Configuration of single URL

Install python
yum install nginx python3 python3-pip
pip3 install flask

Now make directory
mkdir /zimbra-login

Now go in to the dir
cd /zimbra-login/

List the dir
ls
python3 -m venv venv
source venv/bin/activate
pip3 install flask requests

Make dir.
mkdir static

mv single.conf /root/

Unzip the dar file
tar xvfz single_url.tar.gz
ls
cd zimbra-login/
ls

Copy the app.py to zimbra-login
cp app.py /zimbra-login/
cp -pvr static templates /zimbra-login/

cd /zimbra-login/
cp app.py app.py.org
vi app.py

```
            ne_users = load_users('ne_users.txt')  # List of users for ne
            ose_users = load_users('ose_users.txt')  # List of users for ose
            logging.debug("User lists reloaded.")
        time.sleep(600)  # Sleep for 10 minutes (600 seconds)

# Start the background thread for refreshing user lists
refresh_thread = threading.Thread(target=refresh_user_lists)
refresh_thread.daemon = True  # Daemonize thread to exit when the main program exits
refresh_thread.start()

@app.route('/')
def index():
    return render_template('login.html')

@app.route('/zimbra_auth', methods=['POST'])
def zimbra_auth():
    try:
        xml_data = request.data.decode('utf-8')

        if not xml_data:
            return jsonify({"error": "XML payload is required"}), 400

        # Extract email from the XML data to determine the server
        email_start = xml_data.find("<account by=\"name\">") + len("<account by=\"name\">")
        email_end = xml_data.find("</account>")
        email = xml_data[email_start:email_end]

        # Access user lists with thread-safe locking
        with user_list_lock:
            if email in ne_users:
                url = "https://mta1.selo.local/service/soap"
            elif email in ose_users:
                url = "https://mta5.selo.local/service/soap"
            else:
                return jsonify({"status": "failed", "error": "User not found on any server"}), 400

        headers = {
            'Content-Type': 'text/xml; charset=utf-8',
            'SOAPAction': ''
        }

        logging.debug(f"Sending XML data to {url}")
        response = requests.post(url, data=xml_data, headers=headers, verify=False)

        logging.debug(f"Response Status Code: {response.status_code}")
        logging.debug(f"Response Text: {response.text}")

        if response.status_code == 200:
            response_xml = response.text
            root = ET.fromstring(response_xml)
            auth_token = root.find('.//{urn:zimbraAccount}authToken').text

            # URL encode auth_token
            encoded_auth_token = urllib.parse.quote(auth_token)

            if email in ne_users:
                zimbra_webmail_url = f"https://mta1.selo.local/service/preauth?isredirect=1&authtoken={encoded_auth_token}"
            else:
                zimbra_webmail_url = f"https://mta5.selo.local/service/preauth?isredirect=1&authtoken={encoded_auth_token}"
```

vi login.html

```
</head>
<body>
    <div class="container">
        <!-- Logo Section -->
        <img src="/static/crpf.jpeg" alt="CRPF Logo" class="logo">

        <h2>SELO Mail Login</h2>
        <form id="loginForm">
            <div class="form-group">
                <label for="username">Email ID:(SELO-id@selo.local in small letters)</label>
                <input type="text" id="username" name="username" required>
            </div>
            <div class="form-group">
                <label for="password">Password: (SELO AD password)</label>
                <input type="password" id="password" name="password" required>
            </div>
            <button type="submit" class="btn">Login</button>
        </form>

        <div id="response"></div>

        <!-- Links Section -->
        <div class="links">
            <a href="http://seloappdg.selo.local:96/ChangeADPassword.aspx" target="_blank">Change Password</a>
            <a href="http://seloappdg.selo.local:96/resetpassword.aspx" target="_blank">Reset Password</a>
        </div>
    </div>

    <script>
        document.getElementById('loginForm').addEventListener('submit', function(event) {
            event.preventDefault();

            const username = document.getElementById('username').value;
            const password = document.getElementById('password').value;

            fetch('/zimbra_auth', {
                method: 'POST',
                headers: {
                    'Content-Type': 'text/xml',
                },
                body: `<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
                        <soap:Body>
                            <AuthRequest xmlns="urn:zimbraAccount">
                                <account by="name">${username}</account>
                                <password>${password}</password>
                            </AuthRequest>
                        </soap:Body>
                    </soap:Envelope>`
            })
            .then(response => response.json())
            .then(data => {
                if (data.status === 'failed') {
                    document.getElementById('response').innerHTML = 'Login failed: ' + data.error;
                    document.getElementById('response').style.color = 'red';
                } else {
                    // Redirect to Zimbra URL with token
                    window.location.href = data.redirect_url;
                }
            })
            .catch(error => {
```

```
[root@ha1 ~]# cd /zimbra-login/
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]# cp app.py app.py.org
[root@ha1 zimbra-login]# vi app.py
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]# cd static/
[root@ha1 static]# ls
crpf.jpeg  image.png
[root@ha1 static]# cd ../templates/
[root@ha1 templates]# ls
login.html  login.html-image-ok  login.html-multiserver-ok  login.html-without-redirection  login.html-with-single-server  login.html-zimbra-page-multiserver
[root@ha1 templates]# vi login.html
[root@ha1 templates]#
[root@ha1 templates]#
[root@ha1 templates]#
[root@ha1 templates]#
[root@ha1 templates]# cd
[root@ha1 ~]# tmux
-bash: tmux: command not found
[root@ha1 ~]# yum install tmux
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.

Last metadata expiration check: 0:33:08 ago on Wednesday 11 September 2024 03:57:27 PM IST.
Dependencies resolved.
================================================================================================================================================================
 Package                          Architecture                       Version                                  Repository                              Size
================================================================================================================================================================
Installing:
 tmux                             x86_64                             2.7-1.el8                                BaseOSRepo                              317 k
```

Ln 299, Col 1 | 13,179 characters | 100% | Windows (CRLF) | UTF-8

Install tmux

Yum install tmux

```
================================================================================================================================================================
Installing:
 tmux                             x86_64                             2.7-1.el8                                BaseOSRepo                              317 k

Transaction Summary
================================================================================================================================================================
Install  1 Package

Total size: 317 k
Installed size: 770 k
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                                    1/1
  Installing       : tmux-2.7-1.el8.x86_64                                                                                                              1/1
  Running scriptlet: tmux-2.7-1.el8.x86_64                                                                                                              1/1
  Verifying        : tmux-2.7-1.el8.x86_64                                                                                                              1/1
Installed products updated.

Installed:
  tmux-2.7-1.el8.x86_64

Complete!
[root@ha1 ~]# netstat -ntplu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      979/sshd
tcp6       0      0 :::22                   :::*                    LISTEN      979/sshd
[root@ha1 ~]# cd -
/zimbra-login/templates
[root@ha1 templates]# cd ..
[root@ha1 zimbra-login]# tmux
[detached (from session 0)]
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]#
```

Go into timux

Run python script

Come out from tmux

Ctrl + b then d

vi ne_ldap-import.sh

```
#!/bin/bash
set +x
# LDAP credentials
LDAP_HOST="ldap://ldap1.selo.local:389"
LDAP_PORT="389" # or "636" for LDAPS
BIND_DN="uid=zimbra1,ou=people,dc=selo,dc=local"  # Replace with the correct bind DN
BIND_PASSWORD="123456"
SEARCH_BASE="dc=selo,dc=local"

# Output file
OUTPUT_FILE="/opt/zimbra_ne_users.txt"

# Query LDAP for all users
#ldapsearch -x -H "$LDAP_HOST" -p "$LDAP_PORT" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"
ldapsearch -x -H "$LDAP_HOST" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"

# Filter to get only email addresses
grep "^mail:" "$OUTPUT_FILE" | sed 's/^mail: //' > "/zimbra-login/ne_users.txt"
```

Make two files for NE and OSE

mv ldap-import.sh ne-ldap-import.sh

cp ne-ldap-import.sh ose-ldap-import.sh

vi ose-ldap-import.sh

```
#!/bin/bash
set +x
# LDAP credentials
LDAP_HOST="ldap://ldap5.selo.local:389"
LDAP_PORT="389" # or "636" for LDAPS
BIND_DN="uid=zimbra2,ou=people,dc=selo,dc=local"  # Replace with the correct bind DN
BIND_PASSWORD="123456"
SEARCH_BASE="dc=selo,dc=local"

# Output file
OUTPUT_FILE="/opt/zimbra_ose_users.txt"

# Query LDAP for all users
#ldapsearch -x -H "$LDAP_HOST" -p "$LDAP_PORT" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"
ldapsearch -x -H "$LDAP_HOST" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"

# Filter to get only email addresses
grep "^mail:" "$OUTPUT_FILE" | sed 's/^mail: //' > "/zimbra-login/ose_users.txt"
```

cp * /opt/
cd /opt/

```
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]#
[root@ha1 zimbra-login]# cd
[root@ha1 ~]# cd opt/
[root@ha1 opt]# ls
ldap-import.sh
[root@ha1 opt]# vi ldap-import.sh
[root@ha1 opt]# mv ldap-import.sh ne-ldap-import.sh
[root@ha1 opt]# cp ne-ldap-import.sh ose-ldap-import.sh
[root@ha1 opt]# ls
ne-ldap-import.sh  ose-ldap-import.sh
[root@ha1 opt]# cp * /opt/
[root@ha1 opt]# cd
[root@ha1 ~]# cd /opt/
[root@ha1 opt]# ls
ne-ldap-import.sh  ose-ldap-import.sh  rhel-8.4-x86_64-dvd.iso
[root@ha1 opt]# cd
[root@ha1 ~]# vi single.conf
[root@ha1 ~]# cp single.conf single.conf.org
[root@ha1 ~]# vi single.conf
[root@ha1 ~]# cp single.conf /etc/nginx/conf.d/
[root@ha1 ~]# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[root@ha1 ~]#
[root@ha1 ~]#
[root@ha1 ~]#
[root@ha1 ~]#
[root@ha1 ~]# systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[root@ha1 ~]# systemctl start nginx
[root@ha1 ~]#
[root@ha1 ~]#
```

vi single.conf

```
server {
    listen 80;
    server_name single.tetra.in;
    location / {
        proxy_pass http://127.0.0.1:5000;  # Forward to Flask app running on localhost:5000
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # Hide redirection URL
        proxy_redirect off;
    }

    # Redirect HTTP to HTTPS
#    return 301 https://$host$request_uri;
}
```

cp single.conf single.conf.org

Now copy single.conf to /etc/nginx/conf.d/
cp single.conf /etc/nginx/conf.d/
nginx -t

systemctl start nginx
systemctl enable nginx

ip a

cat single.conf

```
[root@ha1 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:a1:30:f3 brd ff:ff:ff:ff:ff:ff
    inet 172.17.1.225/24 brd 172.17.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fea1:30f3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@ha1 ~]# less /var/log/nginx/error.log
[root@ha1 ~]# cd /etc/nginx/conf.d/
[root@ha1 conf.d]# ls
single.conf
[root@ha1 conf.d]# cat single.conf
server {
    listen 80;
    server_name single.tetra.in;
    location / {
        proxy_pass http://127.0.0.1:5000;  # Forward to Flask app running on localhost:5000
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;

        # Hide redirection URL
        proxy_redirect off;
    }

    # Redirect HTTP to HTTPS
#    return 301 https://$host$request_uri;
}

[root@ha1 conf.d]# vi single.conf
```

cat /opt/ne-ldap-import.sh
Change the parameters accordingly

```
[root@ha1 conf.d]# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[root@ha1 conf.d]# systemctl restart nginx
[root@ha1 conf.d]#
[root@ha1 conf.d]#
[root@ha1 conf.d]#
[root@ha1 conf.d]# cd
[root@ha1 ~]# cat /opt/ne-ldap-import.sh
#!/bin/bash
set +x
# LDAP credentials
LDAP_HOST="ldap://ldap1.selo.local:389"
LDAP_PORT="389" # or "636" for LDAPS
BIND_DN="uid=admin,ou=people,dc=tetrain,dc=com"  # Replace with the correct bind DN
BIND_PASSWORD="4Hp2HgG7p"
SEARCH_BASE="ou=people,dc=selo,dc=local"

# Output file
OUTPUT_FILE="zimbra_users.txt"

# Query LDAP for all users
#ldapsearch -x -H "$LDAP_HOST" -p "$LDAP_PORT" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"
ldapsearch -x -H "$LDAP_HOST" -D "$BIND_DN" -w "$BIND_PASSWORD" -b "$SEARCH_BASE" "(objectClass=zimbraAccount)" mail > "$OUTPUT_FILE"

# Filter to get only email addresses
grep "^mail:" "$OUTPUT_FILE" | sed 's/^mail: //' > "/zimbra-login/ne_users.txt"
```

Install openldap-clients
yum install openldap-clients

```
  Verifying        : openldap-clients-2.4.46-16.el8.x86_64                                                                1/1
Installed products updated.

Installed:
  openldap-clients-2.4.46-16.el8.x86_64

Complete!
[root@ha1 ~]# cd /opt/
[root@ha1 opt]# ls
ne-ldap-import.sh  ose-ldap-import.sh  rhel-8.4-x86_64-dvd.iso
[root@ha1 opt]# vi ne-ldap-import.sh
```

Run the script and check in file.

sh ne-ldap-import.sh
vi ne-ldap-import.sh

sh ne-ldap-import.sh
vi ose-ldap-import.sh

```
[root@ha1 opt]# vi ose-ldap-import.sh
[root@ha1 opt]# cd /zimbra-login/
[root@ha1 zimbra-login]# ls
app.py  app.py.org  ne_users.txt  ose_users.txt  static  templates  venv
```

COMPLETED