Complete solution, including the updated .edgerc file with both default and prod environments, the Python script, and the requirements.txt file.

**.edgerc File**

This file should be placed in your home directory (~/.edgerc).

ini

Copy code

[default]

client\_secret = your\_default\_client\_secret

host = your\_default\_akamai\_host

access\_token = your\_default\_access\_token

client\_token = your\_default\_client\_token

[prod]

client\_secret = your\_prod\_client\_secret

host = your\_prod\_akamai\_host

access\_token = your\_prod\_access\_token

client\_token = your\_prod\_client\_token

* Replace your\_default\_client\_secret, your\_default\_akamai\_host, your\_default\_access\_token, and your\_default\_client\_token with your actual credentials for the default environment.
* Similarly, replace the values under the [prod] section with your production environment credentials.

**Python Script: akamai\_access\_group.py**

python

Copy code

import os

import requests

from akamai.edgegrid import EdgeGridAuth, EdgeRc

# Load credentials from the .edgerc file

edgerc = EdgeRc(os.path.expanduser("~/.edgerc"))

# Choose environment: "default" or "prod"

section = "default" # Change to "prod" when using production environment

# Set up the base URL and session with credentials

base\_url = f"https://{edgerc.get(section, 'host')}/"

session = requests.Session()

session.auth = EdgeGridAuth.from\_edgerc(edgerc, section)

def check\_access\_group\_exists(session, base\_url, group\_name):

"""

Check if an access group with the given name already exists.

Parameters:

- session: The authenticated session object.

- base\_url: The base URL for the Akamai API.

- group\_name: The name of the access group to check.

Returns:

- True if the group exists, False otherwise.

"""

endpoint = f"{base\_url}identity/v1/groups"

response = session.get(endpoint)

if response.status\_code == 200:

groups = response.json()

for group in groups['groups']:

if group['groupName'] == group\_name:

return True

else:

print(f"Error fetching access groups: {response.status\_code}")

return False

def create\_access\_group(session, base\_url, group\_name, description):

"""

Create a new access group if it doesn't already exist.

Parameters:

- session: The authenticated session object.

- base\_url: The base URL for the Akamai API.

- group\_name: The name of the access group to create.

- description: A description for the new access group.

"""

if not check\_access\_group\_exists(session, base\_url, group\_name):

endpoint = f"{base\_url}identity/v1/groups"

payload = {

"groupName": group\_name,

"description": description

}

response = session.post(endpoint, json=payload)

if response.status\_code == 201:

print(f"Access group '{group\_name}' created successfully.")

else:

print(f"Failed to create access group. Status code: {response.status\_code}")

print(f"Response: {response.text}")

else:

print(f"Access group '{group\_name}' already exists.")

# Define the group name and description

group\_name = "NewAccessGroup"

description = "This is a description for the new access group."

# Create the access group

create\_access\_group(session, base\_url, group\_name, description)

**requirements.txt**

Include this file in the same directory as your Python script. It specifies the dependencies needed for the script.

text

Copy code

requests==2.31.0

akamai-edgegrid==2.2.4

**Steps to Run the Solution**

1. **Create the .edgerc File**:
   * Place the .edgerc file in your home directory (~/.edgerc).
   * Replace the placeholder values with your actual Akamai credentials for both the default and production environments.
2. **Prepare Your Project Directory**:
   * Your directory should look something like this:

bash

Copy code

/my-akamai-project/

│

├── akamai\_access\_group.py

├── requirements.txt

└── ~/.edgerc # This file is stored in your home directory

1. **Install Dependencies**:
   * Navigate to your project directory and install the required Python packages using pip:

bash

Copy code

pip install -r requirements.txt

1. **Run the Script**:
   * Run the script using Python:

bash

Copy code

python akamai\_access\_group.py

* + By default, the script will use the credentials under the [default] section of the .edgerc file. If you want to use the production credentials, change the section = "default" line in the script to section = "prod".

1. **Switching Environments**:
   * To switch between environments (default and production), simply update the section variable in the script as needed:

python

Copy code

section = "prod" # Use this for the production environment

This complete solution allows you to manage Akamai access groups by checking if they already exist and creating them if necessary, all while securely sourcing credentials from a .edgerc file.