integration
- Check the user can download the file with the below command:

  wget --content-disposition https://frcpnt.com/casb-securityhub-latest

Check network connectivity

Make sure firewalls or other security appliances are not impacting the network connectivity necessary for the operation of all components involved into this integration

- Check that the Log Proxy machine (host machine) has network connectivity to CASB: execute the following command on the host machine:

  ping -c 2 example-casb.url

  Replace the example URL/IP address with the one used by CASB. Check the result is similar to below:

  PING example-casb.url (10.10.120.12) 56(84) bytes of data.
  64 bytes from 10.10.120.12 (10.10.120.12): icmp_seq=1 ttl=128 time=179 ms
  64 bytes from 10.10.120.12 (10.10.120.12): icmp_seq=1 ttl=128 time=181 ms

- Check that the **Log Proxy** machine (host machine) has network connectivity to AWS, execute the following command on the host machine:

ping -c 2 example-aws.url

replacing the example URL/IP address with the current one used. Once done check the result is like the one below:

PING example-aws.url (10.10.120.12) 56(84) bytes of data.
64 bytes from 10.10.120.12 (10.10.120.12): icmp_seq=1 ttl=128 time=179 ms
64 bytes from 10.10.120.12 (10.10.120.12): icmp_seq=1 ttl=128 time=181 ms

Check dependencies are installed

Make sure the software dependencies needed by the components involved into this integration are installed:

- The *setup.sh* script in */fp-casb-exporter-aws/deploy/* takes care of the installation of all dependencies (python3.6, pipenv and systemctl). It will fail in case these are not installed properly. Check the logs at */fp-casb-exporter-aws/logs/casb-siem-aws.log*
- Run the below command to make sure you have the dependencies installed:

  which python3.6; which pipenv

  and check the result is similar to below:

```
neelima@ubuntu:~/Downloads/fp-casb-exporter-aws-v1/deploy$ which python3.6; which pipenv
/usr/bin/python3.6
/usr/local/bin/pipenv
```

Check all components are configured and running properly

Make sure the products and services involved into this integration are configured as expected and they are running:

- Check all components are configured and running as expected by running following commands:

  cd /fp-casb-exporter-aws/deploy
  ./status.sh

  and check the result is similar to below

```
neelima@ubuntu:~/Downloads/fp-casb-exporter-aws-v1/deploy$ ./status.sh
● casb-siem-aws-securityhub-batch.service - This service picks up all SIEM missed files due to network down... and upload them into AWS Security Hub
   Loaded: loaded (/etc/systemd/system/casb-siem-aws-securityhub-batch.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Mon 2020-03-09 06:40:07 PDT; 1h 41min ago
  Process: 2178 ExecStart=/bin/bash -c pipenv run python ${APP_HOME}/src/modules/main_siem_watcher_batch.py (code=exited, status=0/SUCCESS)
 Main PID: 2178 (code=exited, status=0/SUCCESS)

Mar 09 06:40:05 ubuntu systemd[1]: Started This service picks up all SIEM missed files due to network down... and upload them into AWS Security Hub.

● casb-siem-aws-securityhub-batch.timer - Run casb-siem-aws-securityhub-batch.service weekly and 30 minutes after boot up
   Loaded: loaded (/etc/systemd/system/casb-siem-aws-securityhub-batch.timer; enabled; vendor preset: enabled)
   Active: active (waiting) since Mon 2020-03-09 06:10:06 PDT; 2h 11min ago
  Trigger: Mon 2020-03-16 06:40:05 PDT; 6 days left

Mar 09 06:10:06 ubuntu systemd[1]: Started Run casb-siem-aws-securityhub-batch.service weekly and 30 minutes after boot up.

● casb-siem-aws-securityhub.service - This service watches SIEM tool folder for events, incidents and activities and upload them into AWS Security Hub
   Loaded: loaded (/etc/systemd/system/casb-siem-aws-securityhub.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-03-09 06:10:07 PDT; 2h 11min ago
 Main PID: 646 (python)
    Tasks: 4 (limit: 2293)
   CGroup: /system.slice/casb-siem-aws-securityhub.service
           └─646 /root/.local/share/virtualenvs/fp-casb-exporter-aws-v1-gla_tz5y/bin/python /home/neelima/Downloads/fp-casb-exporter-aws-v1/src/modules/main

Mar 09 06:10:07 ubuntu systemd[1]: Started This service watches SIEM tool folder for events, incidents and activities and upload them into AWS Security Hub.
lines 1-24/24 (END)
```

```
● casb-siem-batch-timer.service - Triggers casb-siem-batch.service to run again for any new events to be picked up by casb-siem-aws-securityhub.service
   Loaded: loaded (/etc/systemd/system/casb-siem-batch-timer.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-03-09 08:21:08 PDT; 1min 44s ago
 Main PID: 3183 (bash)
    Tasks: 2 (limit: 2293)
   CGroup: /system.slice/casb-siem-batch-timer.service
           ├─3183 bash /home/neelima/Downloads/fp-casb-exporter-aws-v1/deploy/casb-siem-batch-timer.sh
           └─3210 sleep 60
```

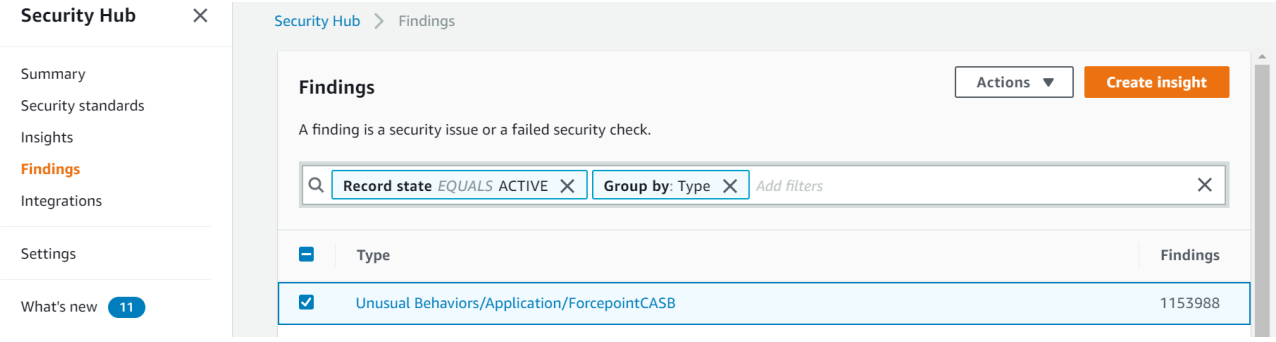
- If the installation fails, try the below for troubleshooting:

1. *cd to /fp-casb-exporter-aws/deploy*
2. Run *./kill-services.sh*
3. Run *./deploy.sh*
4. Check the logs at */fp-casb-exporter-aws/logs* to see if there are any error messages

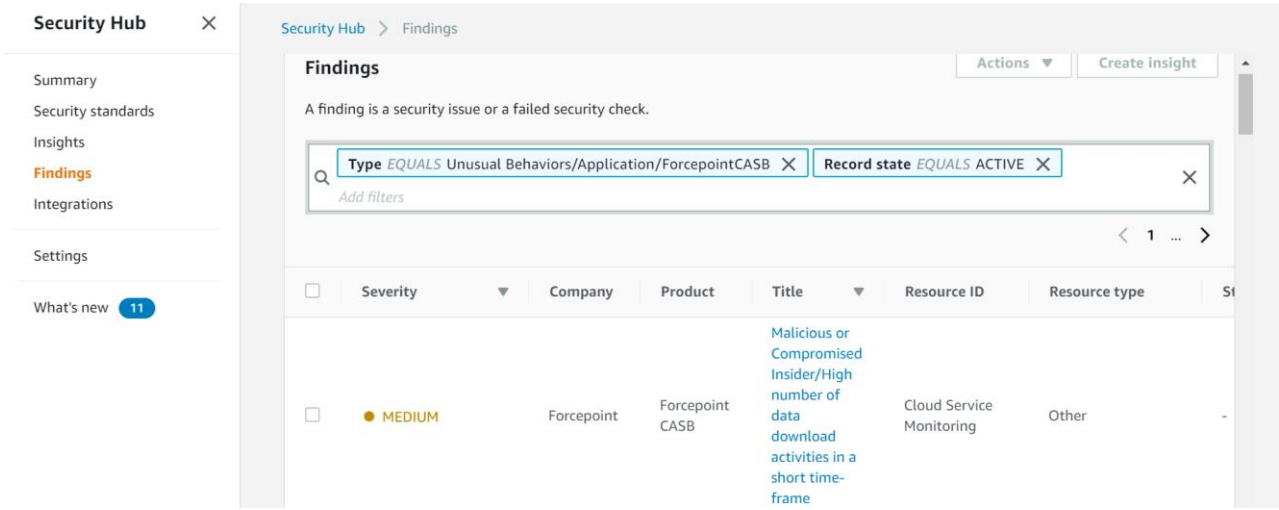
   and check the service status again by running:
   ./status.sh

- Check that the CASB-AWS integration is successful:

  The user should see the Findings in AWS-Security Hub in about 10 minutes. Select the correct AWS region and add filters in Security Hub Findings: Group by > Type > Unusual Behaviors/Application/ForcepointCASB as shown below:



User can get more information on the Finding contents by clicking on the hyperlink in the **Title** of each Finding.



Docker Implementation

Validate the prerequisites

Make sure the prerequisites described in the **Summary** chapter are all satisfied:

- Config file **cfg.json** must be edited with the correct contents (e.g. AWS region name, account id…) as described in the previous chapters of this document
- User must use a CASB account with admin role for this integration
- The docker host machine should meet the minimum hardware requirements of 2GB RAM, 50GB free storage and the system needs to be 64-bit

Check all components are configured and running properly

Make sure the products and services involved into this integration are configured as expected and they are running:

- Check all components are configured and running as expected by running following command

  docker ps

  and check the result is similar to below

```
[neelima@localhost Downloads]$ docker ps
CONTAINER ID        IMAGE                                                  COMMAND                  CREATED             STATUS
          PORTS               NAMES
d[redacted]        [redacted]/fp-casb-exporter-aws    "./container-files..."   28 minutes ago      Up 28 min
utes                          fp-casb-exporter-aws
[redacted]         [redacted]/fp-casb-siem-importer   "./container-files..."   3 hours ago         Up 3 hour
s                             fp-casb-siem-importer
```