Detailed Explanation of the Akamai Access Group Management Script

# Overview

The Akamai Access Group Management Script is designed to help manage access groups within the Akamai platform. The script checks whether a specified group exists under a particular parent group, and if not, it creates the group. 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 provides flexibility in specifying the parent group by either name or ID.

# Script Components

## 1. setup\_session Function

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

This function sets up a session with the Akamai API using the 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 the EdgeGridAuth for secure API communication.

## 2. 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.

## 3. 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.

## 4. 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.

## 5. 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.

## 6. 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, parent\_group\_name, parent\_group\_id, env, edgerc\_file, and section\_name.  
- Trims any leading or trailing spaces from group\_name and parent\_group\_name.  
- Sets the default parent\_group\_id based on the env parameter (DEV or PROD).  
- Prioritizes parent\_group\_id if provided, otherwise resolves the parent\_group\_id from parent\_group\_name.  
- Calls check\_group\_exists to see if the group already exists under the parent group.  
- If the group does not exist, it calls create\_group to create it.

# Parameters and Their Usage

## group\_name (required):

The name of the group to create.

## --parent\_group\_name (optional):

The name of the parent group. Use this if the parent group is identified by name rather than ID.

## --parent\_group\_id (optional):

The numeric ID of the parent group. Takes precedence over parent\_group\_name if both are provided.

## --env (optional):

The environment where the script is running. Defaults to DEV, which uses parent\_group\_id = 199633. If set to PROD, it uses parent\_group\_id = 1234567.

## --edgerc\_file (optional):

The path to the .edgerc file containing Akamai API credentials. Default is ~/.edgerc.

## --section\_name (optional):

The section name in the .edgerc file for credentials. Default is default.

# Usage Examples

## Create a group in the DEV environment with the default parent group ID:

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

## Create a group in the PROD environment with the default parent group ID:

python script.py "NewGroupName" --env=PROD

## Create a group in the DEV environment using a specific parent group ID:

python script.py "NewGroupName" --env=DEV --parent\_group\_id=199633

## Create a group in the PROD environment using a specific parent group ID:

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

## Create a group using a parent group name in the DEV environment:

python script.py "NewGroupName" --parent\_group\_name="Main Street Corporation" --env=DEV

## Create a group using a parent group name in the PROD environment:

python script.py "NewGroupName" --parent\_group\_name="Main Street Corporation" --env=PROD

## Create a group using a parent group ID (ignores --parent\_group\_name if both are provided):

python script.py "NewGroupName" --parent\_group\_id=199633 --parent\_group\_name="Main Street Corporation"

## Specify a custom .edgerc file and section name:

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