

PSK Rotator with Gmail Guide

by Tim Smith, SA – 08/6/2024 – v1.0.0

Overview:

Security is paramount in wireless networks. Regularly rotating the PSK (Pre-Shared Key) for your SSID (Service Set Identifier) helps enhance security by minimizing the risk of unauthorized access. The ExtremeCloud IQ (XIQ) platform allows you to manage PSKs efficiently, ensuring only authorized users can connect to your network.

This guide will walk you through setting up and running the script that rotates the PSK for an XIQ SSID. Whether you're securing a corporate network, a guest network, or any other wireless environment, following these steps will help keep your network safe and your users connected.

This script checks for devices in a mismatched configuration state, retrieves a PSK from a CSV list, updates the SSID in XIQ, and then updates the configuration on the devices. Finally, it emails the updated PSK to a specified list of email addresses using Gmail APIs or SMTP.

Target Audience: Technical

PSK Rotator Use Cases:

- Schedule PSK
- Email new PSK to users
- Secure SSID access

Table of Contents

Overview	1
Target Audience.....	1
PSK Rotator Use Cases:	1
Prerequisites:	3
Scripting Environment Preparation:	3
Information:.....	3
Device Choice:	3
Python Installation:.....	4
Mac OSX Ventura	4
Windows 11	4
Required Modules:.....	4
Checking for existing Modules	5
Installing required modules	5
Script Variables:	5
XIQ and script behavior	5
Email.....	6
SMTP (optional)	6
Generating the XIQ Token.....	7
Creating the Google Cloud Token and Refresh Token	9
Generating a Token and Refresh token	12
XIQ SSID ID	13
SMTP Relay	13
Running the Script:	14
Log File	14
CSV File	14
Scheduling Script to Run.....	14
Mac & Linux-based Systems.....	14
Setting up a Cron Job	14
Windows based Systems	16
Setting up Windows Task Scheduler	16

Prerequisites:

- Knowledge of XIQ by adding access points, creating network policies, and SSIDs
- XIQ PSK SSID
- One or more XIQ native access points
- Download the following files and keep them in the same folder.
 - XIQ_PSK_Rotator.py
 - Version 1.0.0 is the current version. See lines 3-11 in the script.
 - psk_list.csv – list of PSKs to use. Edit this list with the list of PSKs you would like to use
 - variables.yml – variables and information for the script (outlined below)
 - requirements.txt (needed Python modules - see [modules](#) section)
 - app/ (folder containing the following scripts)
 - gmail.py (handles all Gmail APIs)
 - smtp.py (handles SMTP email)
 - logger.py (handles all logging)
 - xiq_api.py (handles all XIQ APIs)

```
3 #####
4 # written by: Tim Smith
5 # e-mail:    tsmith@extremenetworks.com
6 # date:      16 Aug 2024
7 # version:   1.0.0
8 #
9 # Note: The script is not officially supported by Extreme Networks
10 #
11 #####
```

Scripting Environment Preparation:

Information:

The XIQ_PSK_Rotator.py script requires Python 3.9 at minimum and has been tested up to Python 3.12. It can be executed manually but ideally scheduled to run automatically as a cronjob or Windows Task (see [Scheduling the Script](#)). This script can be executed from any device with Python and the needed modules installed. The device running the script needs to be able to reach the following:

1. **ExtremeCloud IQ:** The device must be able to connect to ExtremeCloud IQ.
2. **Gmail APIs:** If using Gmail APIs, the device needs access to Google's Cloud.
3. **SMTP:** If using SMTP, the device must be able to reach the SMTP server or relay.

These requirements ensure the device can communicate with the necessary services for its intended functionality.

When run, the script creates a PSK_rotator_log.log file. This file shows information about the PSK rotation, CSV file updates, and any API errors experienced.

Device Choice:

This script can be executed from any device running Python 3.9 or higher. The device could be a RedHat server, a PC/laptop running Windows or Mac OSX, or even a Raspberry Pi-type device. The device must be on the network and able to reach the Gmail cloud or SMTP server and ExtremeCloud IQ. This can be done through a proxy. Proxy config is beyond the scope of this guide.

NOTE: If using Gmail, the device will also need a browser installed, as that will be required for the initial Gmail API setup.

Python Installation:

Depending on the device used, you may need to install Python or a different version of Python. The easiest way to check the version of Python is to open the terminal (Power Shell on Windows) and type this command.

```
python --version
```

You may need to use this command on Macs, Linux, and other systems with Python 2 installed.

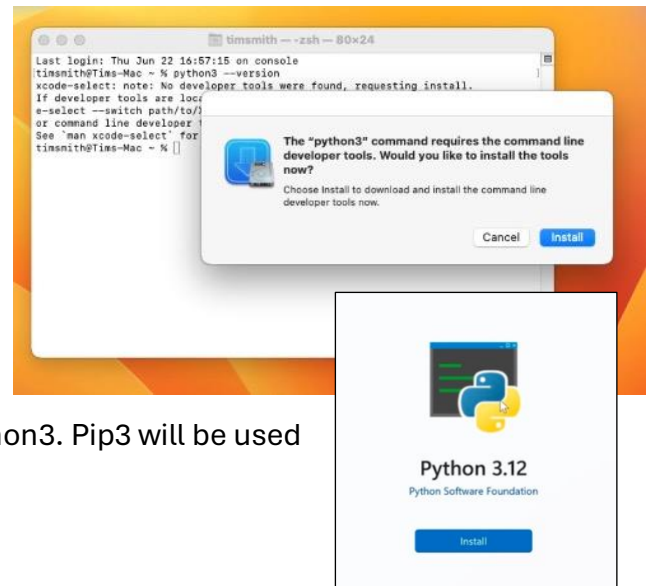
```
Python3 --version
```

Below are some examples of installing Python 3 for Windows and Mac OSX. The tested Linux systems all had Python3.6 or higher installed by default. You should upgrade to at least Python 3.9, as the script will not function correctly. Upgrading Python is beyond the scope of this guide. Here is a link to information.

<https://www.redswitches.com/blog/upgrade-python/>

Mac OSX Ventura Installation

- Open the terminal and enter `python3 --version`
 - This triggers the installation of Developer Tools
- Click Install
- Click Agree
- The Developer tools that installed `python3` will also install `pip3` in Ventura.



Windows 11 Installation

- Search Microsoft Store for Python 3.12 and click install
- Log in with Microsoft credentials
- The Windows store installs `pip3` with `python3`. `Pip3` will be used to install the needed modules.

Required Modules:

The **requests**, **PyYAML**, **google-api-python-client**, **google-auth-http2**, **google-auth-oauthlib**

modules are the only modules required for the `XIQ_PSK_Rotator.py` script.

Checking for existing Modules

You can check if the required module is installed using the terminal (PowerShell for Windows). Run the following command.

```
python3 -c "import requests"
```

The module is not installed if a `ModuleNotFoundError: No module named '<module name>'` error is returned.

Installing required modules

The required module can be installed using pip3 using the downloaded requirements.txt file with the following command.

```
pip3 install -r requirements.txt
```

NOTE: If you are using the 'python --version' command, you should use 'pip install' instead of 'pip3 install'

Or the module can be installed individually using:

```
pip3 install requests
pip3 install PyYAML
pip3 install google-api-python-client
pip3 install google-auth-httpplib2
pip3 install google-auth-oauthlib
```

Script Variables:

The script's custom variables will be loaded through the `variables.yml` file. This file must be updated with the correct values. We will briefly cover each of these and go into more detail below.


XIQ and script behavior

Line 4 `XIQ_Token` – a token can be generated to only allow access to view/edit SSIDs, list devices, deploy configuration, and check the status of operations. Details on generating this token are below in the [Generating the XIQ Token](#), specifically the [Generating Specific Tokens](#) sub-section.

Line 7 SSID_ID—When created, Each SSID in XIQ has a unique ID assigned to it. In the [XIQ SSID ID](#) section, we will cover how to get this ID.

Line 11 `allow_mismatched`—If this is set to `False`, the script will not run if any devices are mismatched. Devices can be corrected, and the script can be rerun. If this is set to `True`, the script will change the PSK and push all pending configuration changes with the PSK change.

```

1 #####
2
3 # The API token for XIQ - Generate a token with set expiration time. It will need to include these permissions. "ssid", "device:list", "deployment", "lrair"
4 XIQ_token: "xxx"
5
6 # The ID of the XIQ SSID - see guide on how to get this using swagger
7 SSID_ID: 0
8
9
10 # allow script to run even if APs are in mismatched state - if False, email will be sent if devices are in mismatched.  will not be updated
11 allow_mismatched: False
12
13 # allow script of push config to devices - if mis matched devices allow_mismatched would need to be True
14 allow_config_push: True
15
16
17 ## File name for PSK_list - Full Path!
18 file_name: "/Path-to-folder/psk_list.csv"
19

```

```

1 #####
2
3 # The API token for XIQ - Generate a token with set expiration time, it will need to include these permissions: "ssid", "device:list", "deployment", "lrai:"
4 XIQ_token: eyJraW01OjE1MzhzZDMwMTVvY2UwMEZodG9MTFNSY2VyJFMUdzZCt1czIyS1ZlLTJMjU2In0.eyJzdWIjOiJ0bm9kaXQ1L2psdG9fNEBmbWFrbc3J5b2Rlc2Y2ZmZDMDIsIHR5cGU6ImNBIiwiaWF0IjYxNDk1OTg0OD83MS0=
5
6 # The ID of the XIO SSID - see guide on how to get this using swagger
7 SSID_ID: 433791697007831
8
9
10
11 # allow script to run even if APs are in mismatched state - If False, email will be sent if devices are in mismatched. PSK will not be updated
12 allow_mismatched: True
13
14 # allow script to push config to devices - If mis matched devices allow_mismatched would need to be True
15 allow_config_push: True
16
17
18 ## File name for PSK_list - Full Path!
19 file_name: /Users/tismith/Scripts/Python/XIQ/XIQ_PSK_Rotator/psk_list.csv
20
21
22

```

Line 14 `allow_config_push` - If this is set to **False**, the script will change the PSK, but a user would need to manually push the configuration to the devices for it to be applied. If this is set to **True**, the script will push a delta config to the devices.

Line 18 `file_name` – This is the filename for the PSK CSV with the full path to the folder. You can update the included `psk_list.csv` file, but the full path needs to be added.

Email

Line 30 `email_type` – You can specify how the script will use email.

The following options are available:

1. **gmail**: use the Gmail API to send emails from a Gmail account. See the [Creating the Google Cloud Token and Refresh Token](#) section.
2. **smtp**: use an SMTP server or SMTP relay. See the [SMTP](#) section. Or the [SMTP Relay](#) section.
3. **disabled**: if emails will not be used by the script. Email messages will be added to the log file.

Line 35 `email_list` – a list of email addresses to which you want the PSK to be sent. Add the email address below with a `-` in front of them as seen.

Line 40 `email_sub` – the subject for the email to be sent

Line 43 `email_msg` – this is what the email body will contain. The PSK, when generated, will be appended to the end of this message.

SMTP (optional)

If you use SMTP, the last section of the `variables.yml` file needs to be updated.

Line 49 `username` – The username for the SMTP user.

Line 51 `password` – The users password

Line 53 `sender_email` – the email address of the sender

Line 57 `smtp_server` – the FQDN or IP address of the SMTP server.

NOTE: You will need to add `""`s around IP addresses.

Line 58 `smtp_port` – 587 is the standard port, but it can be changed here if needed.

```
21 #####
22 # EMAIL
23 #####
24
25 # select type of email
26 ## ~ gmail ~ uses gmail API ~ follow guide to configure
27 ## ~ smtp ~ uses smtp ~ could use a smtp relay like sendgrid if local SMTP server not available https://app.sendgrid.com/ (not affiliated)
28 ## ~ disabled ~ no email will be used
29
30 email_type: gmail
31
```

```
32 # EMAIL Variables - fill out if gmail or smtp is selected for 'email_type'
33 #####
34 ## list of emails to recieve psk
35 email_list:
36 | - user1@example.com
37 | - user2@example.com
38
39 ## email subject
40 email_sub: New Guest PSK
41
42 ## email message - new psk is appended later
43 email_msg: The new PSK for Guest is
44
```

```
46 # SMTP Variables - fill out if smtp is selected for 'email_type'
47 #####
48 ##example username: apikey
49 username:
50 ##example password: SG.B1A10tUjQT
51 password:
52 ##example sender_email: mike@contoso.com
53 sender_email:
54
55 ##example smtp_server: smtp.sendgrid.net
56 ##example smtp_server: "192.168.10.4" - add quotes around IP addresses
57 smtp_server:
58 smtp_port: 587 # change port as required by your SMTP server
```

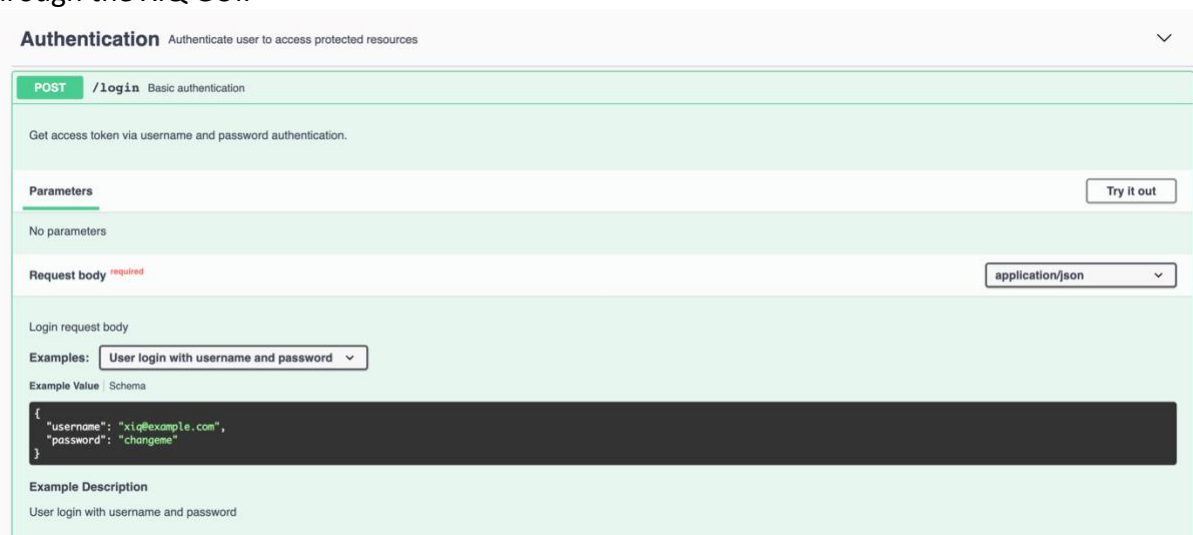
Generating the XIQ Token

You can view our developer portal site at <https://developer.extremecloudiq.com>. You can find a link to our swagger page and other developer tools here. There is also a Communities section to reach out to with any questions.

Swagger

We will use the swagger interface to generate the token <https://api.extremecloudiq.com/>. On the swagger page, clicking on any API will expand information about the API and allow you to try it. Clicking the “Try it out” button, filling out any needed information, and then clicking the execute button will allow you to try that specific API call.

The second-generation APIs are based on access tokens generated by an XIQ account. For more details, see [A Guide to Getting Started with v2 APIs in XIQ](#). Currently, these tokens can only be generated through the /login POST API request; they cannot be generated through the XIQ GUI.



The image shows the Swagger UI for the `POST /login` endpoint. The title is "Authentication" with a subtitle "Authenticate user to access protected resources". The endpoint is described as "Basic authentication" and "Get access token via username and password authentication." There are no parameters. The request body is required and the media type is set to `application/json`. An example is provided for "User login with username and password" with the following JSON body:

```
{  "username": "xiq@example.com",  "password": "changeme"}
```

The example description is "User login with username and password".

Login

Request Body

```
{  "username": "xiq@example.com",  "password": "changeme"}
```

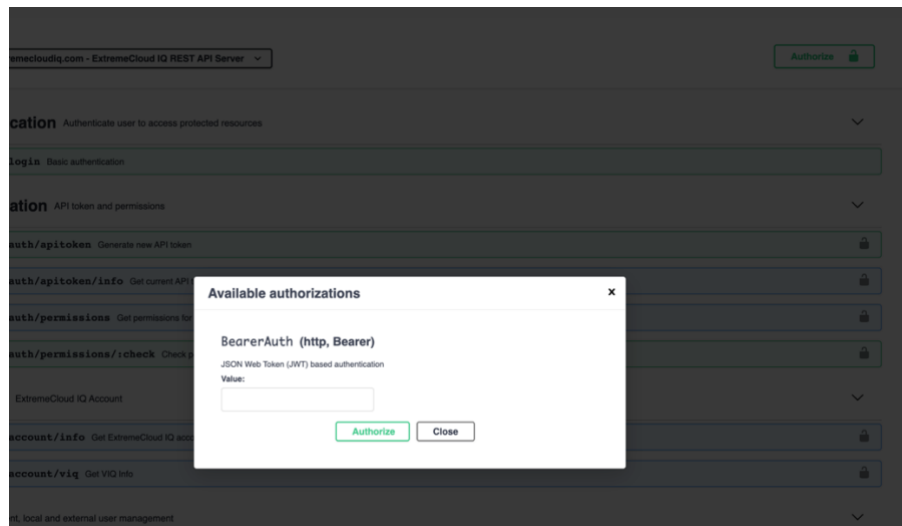
The `/login` POST request is used to generate an access token. In the request body, enter a local administrator XIQ account username and password, and the API will respond with an access token that can be used for any following calls. This token will be valid for **24 hours** after creation, and this token will have the ability to be used for **any** of the API calls the user is authorized for within XIQ.



For this script, we will use this token to generate a separate token with limited access and a specified expiration time. Copy the access token created, not including the ""s.

Authorize in Swagger

At the top of the Swagger page, click the authorize button. A window will pop up, allowing you to paste the access token. Clicking “Authorize” Swagger will set Swagger to use the added access token for the API calls on the page.



Generating Specific Tokens



The `/auth/apitoken` POST request allows you to specify an expiration time and set permissions for a token. This is a great way to create a token for a specific application or script, only allowing the token to perform the needed tasks.

The expiration time uses Epoch time, which is the number of seconds since midnight on Jan 1, 1970 (UTC). <https://www.epochconverter.com/> is a webpage that can convert a readable time to an epoch time or epoch time to a more readable time. Set a time for 1 year out and get the epoch time.

For this script, we will want to have the following permissions - `"ssid", "device:list", "deployment", "lro:r"`

This will give us access to view and update SSIDs, view the devices, deploy delta configuration updates, and check the status of operations.

Adding the desired expiration time and a list of permissions, this API will return a token that is only usable by the specified APIs.

Request Body

```
{
  "description": "Token for XIQ_PSK_Rotator.py script",
  "expire_time": 1723901676,
  "permissions": [
    "ssid",
    "device:list",
    "deployment",
    "lro:r"
  ]
}
```

Response Body

```
{
  "access_token":
    "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ0aW1qc21pdGgyNEBwcm90b25tYWIsLmNvbSIsInNjb3BlcyI6WyJhY2NvdW50OnliXSwidXNlcklkjoyMTc5MjMyMSwicm9sZSI6IkFkbWluaXN0cmF0b3llLCJjdXN0b21cklkjoyMTc5MTk3MSwiY3ZvdG9tZXJNb2RlIjowLCJoaXFFbmFibGVkljpmYWxzZSwib3duZXJJZCI6MTc5MTYxLjJvcmdJZCI6MCwiZGF0YUNlbnRlcil6IklBXC0dDUCiSlmzcyI6ImV4dHJlbWVjbG91ZGxLmNvbSIsImhdCI6MTYyODE4MzA0SwiZXhwIjoxNjI4MTg2NDI4fQ.CtBGq4YVGB9FzCodr6Oi5IG8yy1-4B-77AW5rVG3S0",
  "create_time": "2021-11-29T15:47:57.000+0000",
  "expire_time": "2021-11-29T16:10:08.000+0000",
  "creator_id": 21792321,
  "customer_id": 21791971,
  "description": "Token for XIQ_PSK_Rotator.py script",
  "permissions": [
    "ssid",
    "device:list",
    "deployment",
    "lro:r"
  ]
}
```

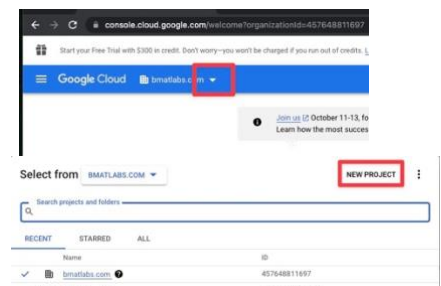
Copy the newly created **access_token** and add it to the **XIQ_token** variable in the variable.yml file.

Creating the Google Cloud Token and Refresh Token

A token must be generated every time the script is run to access the Gmail using API. This token is only valid for 1 hour but can be easily generated within the script using a provided credentials.json file and a Refresh token. The sections below cover what is needed to allow API access to your Google Cloud environment, create the credentials.json file, authorize the script to use the JSON file information and generate the refresh token.

Creating the Google Cloud Project

To create the Google Cloud Project needed to access Gmail using API, open a browser and go to console.cloud.google.com. Once logged in, click the dropdown on the top left. This could be the domain name you are using, another project you've created, or you may say "New Project." Clicking that dropdown will open a new window allowing you to select different projects you have. A button to create a new project is on the top right of that window.



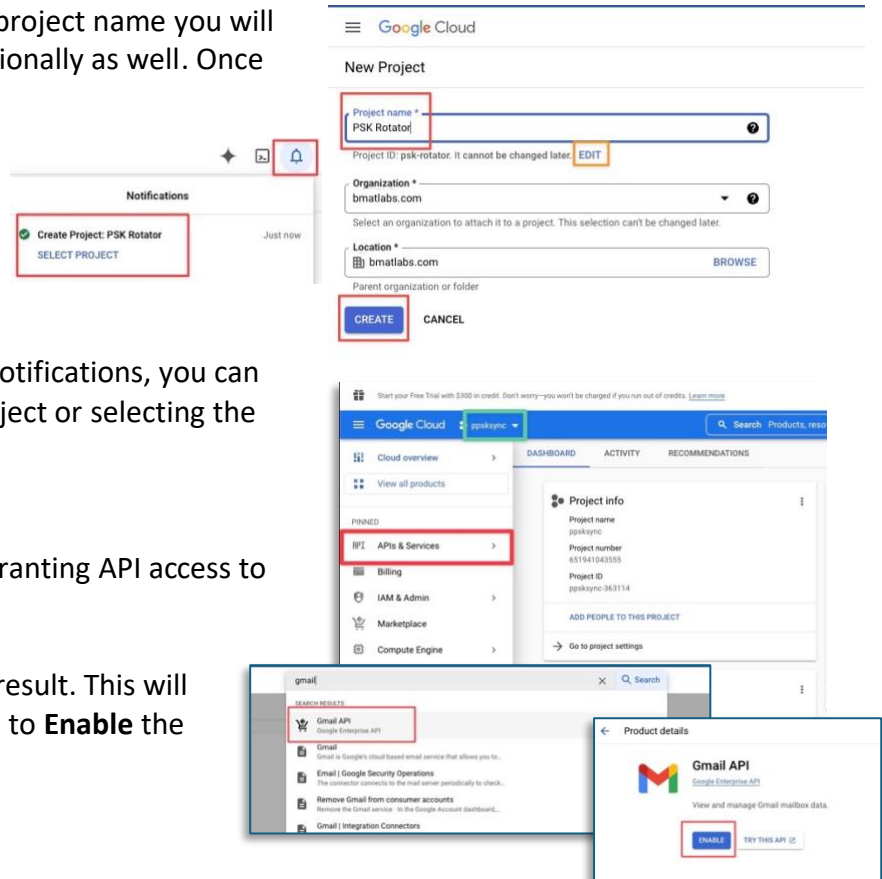
In the newly opened window, set the project name you will create. You can edit the Project ID optionally as well. Once completed, click the **Create button**.

The project will take a second to create. You can monitor the notification section to see once it's complete.

Once you see the green check in the notifications, you can open the project by clicking Select Project or selecting the new project from the dropdown.

First, we must enable the Gmail API, granting API access to the logged-in user's email.

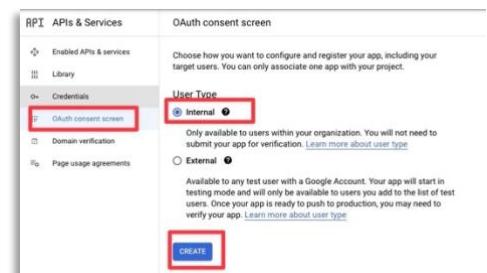
Search for "Gmail API" and select the result. This will open a window where you can choose to **Enable** the APIs.



Creating the Google Cloud API OAuth Consent Screen

Now that your Google Cloud project is created, you must set up the Consent Screen for OAuth. This will allow you to provide approval to grant access to the script. This page will just be shown the first time the script is run. See the [Generating Google Cloud token and refresh token](#) section.

On the left of the APIs & Services page, select the **OAuth consent screen** tab, select the **Internal User Type** button, and click **Create**.



For the app information, fill out the app name, select the user support email, and enter your email for the developer's contact information. Everything else on the form can be left blank.

Click the **save and continue** button. On the scopes page, leave everything blank and click the **save and continue** button at the bottom. We will be setting the scope the first time the script runs.

Creating the Google Cloud API OAuth Credentials

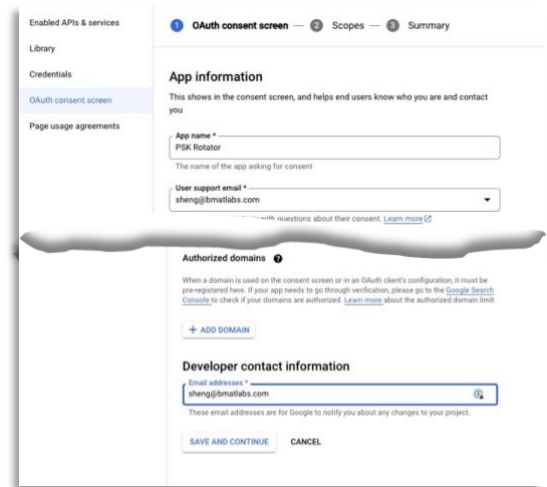
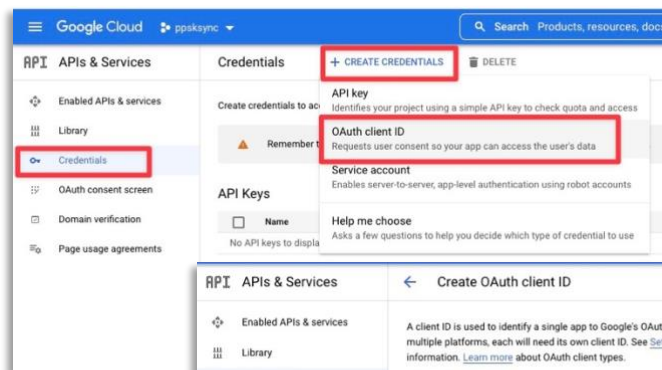
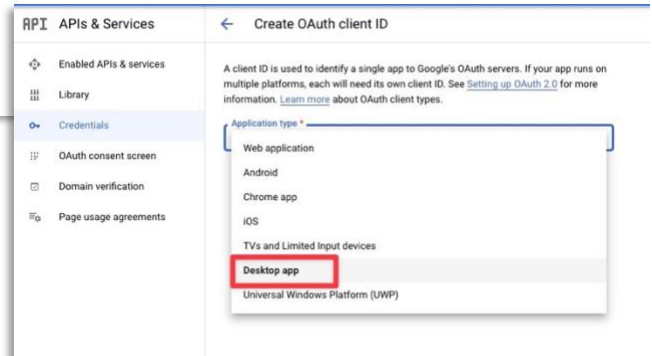
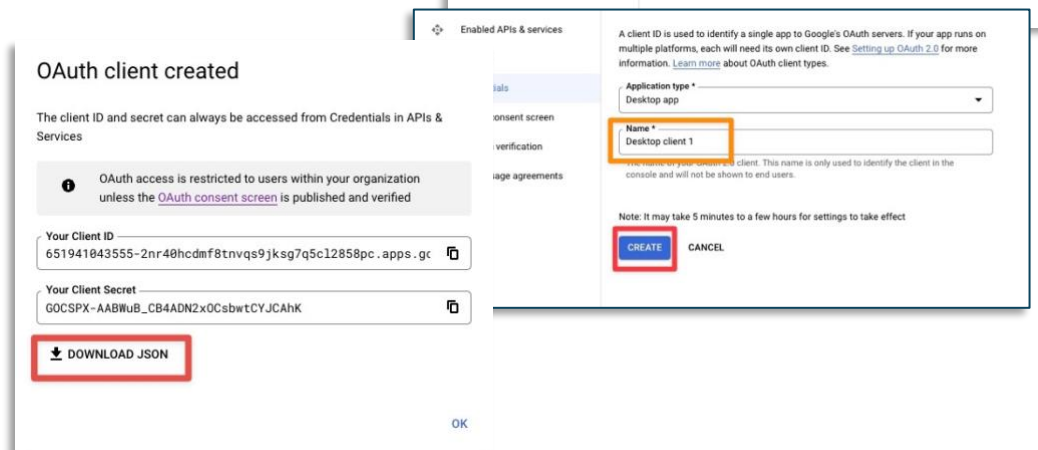
Next, we must create the OAuth credentials for the Google Cloud API.

Select the Credentials tab on the left of the APIs & Services page. At the top of that page, choose **Create Credentials** and select **OAuth client ID** from the drop-down.

When the OAuth client ID opens, select the **Desktop app** for the Application Type.

Then, enter the name you would like for the client in the next window and click **Create**.

Select the 'Download JSON' button when the OAuth client is created.

The JSON file will download to your browser's default location. The file will be named something like “`client_secret_{the_id_created}pc.apps.googleusercontent.com.json`”. Rename this file to “`credentials.json`” and move it to the folder where the script is located.

NOTE: The first time the script runs, it will read the `credentials.json` file and ask you to authenticate the Google app in the browser. After doing this the first time, the script will save a file called `token.pickle`. As long as this `token.pickle` file is present, the script will not need to authenticate in the browser again.

Generating a Token and Refresh token

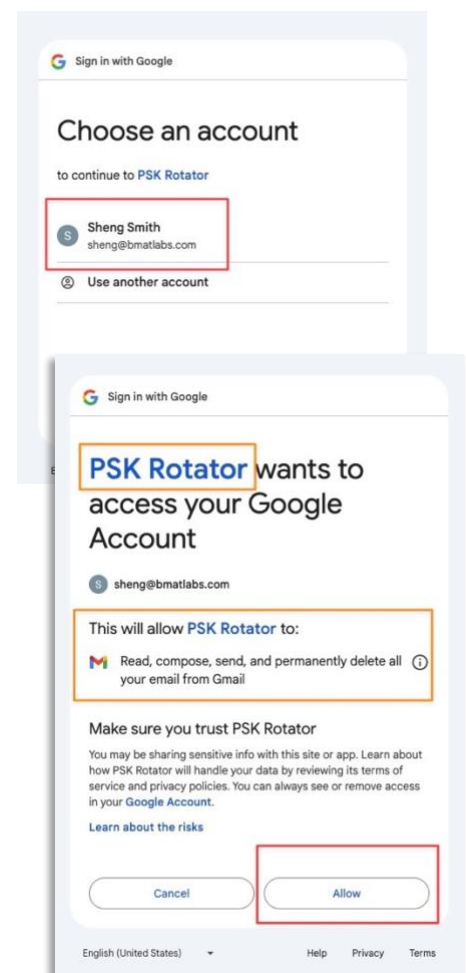
The first time the script is run (or if the `token.pickle` file is missing), the script will use the `credentials.json` file to authorize the script to access your Gmail. Ensure the `credentials.json` file has been renamed and added to the same folder as the script. When the script is run, a URL will be provided for you to authorize access. A web browser may also open this page.

```
(venv) tsmith@EXT-IMP7N07X: ~/X10_PSK_Rotator % python X10_PSK_Rotator.py
Please visit this URL to authorize this application: https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=118712990068-9m288kfcbrfb2k1dbb2daa@hcmr7c3.apps.googleusercontent.com&redirect_uri=http%3A%2F%2Flocalhost%3A5555%2F&scope=https%3A%2F%2Fmail.google.com%2F&state=5X2gAK3Lfg1SMAF1K0ftAT3KCF3ZM&access_type=offline
```

With the linked browser page open, select the Google Cloud user.

On the next page, confirm the Project name. You can also see what permissions the script requests. This grants the script permission to read, compose, send, and delete emails. This script will only ever send emails.

Once validated, select Allow. You can then close the browser, and the script will complete. A new file will be created called `token.pickle`. This file has the needed token and refresh token info, the client ID, and the client secret. It will be used in the `gmail.py` script to generate a new token each time the script is run.



XIQ SSID ID

Each XIQ SSID will be assigned a unique ID when created. This is something that the backend systems use and is not seen in the GUI. The easiest way to get the ID is from the Swagger page.

Return to the swagger page, scroll to the Configuration – Policy section, and find the `/ssids` GET request.



Click the “Try it out” button, then the “Execute” button. When you find the Name of the XIQ SSID you want to use, it will be inside a pair of {curly brackets}. Inside the same pair of curly brackets will be an element called `id`, and this is the ID that is needed.

Response Body

```
{
  "page": 1,
  "count": 10,
  "total_pages": 2,
  "total_count": 12,
  "data": [
    {
      "id": 32000,
      "create_time": "2019-08-19T15:44:36.000+0000",
      "update_time": "2019-08-19T15:44:36.000+0000",
      "org_id": 0,
      "name": "ssid0",
      "broadcast_name": "ssid0",
      "description": "Default SSID profile",
      "predefined": true,
      "advanced_settings_id": 33000,
      "enable_user_profile_assignment": false,
      "enable_radius_attribute_user_profile_assignment": false,
      "attribute_key": 0,
      "access_security": {
        "key_value": "",
        "anti_logging_threshold": 0,
        "transition_mode": false,
        "security_type": "OPEN"
      },
      "radius_client_profile": {
        "default_radius_client_object_id": 0,
        "enable_classification": false,
        "classified_entries": []
      },
      "default_user_profile": 36000,
      "vendor_id": 0,
      "user_profile_assignment_rules": []
    },
    {
      "id": 433791697007831,
      "create_time": "2022-10-25T20:06:18.000+0000",
      "update_time": "2024-05-09T14:24:56.000+0000",
      "org_id": 0,
      "name": "Lab-2",
      "broadcast_name": "Lab-2",
      "description": "",
      "predefined": false,
      "advanced_settings_id": 433791697007137,
    }
  ]
}
```

Once that is obtained, add it to the `SSID_ID` variable in the `variables.yml` file.

SMTP Relay

An SMTP relay is a service that facilitates the transmission of email messages from one server to another. SendGrid, a free cloud service, allows you to send up to 100 daily messages. To use this service, you must create an account and update your API key, ‘To’ address, and ‘From’ address variables. You can sign up at SendGrid, which is not affiliated with Extreme. The setup for the SMTP relay is not covered in this guide, but the necessary information should be collected and added to the SMTP section of the `variable.yml` file.

Running the Script:

To run the script, open the terminal (PowerShell for Windows) to the location of the script and run:

```
Python3 XIQ_PSK_Rotator.py
```

You can also make the script executable by running
Then, you can run the script by typing

```
chmod +x XIQ_PSK_Rotator.py
```

```
./XIQ_PSK_Rotator.py
```

Log File

Upon running the script, a log file named PSK_rotator_log.log will be created. Additional runs of the script will be appended to this log file.

This file will contain successful PSK updates, CSV file updates, and any API issues.

```
PSK_rotator_log.log
2 2024-08-06 15:20:39,133: root - INFO - Successfully updated CSV file /Users/tismith/Scripts/Python/XIQ/XIQ_PSK_Rotator/psk_list.csv
3 2024-08-06 15:25:00,205: Psd_Check_xiq_api - ERROR - Error - HTTP Status Code: 401
4 2024-08-06 15:25:00,205: Psd_Check_xiq_api - WARNING - ('message': '401 Unauthorized: Invalid Token')
5 2024-08-06 15:25:00,205: Psd_Check_xiq_api - WARNING - API to change psk failed attempt 1 of 5 with Error - HTTP Status Code: 401
6 2024-08-06 15:25:00,320: Psd_Check_xiq_api - ERROR - Error - HTTP Status Code: 401
7 2024-08-06 15:25:00,320: Psd_Check_xiq_api - WARNING - ('message': '401 Unauthorized: Invalid Token')
8 2024-08-06 15:25:00,320: Psd_Check_xiq_api - WARNING - API to change psk failed attempt 2 of 5 with Error - HTTP Status Code: 401
9 2024-08-06 15:25:00,455: Psd_Check_xiq_api - ERROR - Error - HTTP Status Code: 401
10 2024-08-06 15:25:00,455: Psd_Check_xiq_api - WARNING - ('message': '401 Unauthorized: Invalid Token')
11 2024-08-06 15:25:00,456: Psd_Check_xiq_api - WARNING - API to change psk failed attempt 3 of 5 with Error - HTTP Status Code: 401
12 2024-08-06 15:25:00,580: Psd_Check_xiq_api - ERROR - Error - HTTP Status Code: 401
13 2024-08-06 15:25:00,580: Psd_Check_xiq_api - WARNING - ('message': '401 Unauthorized: Invalid Token')
14 2024-08-06 15:25:00,590: Psd_Check_xiq_api - WARNING - API to change psk failed attempt 4 of 5 with Error - HTTP Status Code: 401
15
```

CSV File

When run, the script will take the top PSK from the list in the CSV file. After the PSK has been updated in XIQ the script will save the CSV file with that PSK removed. Make sure to add additional PSKs to the CSV as needed. If the CSV list is empty, the script will print and email a message that the list is empty and quit.

```
The csv file /Users/tismith/Scripts/Python/XIQ/XIQ_PSK_Rotator/psk_list.csv is empty
Script is exiting...
```

Scheduling Script to Run

Mac & Linux-based Systems

A Cron job can be set up to run the script at a specified interval automatically. Ideally, this could be set for every day, week, or month, ensuring that the PSK is rotated automatically. The script can also be run manually between those times if a PSK needs to be rotated immediately.

Setting up a Cron Job

Open and edit the crontab and configure the job according to the command you want to run. From the terminal window, enter the following command.

```
crontab -e
```

There are three parts to a cron job configuration.

Cron Job Time Format

Part 1 of the cron job

The first five characters, a b c d e, represent the job's time, date, and repetition.

- a – Minute (0-59)
- b – Hour (0-23)
- c – Day (0-31)
- d – Month (0-12) – 0=None and 12 = December
- e – Day of the Week (0-7) – 0=Sunday and 7=Sunday

- **An asterisk (*)** stands for all values. Use this operator to keep tasks running during all months, or all days of the week.
- **A forward-slash (/)** is used to divide a value into steps. (*2 would be every other value, */3 would be every third, */10 would be every tenth, etc.)

The time format would look like this to set the Cron job to run daily at 3 am.

```
0 3 * * *
```

The time format would look like this to set the Cron job to run at midnight on the 1st of the month.

```
0 0 1 * *
```

The time format would look like this to set the Cron job to run at 8 AM every Sunday.

```
0 8 * * 0
```

Cron Job Script and Script Location

Part 2 of the cron job

The next part is where you enter the script you want to run and its location.

```
python3 /home/admin/documents/scripts/XIQ_PSK_Rotator.py
```

You can make the script executable, so you don't have to type python3 before entering the script. Instead, you will enter a period before the location.

```
chmod +x /home/admin/documents/scripts/XIQ_PSK_Rotator.py
```

```
./home/admin/documents/scripts/XIQ_PSK_Rotator.py
```

Cron Job Output and Job Completion

Optional Part 3 of the cron job

The last part is optional and specifies where the script's output and completion should go. If not set, the cron will email the owner of the crontab file.

It is recommended that something be set for the output to avoid filling up the server's inbox. This can be set to append to a file.

```
>> /home/admin/documents/scripts/ XIQ_PSK_Rotator-Output.txt
```

Or can it be set to turn off the email output


```
> /dev/null 2>&1
```

Cron Job Command Example

The command should be entered in a single line and saved in the crontab file.

Every Monday at 8 am turning off the output

```
0 8 * * 1 python3 /home/admin/documents/scripts/XIQ_PSK_Rotator.py > /dev/null 2>&1
```

Every Month on the 15th at Midnight, with saved output

```
0 0 15 * * ./home/admin/documents/scripts/XIQ_PSK_Rotator.py >> /home/admin/documents/scripts/XIQ_PSK_Rotator-Output.txt
```

Windows based Systems

A Windows task schedule can be set up to run the script at a specified interval automatically. Ideally, this could be set for every day, week, or month, ensuring that the PSK is rotated automatically. The script can also be run manually between those times if a PSK needs to be rotated immediately.

Setting up Windows Task Scheduler

Open Control Panel > System and Security > Administrative Tools > Task Scheduler

Selected 'Create basic task....'

Give your task a name like 'XIQ_PSK_Rotator' and click 'Next.'

Select if you want the script to run Daily, Weekly, Monthly, etc, and click 'Next.'

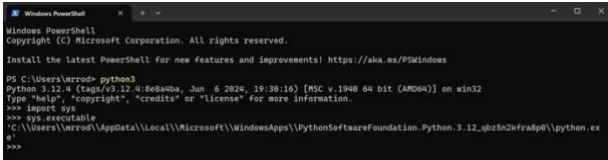
Adjust settings for the scheduled and click 'Next.'

Select 'Start a program and click 'Next.'

Start a Program

For the Program/script: section, enter the path of your python.exe file.

The location of your python.exe file depends on how it was installed. An easy way to find this location is to open Windows PowerShell and enter python3. This will open the Python interpreter. In the interpreter, enter **import sys**, then **sys.executable**.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\jerrad> python3
Python 3.12.4 (tags/v3.12.4:86b86ba, Jun 6 2024, 19:36:16) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more
>>> import sys
>>> sys.executable
'C:\Users\jerrad\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\python.exe'
>>>
```

This will output the location of the python.exe file. Enter **exit()** to exit the interpreter.

Enter the full path of the python.exe file in the Program/Script: field. Make sure not to copy the "s around the path.

Enter the script's name in the Add arguments (optional) field.

XIQ_PSK_Rotator.py

Enter the script's location in the Start in the (optional) field.

C:\user\your_python_project_path

Click 'Next.'

Click 'Finish'

