FORCEPOINT

Forcepoint Behavioral Analytics and Okta

Integration Guide

Dlo Bagari
Mattia Maggioli
04 September 2020
Public

Version	Date	Author	Notes
0.1	05 November 2019	Mattia Maggioli	First draft
0.2	14 November 2019	Jon Knepher and Audra Simons	Review
0.3	15 November 2019	Mattia Maggioli	Updated step 3 with Kafka consumer over SSL
0.4	30 December 2019	Jonathan Knepher	Review
0.5	21 January 2020	Dlo Bagari	Updated package name
0.6	23 March 2020	Neelima Rai	Added troubleshooting chapter
0.7	04 September 2020	Mattia Maggioli	Minor updates

Summary

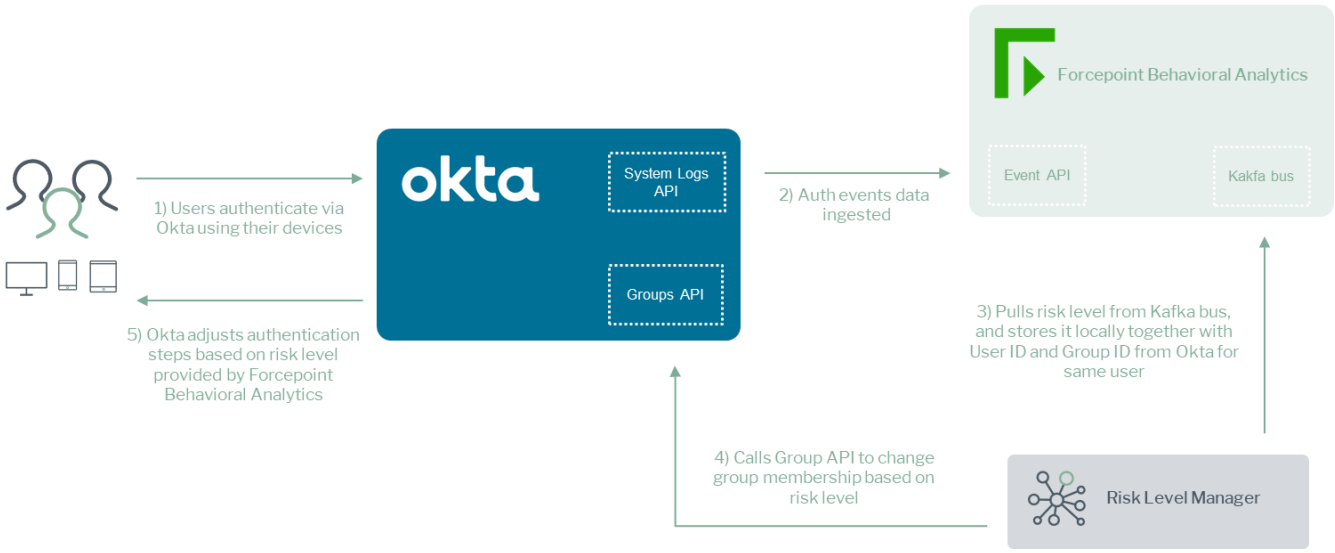
This guide provides step by step instructions to configure Forcepoint Behavioral Analytics and Okta to pass risk scores and login / event information, and to adjust authentication policies accordingly.

The code and instructions provided enable system administrators to automatically:

- Export events from Okta into Forcepoint Behavioral Analytics
- Provide the risk level calculated by Forcepoint Behavioral Analytics for each user to Okta
- Adjust authentication policies applied by Okta to users based on their risk level
- Upon risk level increase, terminate the active sessions of the user: this will force the user to re-authenticate according to the policies applied to the new risk level

This interoperability enriches visibility into user activities, enhances risk scoring, and enables risk-adaptive authentication policy for Okta users based on the intelligence provided by Forcepoint Behavioral Analytics.

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



Caveats

These implementation instructions are tested with the following product versions:

- Forcepoint Behavioral Analytics 3.1.0
- Okta 2020.02.01

This interoperability uses the “System Logs API” of Okta, which provides both successful events and

failed login attempts to Forcepoint Behavioral Analytics via the Forcepoint Streaming Ingest Public API.

The following activities are out of the scope of this document and therefore left to the system administrator, as part of ordinary maintenance procedures to be put in place within the existing infrastructure:

- configuration of appropriate hygiene procedures to handle logs produced during any step of the solution workflow
- monitoring of the scripts, services and applications involved in the solution

Implementation

The solution described in this chapter requires the following files available at this link:
https://frcpnt.com/fba-okta-latest

- fp-fba-connector-okta-v1.tar.gz

The **fp-fba-connector-okta-v1.tar.gz** contains all files necessary to setup and run all the services used by the Risk Level Manager to accomplish the interoperability between Forcepoint Behavioral Analytics and Okta:

- **Event Service**: pulls events from Okta, transforms data into the format used by Forcepoint Behavioral Analytics to ingest data
- **User Service**: validates users in Okta events and creates **Entities** in Forcepoint Behavioral Analytics so that **Identifiers** and **Aliases** are setup appropriately
- **Consumer Manager Service**: obtains risk level from Forcepoint Behavioral Analytics, confirms the risk level is related to a known Okta user before sending the risk level to the Connector Service
- **Connector Service:** forwards the risk level of Okta users to the Risk Level Manager Service to change group membership based on the risk level
- **Risk Level Manager Service**: orchestrates group membership changes upon changes of the risk level

We suggest deploying the **Risk Level Manager** on a CentOS 7.x machine with at least 2 GB RAM and 20 GB free storage, the instructions provided in this document are based on this operating system and the following packages

- Python 3
- Python modules: requests, flask, confluent_kafka

The software packages and related dependencies are automatically installed by the **installer.sh** script provided inside the **fp-fba-connector-okta-v1.tar.gz** file, which will execute the following commands as part of the deployment script of the Risk Level Manager:

sudo yum install -y https://centos7.iuscommunity.org/ius-release.rpm
sudo yum install -y python36u python36u-libs python36u-devel python36u-pip
sudo pip3 install flask
sudo pip3 install requests
sudo pip3 install confluent-kafka

The machine hosting the Risk Level Manager will be referenced in the rest of this document with the name "**RLM-host**".

Step 1 – Create an API token in Okta

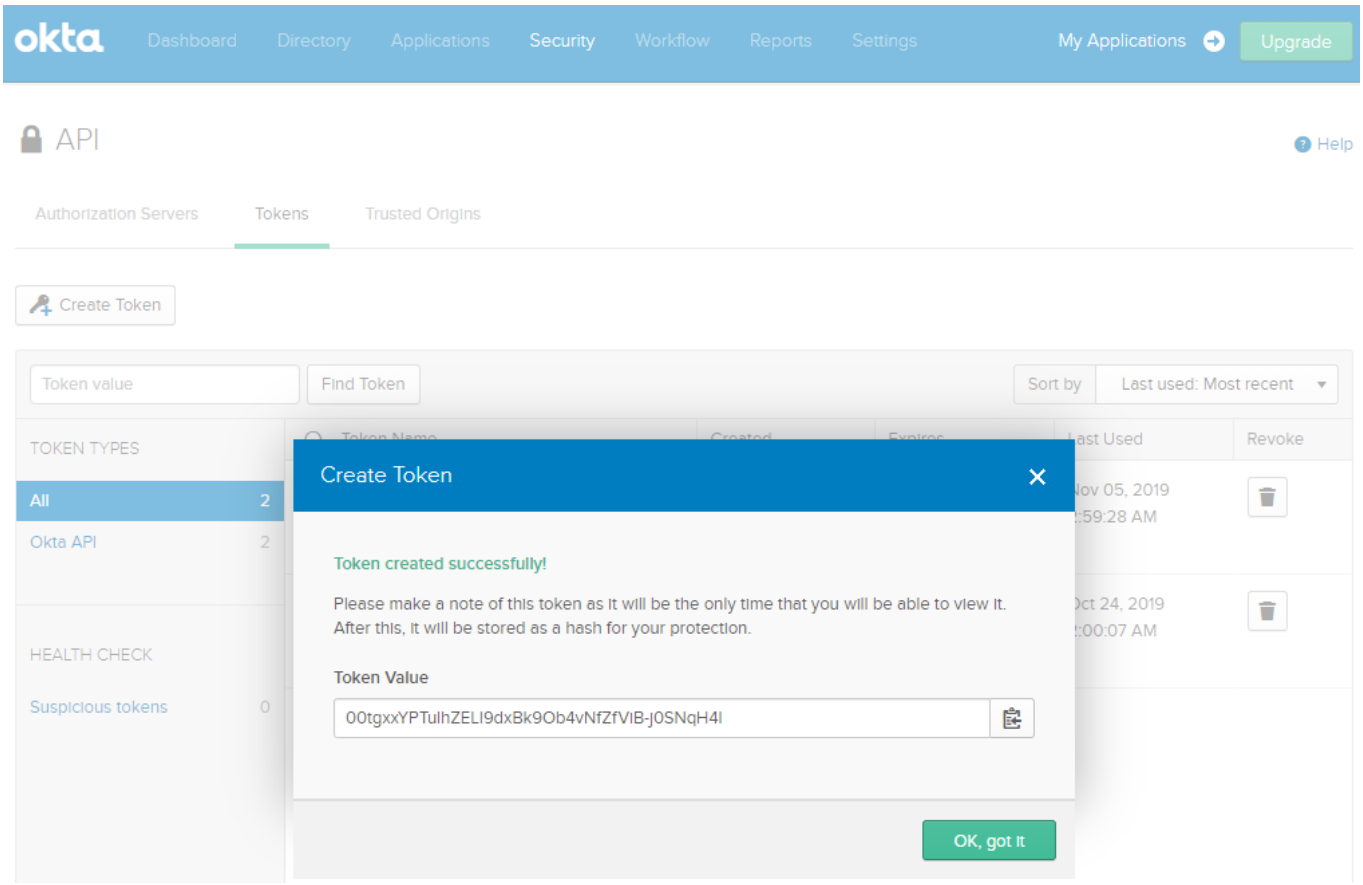
In order to connect and perform operations inside Okta, the Risk Level Manager requires a valid token. API tokens have the same permissions of the user who creates them, and the Risk Level Manager requires administrative access in order to perform its tasks.

It is recommended to create a new user with the minimum roles necessary to issue tokens and to perform the operations of the Risk Level Manager. Only the following **Administrator Roles** are necessary:

- **Read Only Administrator**
- **API Access Management Administrator**

Once a user is created, follow these steps to issue a new token under that user:

1. Sign into your Okta organization using the newly created administrator
2. Make sure you are using the **Classic UI**: you can check this setting in the top left corner of the Okta page, right above the Okta logo
3. Go to **Security > API** and click on the **Tokens** tab
4. Click **Create Token**, name your token and click **Create Token**
5. Save this **Token Value** in a secure location, as this is the only time it can be viewed.



Step 2 – Groups into Okta

Authentication steps applied to users authenticating through Okta are defined as **Sign On rules** configured into **Groups**, for example:

- Source IP of the user authenticating
- Okta Verify (either code or push notification)
- SMS authentication
- Voice Call Authentication
- External MFA apps
- Security question

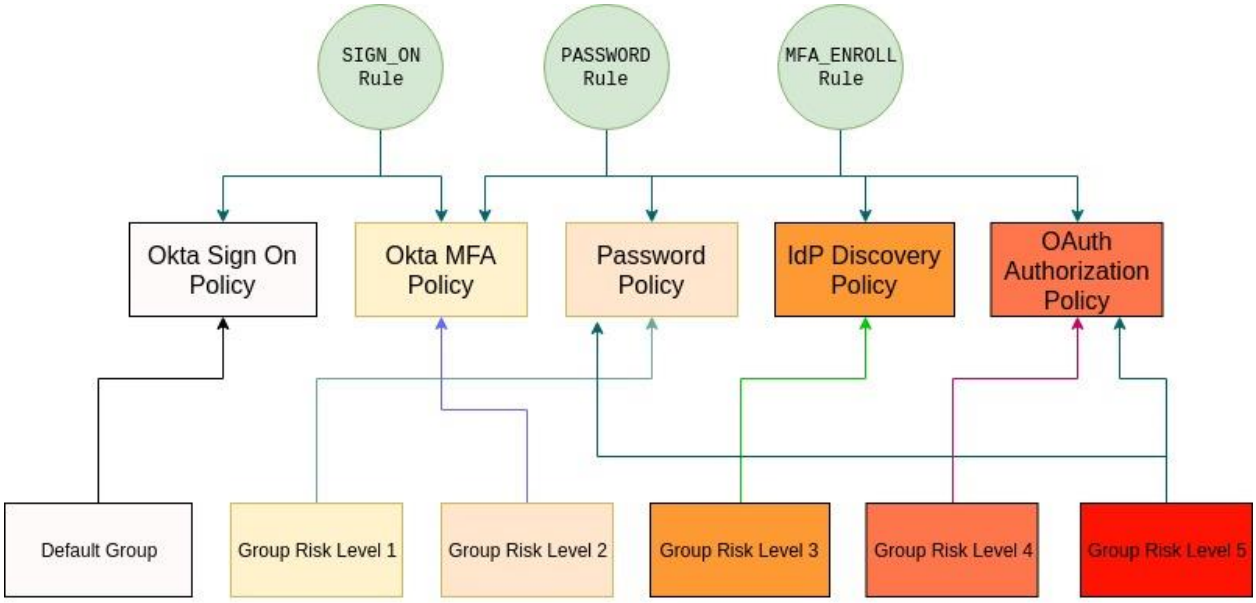
A user authenticating through Okta will be challenged according to the policies configured via the user's group membership. To do this, create user groups to map the desired risk level policies to users.

A typical group and policy configuration is as follows:

- Users with risk_level 1 and 2 (low risk) are assigned to groups with standard authentication

policies (e.g. push notification)

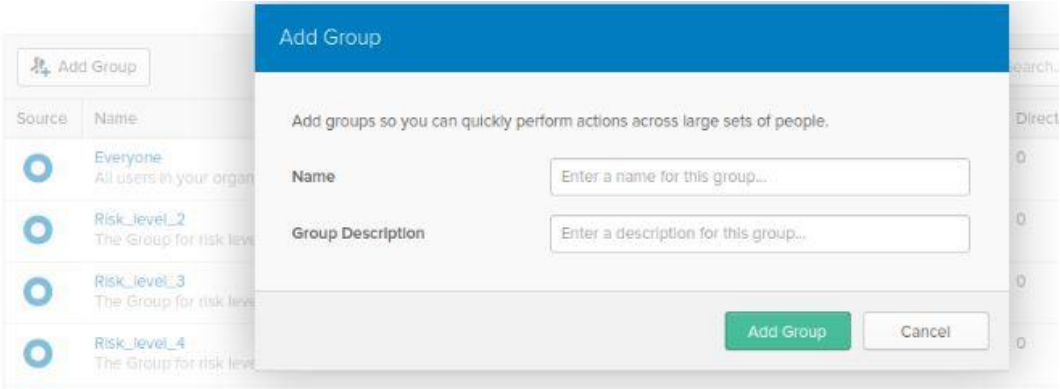
- Users with risk_level 3 and 4 (medium risk) are opposed more complex authentication policies (e.g. push notification and SMS authentication)
- Users with risk_level 5 are denied authentication



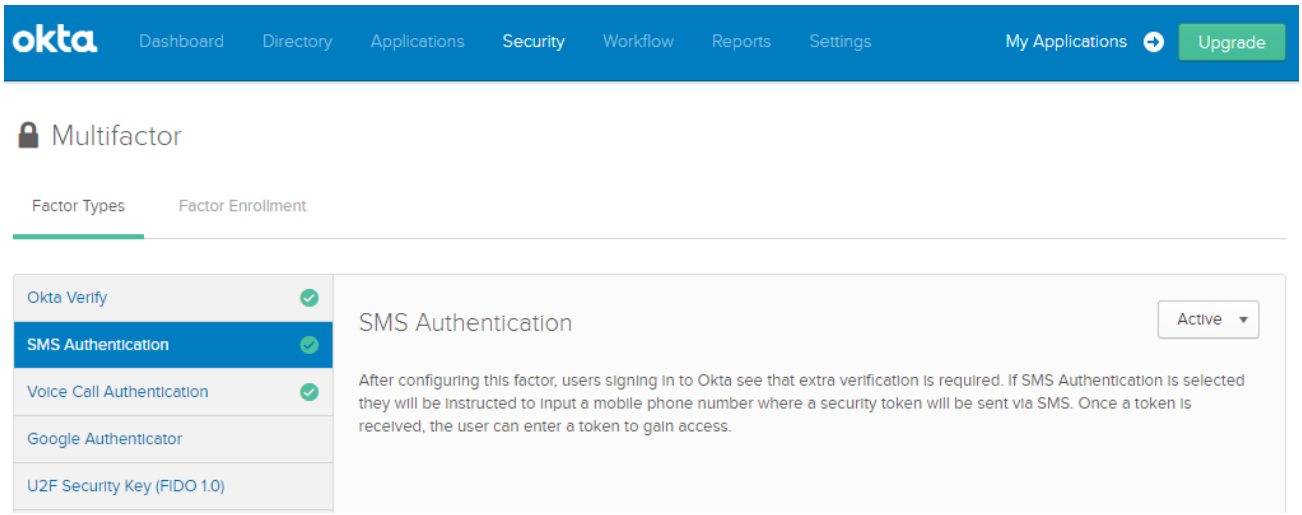
If multiple user groups with existing policies are already configured into Okta, then skip this part and go to **Step 3**.

In the following example, we create a new user group and assign a multifactor policy based on **SMS Authentication**:

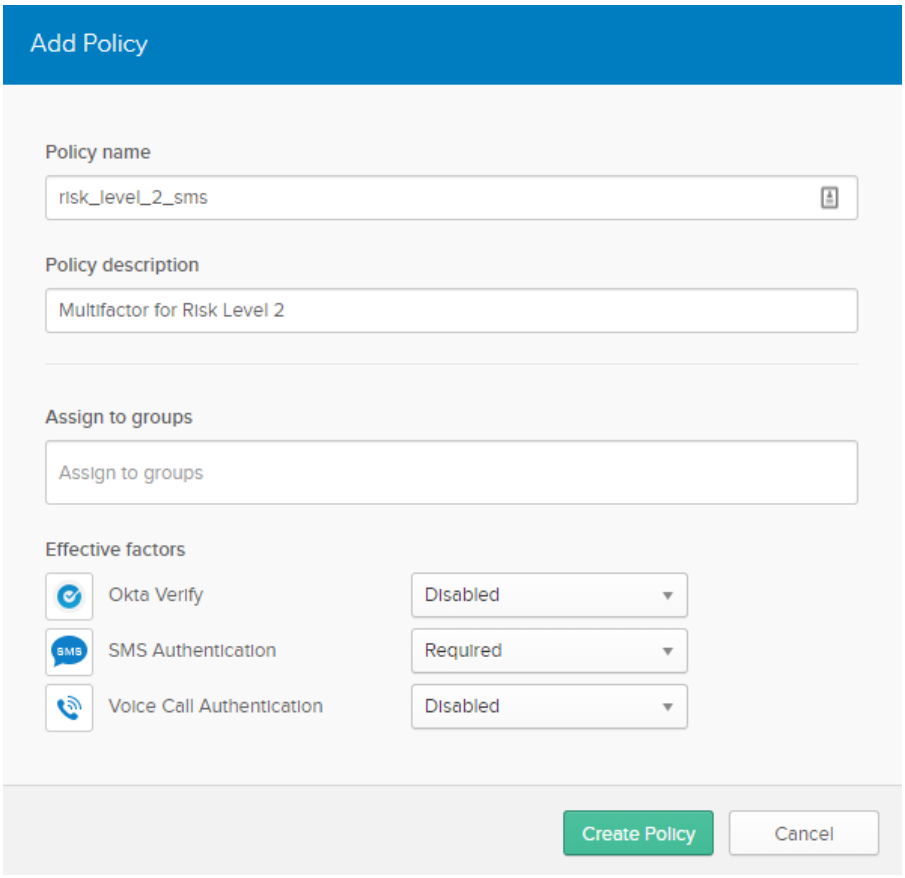
1. Go to **Directory > Groups**, click on **Add Group**
2. Name the new group and add a description, then click **Add Group** to create it



3. Go to **Security > Multifactor**, click **SMS Authentication** and then the drop-down menu so that the new factor is set to **Active**



4. Switch to the **Factor Enrollment** tab and click **Add Multifactor Policy**
5. Name your policy and enter a description, assign it to the group created for this risk level
6. Set the "**SMS Authentication**" to **Required** using the drop-down menu, set to **Disabled** any other factor not assigned to this policy. Once done click **Create Policy**.



7. Now click on the newly created policy and click **Add Rule**
8. Name the rule and set the conditions for the multi-factor enrollment

Multi-factor challenges must be enrolled in order to be applied to any further authentication attempt. It is recommended for all users to be pre-enrolled for any authentication method that may be required. Alternatively, steps 7 and 8 above can also be used to configure multi-factor enrollment when the user is challenged for the first time. In this case it is suggested to configure the rule to only allow first time enrollment if the user is connecting from trusted networks (e.g. configure a network **Zone** in **Networks** so that trusted corporate IP ranges are known to Okta, and use the trusted **Zone** in a rule which allows multi-factor enrollment only when users authenticate from the corporate networks).

Repeat the steps above to create additional groups, each configured with the desired authentication policies to be mapped to each of the 5 risk levels assigned by Forcepoint Behavioral Analytics to the monitored entities.

Step 3 – Configure a Kafka consumer

Configure a Kafka consumer that will connect to the existing Kafka server of Forcepoint Behavioral Analytics over SSL, so that Risk Level Manager can receive newly calculated risk levels, and trigger

changes to group membership.

1. Login via SSH to the Kafka server of Forcepoint Behavioral Analytics
2. Move to **/usr/lib/kafka/config/**
3. Open the file **server.properties** and save in a secure location the password of the keystore and truststore files

   ssl.keystore.location=/etc/kafka/conf/kafka-host-keystore.p12
   ssl.keystore.password=keystore-password
   ssl.key.password=key-password
   ssl.keystore.type=PKCS12
   ssl.truststore.location=/etc/kafka/conf/kafka-host-truststore.p12
   ssl.truststore.password=truststore-password
   ssl.truststore.type=PKCS12

4. Inside the **server.properties** file look also for the configuration of the Kafka listeners

   listeners=SSL://kafka-host:9093
   listener.security.protocol.map=SSL:SSL
   advertised.listeners=SSL://kafka-host:9093

The Kafka server advertises its listeners by sending this information as metadata to the consumers. In the example above the hostname of the listener **kafka-host** is not a FQDN. Assure that the RLM-host can resolve the hostname of the Kafka host, or manually add an entry to the **host** file of the RLM-host, and verify the **kafka-host** hostname can be resolved correctly.

5. Extract the client certificates and key from the keystores using the following commands, replacing the parts in red with the name of your files and entering the passwords (noted before from the **server.properties** file) when requested

openssl pkcs12 -in /etc/kafka/conf/kafka-host-keystore.p12 -nocerts -nodes | sed -ne '/-BEGIN PRIVATE KEY-/,/-END PRIVATE KEY-/p' > client.key

openssl pkcs12 -in /etc/kafka/conf/kafka-host-keystore.p12 -clcerts -nokeys | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > client.cer

openssl pkcs12 -in /etc/kafka/conf/kafka-host-truststore.p12 -cacerts -nokeys -chain | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > client-ca.cer

6. Three files will be created as a result of the commands just executed: **client-ca.cert**, **client.cer** and **client.key**. Move the files to RLM-host and save in a secure location the path to those files as that will be needed in the next chapter of this guide

7. Check the Kafka service is running without errors by using the command **systemctl status kafka**

   If the service is not running, use the command **systemctl start kafka** or

   /usr/lib/kafka/bin/kafka-server-start.sh /usr/lib/kafka/config/server.properties

Step 4 – Configure the Risk Level Manager

All parameters required by the Risk Level Manager to operate its services are stored in a single file called **configs.json**:

```
"application_directory": "/var/okta",
"logs_directory": "/var/okta/logs",
"organization_url": "organization.okta.com",
"fba_api": "fba-host.organization.net",
"rose_api": "rose-host.organization.net",
"mds1_api": "mds1-host.organization.net",
"risk_level_groups": [
  "risk_level_1",
  "risk_level_2",
  "risk_level_3",
  "risk_level_4",
  "risk_level_5"
],
"risk_level_manager_portal": "127.0.0.1:5001",
"connector_portal": "127.0.0.1:5000",
"user_app_portal": "127.0.0.1:5003",
"org_token": "00pOzAdi2HO5BPOpoB-tQhGFYHdu9hnfZivC3BvkaU",
"org_name": "organization",
"kafka_broker": "kafka-host.organization.net:9093",
"kafka_topic_name": "ENTITY_RISK_LEVEL",
"kafka_group_name": "okta_risk_level_ssl_test",
"client-ca.cer": "/var/okta/key_store/client-ca.cer",
"client.cer": "/var/okta/key_store/client.cer",
"client.key": "/var/okta/key_store/client.key",
"key_store_pass": "keystore-password"
```

The following table provides a description of every parameter in the **configs.json** file:

Parameter	Description
application_directory	Directory where the Risk Level Manager is stored, by default /var/okta
logs_directory	Directory where log files are created, by default /var/okta/logs
organization_url	Organization URL for Okta, (used for authenticating users and for API calls by the Risk Level Manager and its services)
fba_api	FQDN of the Forcepoint Streaming Ingest Public API

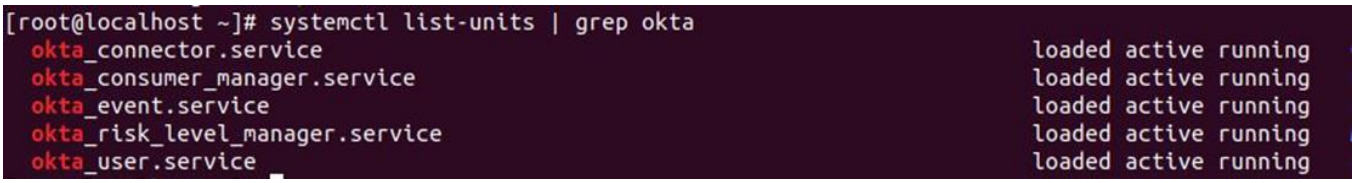
rose_api	FQDN of the Forcepoint ROSE API
mds1_api	FQDN of the Forcepoint Master Data Service API
risk_level_groups	List containing the names of the 5 risk level groups available in Okta
risk_level_manager_portal	FQDN of the Risk Level Manager used by its services
connector_portal	FQDN of the Connector Service API
user_app_portal	FQDN of the User Service API
org_token	API token created in Okta as described in step 1 of this document
org_name	Organization name defined in Okta
kafka_broker	FQDN of the Kafka broker, the same machine where the Kafka host of Forcepoint Behavioral Analytics is running
kafka_topic_name	Topic name used by Forcepoint Behavioral Analytics to announce the risk levels, do NOT change
kafka_group_name	Group name used by Forcepoint Behavioral Analytics to announce the risk levels, do NOT change
client-ca.cer	Path to the certificate of the CA issuer of the certificate used by the Kafka consumer
client.cer	Path to the certificate used by the Kafka consumer
client.key	Path to the private key of the certificate used by the Kafka consumer
key_store_pass	Password of the keystore containing the key pairs used by Kafka

To set up the Risk Level Manager, proceed as follows:

1. Login via SSH to the **RLM-host** and copy the **fp-fba-connector-okta-v1.tar.gz** file into **/root** folder of the machine that will host the Risk Level Manager
2. Decompress the file using the command **tar -zxvf fp-fba-connector-okta-v1.tar.gz**
3. Go into the **/root/installer** folder and edit the **configs.json** file so that the parameters match the current setup of Forcepoint Behavioral Analytics and Okta
4. Make sure the installer.sh file is executable using the command **sudo chmod a+x installer.sh**
5. Install the Risk Level Manager using the command **sudo ./installer.sh**

   The installer script will read the **configs.json** file, move the services to the **application_directory** and create all services. The **configs.json** file will then be moved to **application_directory**: do not change the location of the file.
6. Once the installation is completed, reboot the machine
7. After reboot is completed, log into the machine and verify all 5 services of the Risk Level

Manager are running with the command **systemctl list-units | grep okta**



If all services are running, the Risk Level Manager is operating normally and the interoperability between Forcepoint Behavioral Analytics and Okta is completed: login events will then be visible in the Forcepoint Behavioral Analytics dashboard as soon as users authenticate through Okta. Group membership and policies will be adjusted dynamically as soon as a new risk level is calculated.

Troubleshooting

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

Validate the prerequisites

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

- Check the versions of Forcepoint Behavioral Analytics and Okta in use are listed as compatible

  Forcepoint Behavioral Analytics 3.1.0
  Okta 2020.02.01

- Verify the integration component correctly operates on a CentOS 7.x machine with at least 2 GB RAM and 20 GB free storage
- User must be root to run the installer.sh
- Check the user can download the file with the below command:

  wget --content-disposition https://frcpnt.com/fba-okta-latest

- Verify the Kafka machine's IP address and domain name are configured in /etc/hosts

  For example, if the IP address of kafka machine is 10.1.1.0 and domain name is kafka.machine.com, then the configuration of this machine in /etc/hosts would be as below:

  10.1.1.0 kafka.machine.com

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 **Risk Level Manager** has network connectivity to Forcepoint Behavioral Analytics: execute the following command on the **RLM-host** machine:

  ping -c 2 example.url

  Replacing the example URL/IP address with the current one used inside the Forcepoint Behavioral Analytics setup (it is stored inside the file **configs.json**, parameter "fba_api" of the RLM). Check the result is similar to below:

  PING example.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 Risk Level Manager has network connectivity to the Kafka bus of Forcepoint Behavioral

Analytics: execute the following command on the **RLM-host** machine:

ping -c 2 example.url

Replacing the example URL/IP address with the current one used inside the Forcepoint Behavioral Analytics setup (it is stored inside the file configs.json, parameter "kafka_broker" of the RLM). Check the result is similar to below:

PING example.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

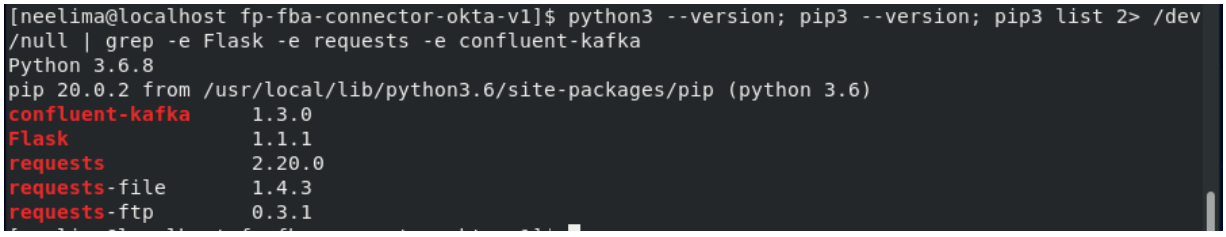
Check dependencies are installed

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

- Check all dependencies are installed: execute the following command on RLM-host:

  python3 --version; pip3 --version; pip3 list 2> /dev/null | grep -e Flask -e requests -e confluent-kafka

  and check the result is similar to below:



  Note: The software versions may change depending on the last version of pip.

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:

  systemctl list-units| grep okta

  and check the result is similar to below:

```
[root@localhost neelima]# systemctl list-units| grep okta
okta_connector.service
    loaded active running   Connects risk level publisher with risk level consumer

okta_consumer_manager.service
    loaded active running   Pull risk levels from kafka bus and send it to the connector

okta_event.service
    loaded active running   Pulls events from Okta and send them to FBA

okta_risk_level_manager.service
    loaded active running   Manage user's group membership

okta_user.service
    loaded active running   perform user validation and entity creation
```