Detailed Explanation of the Akamai Access Group Management Script

# Overview

The Akamai Access Group Management Script is a Python-based tool designed to automate the management of access groups within the Akamai platform. It checks whether a specified group exists under a particular parent group, and if not, it creates the group along with regional subgroups (APAC, EMEA, AMER). Each regional subgroup further includes automatically created `Prod` and `PreProd` subgroups. The script is environment-aware, allowing it to function differently based on whether it is running in a development (DEV) or production (PROD) environment. It also offers flexibility by allowing the user to specify the parent group by either name or ID.

## 1. main Function

def main():

The `main` function coordinates the script's overall flow, handling parameter parsing, environment setup, and function calls.

### Parameters:

None directly, but it parses command-line arguments.

### Key Steps:

- Parses command-line arguments, including `group\_name\_segment`, `parent\_group\_name`, `parent\_group\_id`, `env`, `edgerc\_file`, and `section\_name`.  
- Trims any leading or trailing spaces from `group\_name\_segment`, `group\_prefix`, and `parent\_group\_name`.  
- Sets the default `parent\_group\_id` based on the `env` parameter (`DEV` or `PROD`).

## 2. setup\_session Function

def setup\_session(edgerc\_file, section\_name):

This function establishes a session with the Akamai API using credentials stored in a `.edgerc` file.

### Parameters:

- edgerc\_file: Path to the `.edgerc` file containing Akamai API credentials.  
- section\_name: The section name within the `.edgerc` file to use for retrieving credentials.

### Returns:

A session object configured with the necessary authentication headers, and the base URL for the Akamai API.

### Key Steps:

- Reads the `.edgerc` file.  
- Extracts the host, client token, client secret, and access token from the specified section.  
- Sets up the session with `EdgeGridAuth` for secure API communication.

## 3. check\_group\_exists Function

def check\_group\_exists(session, base\_url, group\_name, parent\_group\_name=None, parent\_group\_id=None):

This function coordinates the search for the parent group and checks for the existence of the specified group under it.

### Parameters:

- session: The session object configured for API communication.  
- base\_url: The base URL of the Akamai API.  
- group\_name: The name of the group to check for.  
- parent\_group\_name: The name of the parent group.  
- parent\_group\_id: The ID of the parent group.

### Returns:

A tuple `(group\_exists, existing\_group\_id, resolved\_parent\_group\_id)` indicating whether the group exists, the ID of the existing group, and the ID of the resolved parent group.

### Key Steps:

- Calls the Akamai API to retrieve the list of groups.  
- Uses `find\_group\_by\_name\_or\_id` to locate the parent group.  
- Calls `check\_group\_exists\_under\_parent` to check for the existence of the specified group.  
- Returns the results of the check.

## 4. find\_group\_by\_name\_or\_id Function

def find\_group\_by\_name\_or\_id(groups, parent\_group\_name=None, parent\_group\_id=None):

This function recursively searches through the group hierarchy to find a parent group based on either its name or ID.

### Parameters:

- groups: The list of groups and subgroups to search within.  
- parent\_group\_name: The name of the parent group to search for.  
- parent\_group\_id: The numeric ID of the parent group to search for.

### Returns:

The parent group object if found, otherwise `None`.

### Key Steps:

- Iterates through the provided groups.  
- If a group matches the `parent\_group\_name` or `parent\_group\_id`, it returns that group.  
- Recursively searches through subgroups if the group is not found at the current level.

## 5. check\_group\_exists\_under\_parent Function

def check\_group\_exists\_under\_parent(parent\_group, group\_name):

This function checks if a specific group exists under the identified parent group.

### Parameters:

- parent\_group: The parent group object under which to search.  
- group\_name: The name of the group to check for.

### Returns:

A tuple `(True, groupId)` if the group exists, or `(False, None)` if it does not.

### Key Steps:

- Iterates through the subgroups of the parent group.  
- If a subgroup matches the `group\_name`, it returns `True` along with the group's ID.  
- If no match is found, it returns `False`.

## 6. create\_group Function

def create\_group(session, base\_url, group\_name, parent\_group\_id):

This function creates a new group under the specified parent group.

### Parameters:

- session: The session object configured for API communication.  
- base\_url: The base URL of the Akamai API.  
- group\_name: The name of the group to create.  
- parent\_group\_id: The ID of the parent group under which the new group will be created.

### Key Steps:

- Constructs the API endpoint URL using the `parent\_group\_id`.  
- Sends a POST request to the Akamai API to create the group.  
- Prints the result, including the new group's ID if the creation is successful.

## 7. ensure\_subgroups Function

def ensure\_subgroups(session, base\_url, parent\_group\_id, region):

This function ensures that regional subgroups (`APAC`, `EMEA`, `AMER`) and their respective `Prod` and `PreProd` subgroups are created under the specified parent group.

### Parameters:

- session: The session object configured for API communication.  
- base\_url: The base URL of the Akamai API.  
- parent\_group\_id: The ID of the parent group under which the regional subgroups will be created.  
- region: The region name to append to the subgroup names.

### Key Steps:

- Constructs the regional group name and checks if it exists.  
- Creates the regional group if it does not exist.  
- Ensures that `Prod` and `PreProd` subgroups are created under the regional group.

**Title:** Detailed Explanation of the Akamai Access Group Management Script

**Overview:**

The Akamai Access Group Management Script is a Python-based tool designed to automate the management of access groups within the Akamai platform. It checks whether a specified group exists under a particular parent group, and if not, it creates the group along with regional subgroups (APAC, EMEA, AMER). Each regional subgroup further includes automatically created Prod and PreProd subgroups. The script is environment-aware, allowing it to function differently based on whether it is running in a development (DEV) or production (PROD) environment. It also offers flexibility by allowing the user to specify the parent group by either name or ID.

**Script Components in Logical Flow Order:**

**1. main Function**

python

Copy code

def main():

The main function coordinates the script's overall flow, handling parameter parsing, environment setup, and function calls.

**Parameters:**

* **group\_name\_segment (required):** The name segment for the group to create.
* **--group\_prefix (optional):** The prefix for the group name. Default is KSSS-DDoS-KSD.
* **--regions (optional):** Specifies the regions for which subgroups will be created (APAC, EMEA, AMER). Default is all three regions.
* **--parent\_group\_name (optional):** The name of the parent group. You must provide either this or --parent\_group\_id.
* **--parent\_group\_id (optional):** The numeric ID of the parent group. You must provide either this or --parent\_group\_name.
* **--env (optional):** The environment (DEV or PROD). Default is DEV. This determines the default parent group ID.
* **--edgerc\_file (optional):** Path to the .edgerc file for Akamai API authentication. Default is ~/.edgerc.
* **--section\_name (optional):** Section name in the .edgerc file. Default is default.

**Key Steps:**

* Parses command-line arguments, including group\_name\_segment, parent\_group\_name, parent\_group\_id, env, edgerc\_file, and section\_name.
* Trims any leading or trailing spaces from group\_name\_segment, group\_prefix, and parent\_group\_name.
* Sets the default parent\_group\_id based on the env parameter (DEV or PROD).

**2. setup\_session Function**

python

Copy code

def setup\_session(edgerc\_file, section\_name):

This function establishes a session with the Akamai API using credentials stored in a .edgerc file.

**Parameters:**

* **edgerc\_file:** Path to the .edgerc file containing Akamai API credentials.
* **section\_name:** The section name within the .edgerc file to use for retrieving credentials.

**Returns:**

* A session object configured with the necessary authentication headers, and the base URL for the Akamai API.

**Key Steps:**

* Reads the .edgerc file.
* Extracts the host, client token, client secret, and access token from the specified section.
* Sets up the session with EdgeGridAuth for secure API communication.

**3. check\_group\_exists Function**

python

Copy code

def check\_group\_exists(session, base\_url, group\_name, parent\_group\_name=None, parent\_group\_id=None):

This function coordinates the search for the parent group and checks for the existence of the specified group under it.

**Parameters:**

* **session:** The session object configured for API communication.
* **base\_url:** The base URL of the Akamai API.
* **group\_name:** The name of the group to check for.
* **parent\_group\_name:** The name of the parent group.
* **parent\_group\_id:** The ID of the parent group.

**Returns:**

* A tuple (group\_exists, existing\_group\_id, resolved\_parent\_group\_id) indicating whether the group exists, the ID of the existing group, and the ID of the resolved parent group.

**Key Steps:**

* Calls the Akamai API to retrieve the list of groups.
* Uses find\_group\_by\_name\_or\_id to locate the parent group.
* Calls check\_group\_exists\_under\_parent to check for the existence of the specified group.
* Returns the results of the check.

**4. find\_group\_by\_name\_or\_id Function**

python

Copy code

def find\_group\_by\_name\_or\_id(groups, parent\_group\_name=None, parent\_group\_id=None):

This function recursively searches through the group hierarchy to find a parent group based on either its name or ID.

**Parameters:**

* **groups:** The list of groups and subgroups to search within.
* **parent\_group\_name:** The name of the parent group to search for.
* **parent\_group\_id:** The numeric ID of the parent group to search for.

**Returns:**

* The parent group object if found, otherwise None.

**Key Steps:**

* Iterates through the provided groups.
* If a group matches the parent\_group\_name or parent\_group\_id, it returns that group.
* Recursively searches through subgroups if the group is not found at the current level.

**5. check\_group\_exists\_under\_parent Function**

python

Copy code

def check\_group\_exists\_under\_parent(parent\_group, group\_name):

This function checks if a specific group exists under the identified parent group.

**Parameters:**

* **parent\_group:** The parent group object under which to search.
* **group\_name:** The name of the group to check for.

**Returns:**

* A tuple (True, groupId) if the group exists, or (False, None) if it does not.

**Key Steps:**

* Iterates through the subgroups of the parent group.
* If a subgroup matches the group\_name, it returns True along with the group's ID.
* If no match is found, it returns False.

**6. create\_group Function**

python

Copy code

def create\_group(session, base\_url, group\_name, parent\_group\_id):

This function creates a new group under the specified parent group.

**Parameters:**

* **session:** The session object configured for API communication.
* **base\_url:** The base URL of the Akamai API.
* **group\_name:** The name of the group to create.
* **parent\_group\_id:** The ID of the parent group under which the new group will be created.

**Key Steps:**

* Constructs the API endpoint URL using the parent\_group\_id.
* Sends a POST request to the Akamai API to create the group.
* Prints the result, including the new group's ID if the creation is successful.

**7. ensure\_subgroups Function**

python

Copy code

def ensure\_subgroups(session, base\_url, parent\_group\_id, region):

This function ensures that regional subgroups (APAC, EMEA, AMER) and their respective Prod and PreProd subgroups are created under the specified parent group.

**Parameters:**

* **session:** The session object configured for API communication.
* **base\_url:** The base URL of the Akamai API.
* **parent\_group\_id:** The ID of the parent group under which the regional subgroups will be created.
* **region:** The region name to append to the subgroup names.

**Key Steps:**

* Constructs the regional group name and checks if it exists.
* Creates the regional group if it does not exist.
* Ensures that Prod and PreProd subgroups are created under the regional group.

**Usage:**

Here are some examples of how to execute the script:

1. **Create a group in the DEV environment with a default parent group ID:**

bash

Copy code

python script.py "TestGroup" --env=DEV

1. **Create a group in the PROD environment with a specific parent group ID:**

bash

Copy code

python script.py "ProdGroup" --env=PROD --parent\_group\_id=1234567

1. **Create a group using a parent group name:**

bash

Copy code

python script.py "NamedGroup" --parent\_group\_name="Main Street Corporation"

1. **Create a group with a custom prefix in the DEV environment:**

bash

Copy code

python script.py "CustomSegment" --group\_prefix="ABCD-DDoS-KSD" --env=DEV

1. **Create a group and specify regions for subgroups:**

bash

Copy code

python script.py "RegionalGroup" --regions="APAC,EMEA" --env=DEV

1. **Specify a custom .edgerc file and section name:**

bash

Copy code

python script.py "CustomEdgercGroup" --parent\_group\_id=199633 --edgerc\_file="/path/to/.edgerc" --section\_name="custom\_section"

**Summary:**

This document provides a detailed explanation of the Akamai Access Group Management Script. The script is designed to automate the creation and management of access groups within the Akamai platform, ensuring that necessary groups and subgroups are created under specified parent groups. It is environment-aware, functioning differently based on whether it is running in a development or production environment.

The main function orchestrates the script's flow, setting up the necessary session and checking for the existence of groups. If a group does not exist, the script creates it, along with regional subgroups (APAC, EMEA, AMER) and their respective Prod and PreProd subgroups. The script is highly flexible, allowing for group creation based on either a parent group name or ID.

This document is structured to follow the logical flow of the script, starting with the main function and proceeding through the various functions that handle session setup, group existence checks, group creation, and subgroup management. Each function is explained in detail, with descriptions of its parameters, return values, and key steps.