Project Proposal: **Web based Certificate and Profile User Interface**

Background

There are shell scripts for creating X.509 certificates, revoking certificates and signing CRLs and scripts for the creation of Profile certificate files for certain devices such as Linux, Apple OS X, Windows, iOS, etc., these require careful specification of various certificate attributes so that these certificates work on a variety of devices: Android, Windows, iOS/OSX, Linux, etc. The goal of this project is to gather all that knowledge into a simple interface.

Objectives

The interface should support the following:

- Generating the proper ipsec.conf configuration based on web admin interface including DNS/split-DNS configurations.
- Allow Administrator to invite new users using email id.
- A new user after account validation can download the generated certificate/profile (over TLS) for different platforms.
- The generated certificates/profiles can only be downloaded once, through the portal.
- Admin can list, revoke/disable (temporary revocation) user certificates/profiles.
- Generate *PKCS#12* certificates for users.
- Generate iOS/OSX *.mobileconfig* profiles for automatic installation on iOS/OSX.
- *Ipsilon* user authentication to web application.
- Configure munin-node to work with libreswan plugin.

Technical

Components/Modules

This project has been divided into 4 components, which are as follows:

1. **Initial Configuration setup:**
    a. Initial Django setup and configuration.
    b. Development of project configuration scripts, which will be used to install and configure it for the 1st time use.
    c. It'll also include adding/updating DNS/Split DNS configurations.
    d. Admin profile creation and keys/certificates generation.

Note - The configuration script/scripts will take input, the values to store in configuration file. Which will be used to establish connections and more.

2. **Admin interface setup:**
    a. Admin login through Web Interface.
    b. Adding functionality options to admin profile.
    c. Creating add/invite user interface along with token validation.
    d. User profile creation, validation along with user certificates generation.

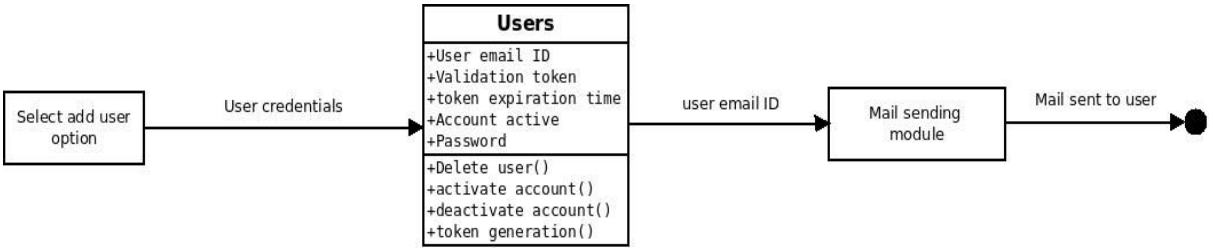
3. **Managing user access:**
    a. Adding user revocation functionality.
    b. User deletion functionality.

4. **Traffic management:**
    a. Showing traffic analytics to Admin.
    b. Users data quota tracking.
    c. On quota exhaustion show user the relevant available options.

Architecture/Design

Architecture and design for working of various features are as follows:

1. **Add user DFD**



[Figure 1: Add user DFD]

    a. Admin will select add user option and enter the user email id for inviting.
    b. On the entered email id, validation token will be sent. The validation token will be valid for a particular time span.
    c. User account will be active once the token is used in the given time span.
    d. The user will set a password after email verification.

Note - Validation token will be generated using a python library Lib/secrets.py. The secrets module is used for generating cryptographically strong random numbers suitable for managing data such as passwords, account authentication, security tokens, and related

secrets. Below is the function for token generation.

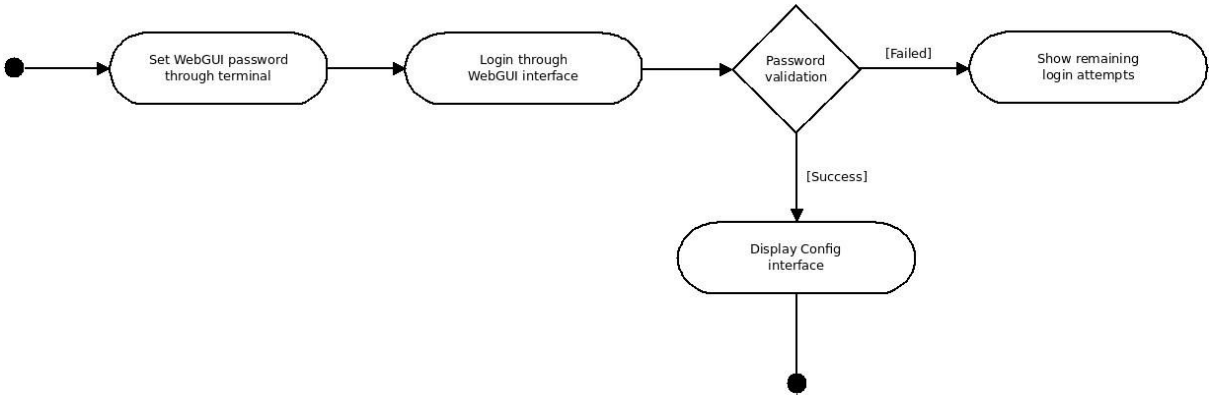
```
#Function for token generation
secrets.token_urlsafe([nbytes=None])

"""
The above function returns a random URL-safe text string,
containing nbytes random bytes. The text is Base64 encoded, so on
average each byte results in approximately 1.3 characters. If
nbytes is None or not supplied, a reasonable default(32 bytes) is
used.
"""
#Using the function
>>> token_urlsafe(16)
'Drmhze6EPcv0fN_81Bj-nA'
```

Generating a hard-to-guess temporary URL containing a security/Validation token:

```
#email verification/account validation url for user
url = 'https://mydomain.com/token=' + token_urlsafe()
```

Note - When user visits the above url the token will be validated and on the basis of token's validity, his/her account will be activated.

2. Admin activity diagram



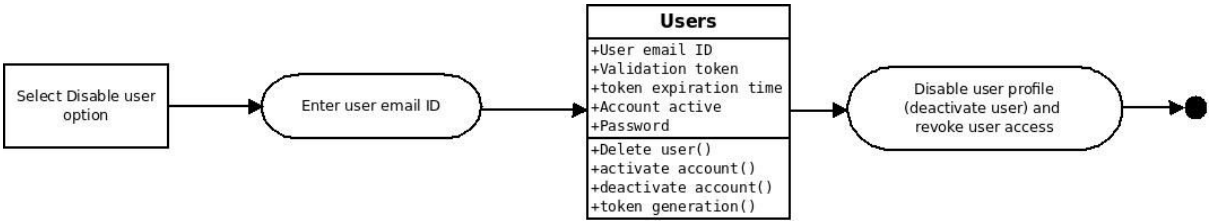
[Figure 2: Admin activity diagram]

a. Admin during the initial setup of the project will set WebGUI password through terminal along with setting other essential DNS/IP related configuration.

b. Once the initial configuration setting through terminal is done, admin will then login through its web interface.
c. On successful login, admin will see various configuration/user management options.

Note - Admin and user authentication will be either done using `'django.contrib.auth'` module and `'django.contrib.contenttypes'` will be used for allowing permissions to be associated with models.
OR
It'll be done using `ipsilon`.

3. Disable user



[Figure 3: Disable user]

a. Admin through the web interface can execute various tasks and one of them is disabling user. Admin will select the disable user option.
b. Then will enter the user's email id.
c. Then user's certificates will be temporarily revoked using pam_url's and user's account status will be put to deactivate in the database. *[User access revocation is done through PAM].*

Note - PAM will be added to Django by:

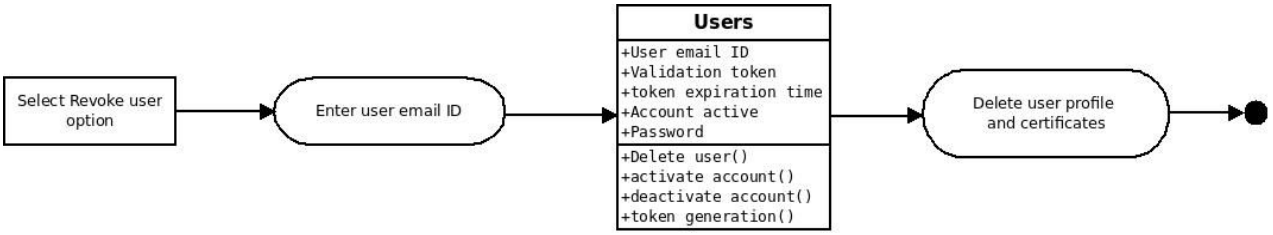
```
INSTALLED_APPS = [
    ...
    'django_pam',
]
```

Now adding Django PAM backend to the `AUTHENTICATION_BACKENDS:`

```
AUTHENTICATION_BACKENDS = [
    'django_pam.auth.backends.PAMBackend',
    'django.contrib.auth.backends.ModelBackend',
]
```

Reference - django-pam 1.2.0 .

4. Revoke user DFD



[Figure 4: Revoke user]

a. Admin will select the revoke user option.
b. On entering the user email id, the user will be deleted from the database and all the certificates generated for the user will be deleted permanently.

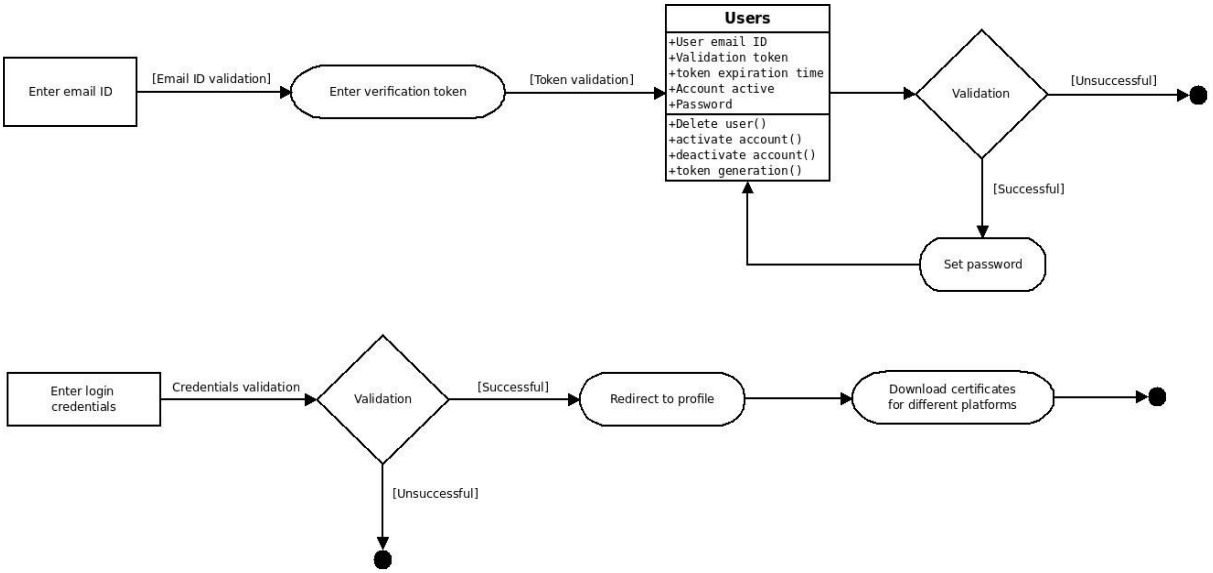
5. Certificate Generation

a. Certificates will be generated using Certbot and then can be imported as the PKCS#12 file into libreswan using "ipsec import".

6. Traffic and Quota(Data) management

a. For maintaining users data usage detail, we'll update the user's daily data usage in the database.
b. When a particular threshold is reached, the user will be redirected to a particular page. *[This can be used to suggest user to upgrade the existing plan].*
c. Once the data is gathered for over a week, admin can view all the data in a graphical format, for analytic purposes.

7. User account activation



[Figure 5: User account activation]

a. User will enter his/her email id along with the validation token.
b. The token will then be checked for its validation. On successful validation the user can set a password to his profile. *[Then the user needs to login]*
c. On successful login, the user can download the generated certificates/profiles for available devices/different platforms.

Timeline

In this the project is divided into several stages. Each stage constitutes various modules/components and is distributed over 12 weeks period.

Project Timeline		
Stage no.	**Stage specifications**	**Time required**
Stage 1	Initial Configuration setup	3 Weeks
Stage 2	Admin interface setup	3 Weeks
Stage 3	Managing user access	2 Weeks
Stage 4	Traffic management	2 Weeks
Stage 5	Debugging	2 Weeks

Evaluation

1. **Evaluation 1 [June 11 - 15, 2018]**

Duration	Tasks/Modules/Components
Week (1-4)	I. Django configured for optimum use by the project. II. Project 1st time installation scripts will be all working. III. Adding/Updating DNS configuration will be working. IV. Admin profile will be created along with certificates generation. V. Various functionality options will be added.

2. **Evaluation 2 [July 9 - 13, 2018]**

Duration	Tasks/Modules/Components
Week (5-8)	I. Add/Invite user interface will be working. II. User profile creation, validation and user keys/certificates generation(PKCS#12). III. User revocation will be in working.

3. **Evaluation 3 [August 6 - 14, 2018]**

Duration	Tasks/Modules/Components
Week (9-12)	I. User deletion will be complete. II. Debugging. III. Traffic management. IV. Reporting traffic data analytics to admin.

Personal

1. I am a passionate computer Science and Engineering student at one of the prestigious institutes of India - National Institute of Technology Hamirpur (H.P.), India - 177005.
2. I love contributing to Open Source.
3. Enjoy blogging, loves traveling and using Linux.

GitHub, Email, Website

- GitHub Username - Rishabh04-02
- Email ID's - cs14mi508@nith.ac.in , rishabh0402@gmail.com
- Website - https://rishabhchaudhary.in/

My timezone

- UTC/GMT +5:30 hours.
- Asia/Kolkata : IST (India Standard Time)

My comfort working with a remotely available mentor

I am comfortable working with a mentor who is several time zones away/located remotely. I have worked in this way before.

Tracking my progress

I have divided my work into stages and have developed a weekly schedule. This schedule will help me as well as my mentors to track my progress.

My native language

1. My native language is Hindi, but I've studied English and I am fluent in it.
2. English is my preferred language when working on the project.

Am I likely to finish the project in the allotted time?

1. I've divided the project into stages and in this way I can achieve my goals in a more convenient/sorted and planned way. Also in this way, it'll be easier for the mentors to track my progress.
2. I've designed Activity diagrams, DFD's for various tasks to be done by the web based system and hence, have a strong understanding of the project.
3. Also the basic tasks required for the project involves *working in python, database designing, Users management, authentication and validation, UI design, working with shell scripts, Sessions management, Using and generating certificates, creating and modifying configuration files.* I've worked on most of the above mentioned tasks, some in my projects and on some while contributing to other open source projects. *Ref - Rishabh04-02 .*