



Forcepoint Behavioral Analytics and Azure Active Directory

Integration Guide

Dlo Bagari

Mattia Maggioli

04 September 2020

Public

Summary2

Caveats.....2

Implementation3

Step 1 – Register an App inside Azure Active Directory4

Step 2 –Groups into Azure Active Directory6

Step 3 – Configure a Kafka consumer8

Step 4 – Configure the Risk Level Manager 10

Appendix A – Change message retention policies of the Risk Level Manager13

Troubleshooting 14

Version	Date	Author	Notes
0.1	21 November 2019	Mattia Maggioli	First draft
0.2	26 November 2019	Jonathan Knepher, Audra Simons	Review
0.3	15 January 2020	Dlo Bagari	Added auto session termination
0.4	20 January 2020	Dlo Bagari	Updated package name
0.5	23 March 2020	Neelima Rai	Added troubleshooting chapter
0.6	04 September 2020	Mattia Maggioli	Minor updates

Summary

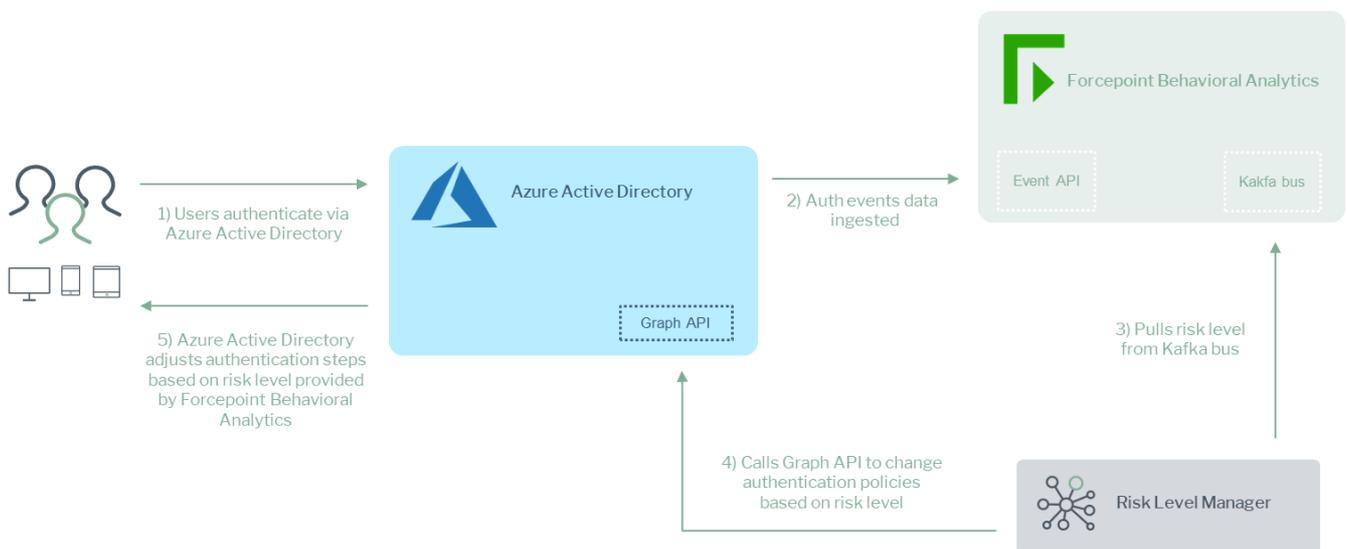
This guide provides step by step instructions to configure Forcepoint Behavioral Analytics and Azure Active Directory to pass risk scores and login / event information, and to adjust authentication policies accordingly.

The code and instructions provided enable system administrators to automatically:

- ▶ Export authentication events from Azure into Forcepoint Behavioral Analytics
- ▶ Provide the risk level calculated by Forcepoint Behavioral Analytics for each user to Azure
- ▶ Adjust authentication policies applied by Azure to users based on their risk level
- ▶ Upon risk level increase, terminates the active sessions of the user who will have to log in again 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 Azure 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
- ▶ Microsoft Graph API 1.0 – November 2019

This interoperability uses the “Graph API” of Microsoft, 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-azure-latest>

- ▶ fp-fba-connector-azure-v1.tar.gz

The **fp-fba-connector-azure-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 Azure Active Directory:

- ▶ **Event Service:** handles the access token need to communicate to Azure AD, extracts authentication logs and transforms data into the format used by Forcepoint Behavioral Analytics to ingest data
- ▶ **User Service:** creates Entities in Forcepoint Behavioral Analytics so that Identifiers and Aliases for Azure AD users are setup appropriately
- ▶ **Consumer Service:** obtains risk level from Forcepoint Behavioral Analytics, confirms the risk level is related to a known Azure AD user before triggering the change in authentication policies into Azure AD
- ▶ **FBA Service:** handles and verifies the exchange of data between the Kafka bus of Forcepoint Behavioral Analytics and the other services of the Risk Level Manager
- ▶ **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 of free storage, the instructions provided in this document are based on this operating system and the following packages

- ▶ Java 8
- ▶ Python 3 and modules: requests, flask, confluent_kafka, PyYAML, pytz, python-dateutil, adal

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

```
sudo pip3 install flask
sudo pip3 install requests
sudo pip3 install confluent-kafka
sudo pip3 install PyYAML
sudo pip3 install pytz
sudo pip3 install python-dateutil
sudo pip3 install adal
```

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

Step 1 – Register an App inside Azure Active Directory

In order to use Microsoft Graph API we need to register an App inside Azure Active Directory: this will be used to obtain security tokens used by the Risk Level Manager to exchange information via Microsoft Graph API.

1. Login to the Azure portal with an account with administrative access
2. Click on **Azure Active Directory**
3. In the **Manage** section of the side menu click **App registrations > New registration**
4. Enter a name for the new App registration and select **Accounts in any organization directory**

* Name

The user-facing display name for this application (this can be changed later).

Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (bzforcepoint only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

5. Click **Register** and save the **Application (client) ID** in a secure location, as it will be needed to complete the steps in the next chapter of this document
6. Click **View API permissions > Add a permission > Microsoft Graph**
7. Click **Delegated permissions** and select
 - Directory > DirectoryAccessAsUser.All
 - Directory > Directory.Read.All
 - Directory > Directory.ReadWrite.All
 - User > User.Read

Once done click **Application permissions** and select

- Application > Application.Read.All
- AuditLog > AuditLog.Read.All
- Directory > Directory.Read.All
- Directory > Directory.ReadWrite.All
- IdentityRiskyUser > IdentityRiskyUser.Read.All
- User > User.Export.All
- User > User.Read.All
- User > User.ReadWrite.All

then click **Add Permissions**. In the next step, click **Grant admin consent for <organization name>**

8. From the **Manage** section of the side menu click **Certificates & secrets > New client secret**
9. Enter a description for this client secret, set an expiration date and click **Add**
10. Once the secret key is generated, store it in a secure location since it will be needed for the configuration of the Risk Level Manager in the next chapters of this document

Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value
azure_fba	11/11/2020	uCO*****

Step 2 –Groups into Azure Active Directory

Authentication steps applied to users authenticating through Azure Active Directory are defined as **Security policies** configured into **Groups**.

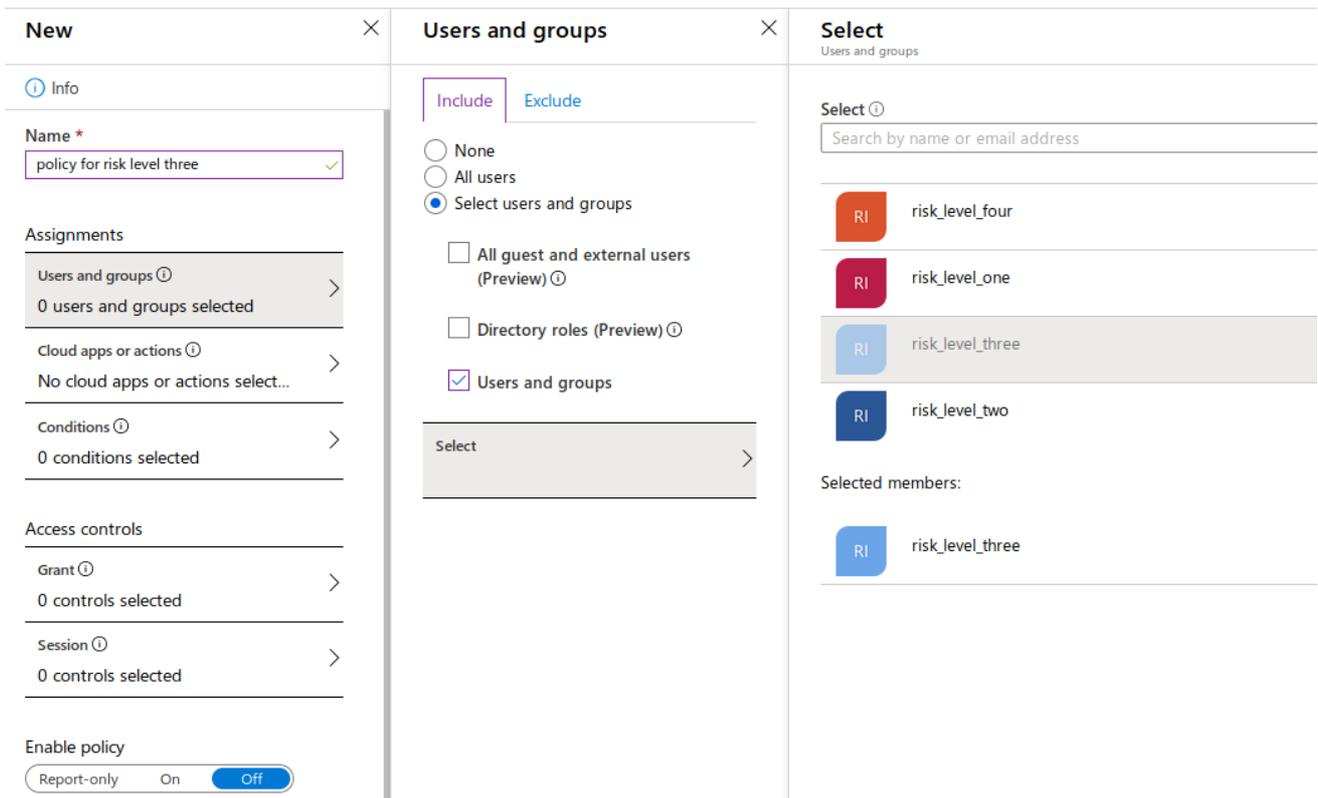
A user authenticating through Azure Active Directory will be challenged according to the policies configured based on the user’s group membership. A typical group and policy configuration is as follows:

- ▶ Users with risk_level 1 and 2 (low risk) are assigned to groups with standard authentication policies (e.g. username and password)
- ▶ Users with risk_level 3 and 4 (medium risk) are opposed more complex authentication policies (e.g. username and password, then multi-factor authentication)
- ▶ Users with risk_level 5 are denied authentication

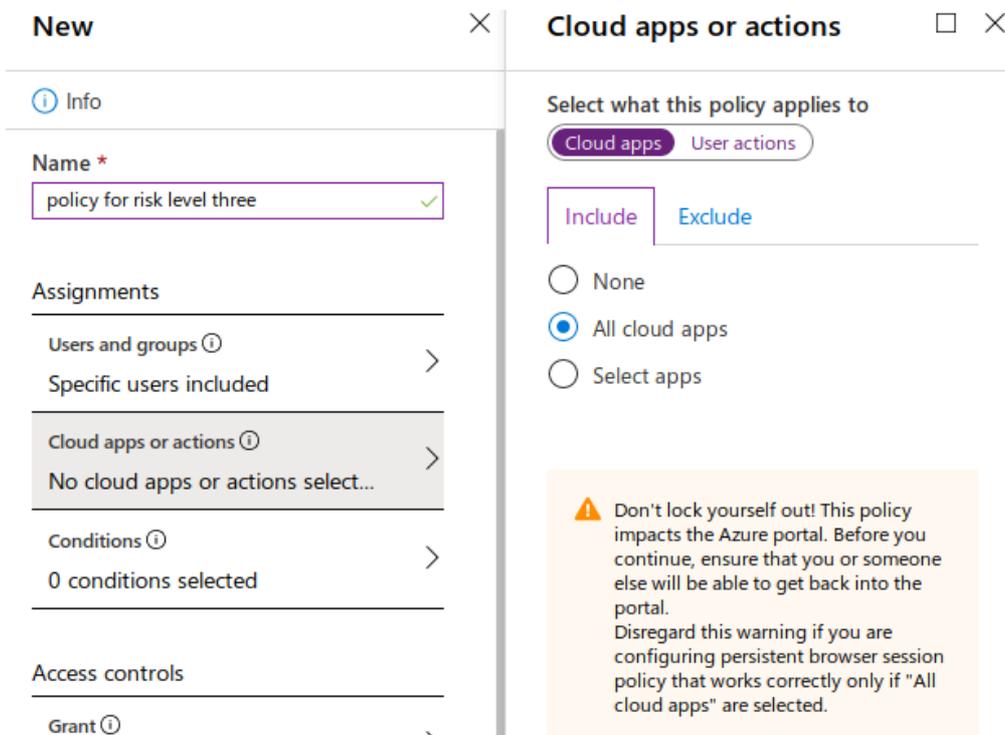
If multiple user groups with existing policies are already configured into Azure Active Directory, then skip this part and go to **Step 3**.

In the following example, we create a new user group for users with risk_level 3 and enable **Multi-factor authentication** for **All cloud apps**:

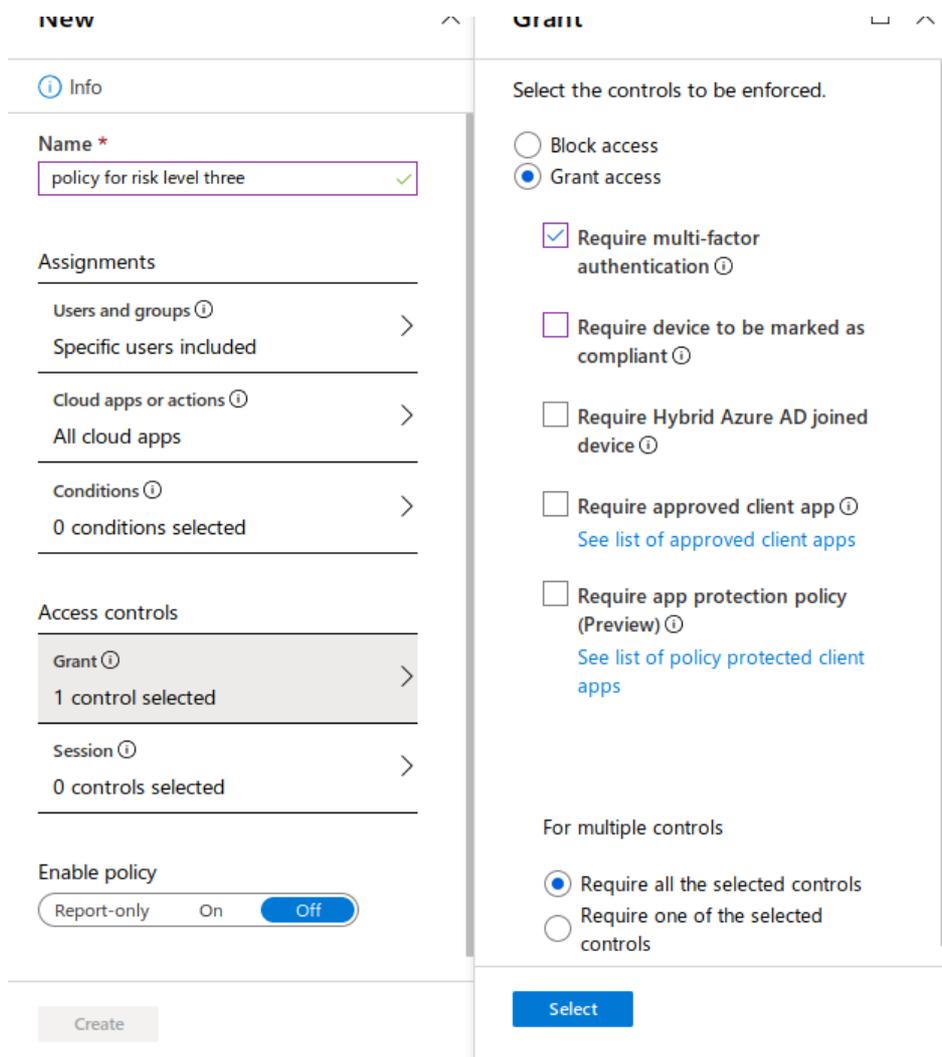
1. Go to **Azure Active Directory > Groups > New group**
2. Select “Security” as **Group type** and enter “risk_level_three” as group name, leave **Membership type** to “Assigned” and click **Create**
3. Go to **Azure Active Directory > Security > Conditional Access > New policy**
4. Name your policy and assign it to the **risk_level_three** group, then click **Select > Done**



5. Click **Cloud apps or actions** and select **All cloud apps** inside the **Include** tab, then click **Done**



6. Click **Grant > Grant access** and tick **Require multi-factor authentication**, then click **Select**



7. Under **Enable policy** make sure to click **On** before clicking **Create**, otherwise the new policy will not be enforced

Multi-factor authentication must be configured 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. Multi-factor authentication can be configured by clicking **Azure Active Directory > Security > MFA**.

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 sending this information as metadata to the consumers. In the example above the hostname of the listener **kafka-host** is not a FQDN hence under normal circumstances the consumer would not be able to connect to the Kafka broker because the hostname cannot be resolved into an IP address.

If that's the case, make sure an entry into the DNS server is created to resolve the hostname of the Kafka host, or manually add an entry to the **host** file of the machine where the Risk Level Manager will be installed into, 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
```

- 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

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 **settings.yml**:

```

app_id: 0123456789abcdefghijklmnpqrstuvz
app_secret: uCOB2DB7T-N20j2nc(8hc239f209jy[
tenant_name: organization-name.onmicrosoft.com
resource: https://graph.microsoft.com
logs_endpoint: https://graph.microsoft.com/v1.0/auditLogs/signIns
application_directory: /var/azure_fba
logs_directory: /var/azure_fba/logs
microsoft_login: https://login.microsoftonline.com
kafka_directory: /usr/local/kafka
logs_topic_name: AZURE-FBA-LOGS
risk_level_topic_name: FBA-RISK-LEVEL
kafka_bootstrap_server: localhost:9092
fba_consumer_group_name: fba-service
user_consumer_group_name: user-service
risk_level_consumer_group: risk-level-service
user_service_ip: 127.0.0.1
user_service_port: 5000
fba_events_end_point: fba-host.organization.net
rose_api_host_name: rose-host.organization.net
mdsl_api_host_name: mdsl-host.organization.net
fba_kafka_bootstrap_server: kafka-host.organization.net:9093
fba_kafka_topic_name: ENTITY_RISK_LEVEL
fba_kafka_consumer_group_name: azure_fba_consumer
ssl_ca_location: /var/azure_fba/key_store/client-ca.cer
ssl_certificate_location: /var/azure_fba/key_store/client.cer
ssl_key_location: /var/azure_fba/key_store/client.key
key_store_pass: changeme
risk_level_groups_name:
  risk_level_one: risk_level_one
  risk_level_two: risk_level_two
  risk_level_three: risk_level_three
  risk_level_four: risk_level_four
  risk_level_five: risk_level_five
    
```

The following table provides a description of every parameter in the **settings.yml** file:

Parameter	Description	Requires to be changed
app_id	Application ID, issued by Azure Active Directory when registering a new App	YES
app_secret	Secret key issued by Azure Active Directory when registering a new App	YES

Forcepoint Behavioral Analytics and Azure Active Directory – Integration Guide

tenant_name	Tenant name from Azure Active Directory > Custom domain names > Name	YES
resource	URL of Microsoft Graph API	NO
logs_endpoint	Microsoft Graph API endpoint to extract authentication events	NO
application_directory	Directory where the Risk Level Manager is stored, by default /var/azure_fba	NO
kafka_directory	Directory where the Kafka server of the Risk Level Manager is stored (not the Kafka server used by Forcepoint Behavioral Analytics), by default /usr/local/kafka	NO
logs_topic_name	Topic name used inside the Kafka server of the Risk Level Manager to publish messages containing Azure events data	NO
risk_level_topic_name	Topic name used by the Consumer Service to publish risk level messages from the Kafka server of Forcepoint Behavioral Analytics into the Kafka server used by the Risk Level Manager	NO
risk_level_consumer_group	Group name used by the Consumer Service to publish risk level messages from the Kafka server of Forcepoint Behavioral Analytics into the Kafka server used by the Risk Level Manager	NO
kafka_bootstrap_server	FQDN and port of the Kafka broker used by the Risk Level Manager	NO
fba_consumer_group_name risk_level_consumer_group	Group names for Kafka consumers of messages related to risk level information	NO
user_service_ip	IPv4 address of the User service	NO
user_service_port	Port used by the User service	NO
fba_events_end_point	FQDN of the Forcepoint Streaming Ingest Public API	YES
rose_api_host_name	FQDN of the Forcepoint ROSE API	YES
mds1_api_host_name	FQDN of the Forcepoint Master Data Service API	YES
risk_level_groups_name	List containing the names of the 5 risk level groups available in Azure Active Directory	YES
fba_kafka_bootstrap_server	FQDN of the Kafka broker, the same machine where the Kafka host of Forcepoint Behavioral Analytics is running	YES
fba_kafka_topic_name	Topic name used by Forcepoint Behavioral Analytics to announce the risk levels, do NOT change	NO
fba_kafka_consumer_group_name	Group name used by Forcepoint Behavioral Analytics to announce the risk levels	NO
ssl_ca_location	Path to the certificate of the CA issuer of the certificate used by the Kafka consumer	YES
ssl_certificate_location	Path to the certificate of the certificate used by the Kafka consumer	YES
ssl_key_location	Path to the private key of the certificate used by the Kafka consumer	YES
key_store_pass	Password of the keystore containing the key pairs used by Kafka	YES

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

1. Login via SSH to the **RLM-host** and copy the **fp-fba-connector-azure-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-azure-v1.tar.gz** Go into the **/root/fp-fba-connector-azure-v1/** folder and edit the **settings.yml** file so that the parameters match the current setup of Forcepoint Behavioral Analytics and Azure Active Directory, change only the entries that require to be changed according to the table above
3. Make sure the installer.sh file is executable using the command **sudo chmod a+x azure_afba_installer.sh**
4. Install the Risk Level Manager using the command **sudo ./azure_afba_installer.sh**

The installer script will install the dependencies, read the **settings.yml** file, move the code to the relevant directories, create and enable at boot all services:

```
azure_fba_consumer.service  
azure_event.service  
azure_risk_level.service  
azure_user.service  
azure_fba.service  
kafka.service  
zookeeper.service
```

5. Once the installation is completed move the SSL key and certificates generated during step 3 into **/var/azure_fba/key_store/**
6. Reboot the RLM-host machine then log into the machine and verify all services of the Risk Level Manager are running with the commands

systemctl list-units | grep -e azure -e zookeeper -e kafka

```
[root@localhost ~]# systemctl list-units | grep -e azure -e zookeeper -e kafka  
azure_event.service      loaded active running  
azure_fba.service        loaded active running  
azure_fba_consumer.service loaded active running  
azure_risk_level.service loaded active running  
azure_user.service       loaded active running  
kafka.service            loaded active running  
zookeeper.service        loaded active running
```

If all services are running, the Risk Level Manager is operating normally and the interoperability between Forcepoint Behavioral Analytics and Azure Active Directory is completed: login events will then be

visible in the Forcepoint Behavioral Analytics dashboard as soon as users authenticate into Azure. Group membership and policies will be adjusted dynamically as soon as a new risk level is calculated and, in case of risk level increase, the user sessions will be terminated and the user will have to log in again according to the policies assigned to the group associated to the higher risk level.

Appendix A – Change message retention policies of the Risk Level Manager

The components of the Risk Level Manager exchange messages using a Kafka server installed inside the RLM-host by the **azure_fba_installer.sh** script.

By default, the Kafka server inside the RLM-host is configured so that

- ▶ Broker address is **localhost**
- ▶ Messages expire after **24hrs** from publishing
- ▶ Maximum disk space available for storing messages is **1GB**, older messages will be removed automatically if space quota is reached

The parameters above can be customized by the system administrator by changing the values of the following lines into the **server.properties** file stored into **/usr/local/kafka** inside the RLM-host machine:

```
log.cleaner.enable=true  
log.cleanup.policy=delete  
log.retention.hours=24  
log.retention.bytes=1073741824
```

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 Azure Microsoft Graph API in use are listed as compatible

Forcepoint Behavioral Analytics 3.1.0
Microsoft Graph API 1.0 – March 2020

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

```
yum install wget  
wget --content-disposition https://frcpnt.com/fba-azure-latest
```

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

e.g. if the IP address of kafka machine is 10.1.1.0 and domain name is **kafka.machine.com**, then the entry 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-fba.url
```

replacing the example URL/IP address with the current one used. Check the result is similar to below:

```
PING example-fba.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-kafka.url
```

replacing the example URL/IP address with the current one used. Check the result is similar to below:

```
PING example-kafka.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 -e PyYAML -e pytz -e python-dateutil -e adal
```

and check the result is similar to below:

```
[root@localhost ~]# python3 --version; pip3 --version; pip3 list 2> /dev/null | grep -e Flask -e requests -e confluent-kafka -e PyYAML  
-e pytz -e python-dateutil -e adal  
Python 3.6.8  
pip 9.0.3 from /usr/lib/python3.6/site-packages (python 3.6)  
adal (1.2.2)  
confluent-kafka (1.3.0)  
Flask (1.1.1)  
python-dateutil (2.8.1)  
pytz (2019.3)  
PyYAML (5.3)  
requests (2.23.0)
```

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 -e azure -e zookeeper -e kafka
```

and check the result is similar to below:

Forcepoint Behavioral Analytics and Azure Active Directory – Integration Guide

```
[root@localhost ~]# systemctl list-units | grep -e azure -e zookeeper -e kafka
azure_event.service                                loaded active running
  Connects with Microsoft Graph and pull logs and process them
azure_fba.service                                  loaded active running
  run FBA service
azure_fba_consumer.service                         loaded active running
  Consumer Service, pull risk level and create risk level objects
azure_risk_level.service                           loaded active running
  Run azure risk level manager service
azure_user.service                                 loaded active running
  User service. create entities, validate user and change users group
kafka.service                                       loaded active running
  Starts the kafka server Service
zookeeper.service                                  loaded active running
  Starts the zookeeper Service
```



© 2020 Forcepoint

Forcepoint and the FORCEPOINT logo are trademarks of Forcepoint.
All other trademarks used in this document are the property of their respective owners.

