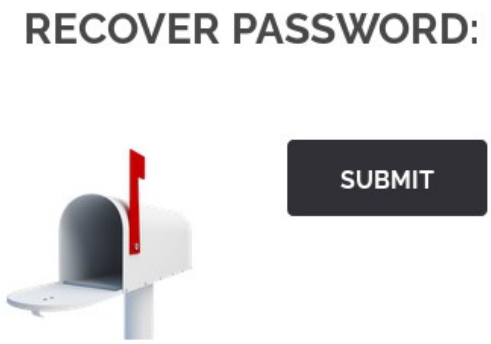
Highjacking password recovery

EXPLICATION

Comme relevé en énumération (voir ENUMERATION – partie 2), une page offre une fonctionnalité de *password recovery*. Aucun input utilisateur n'est demandé, et seul un bouton **Submit** est affiché :



On observe la requête envoyée lorsqu'on clique justement sur Submit. Celle-ci a la forme suivante :

```
1  POST /index.php?page=recover HTTP/1.1
2  Host: 192.168.1.37
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Referer: http://192.168.1.37/index.php?page=recover
8  Content-Type: application/x-www-form-urlencoded
9  Content-Length: 44
10 Cookie: I_am_admin=68934a3e9455fa72420237eb05902327
11 Connection: close
12 Upgrade-Insecure-Requests: 1
13
14 mail=webmaster%40borntosec.com&Submit=Submit
```

L'adresse mail de destination est transmise à l'application via un paramètre du corps de la requête. On voit ici qu'il s'agira toujours du mail d'un administrateur de l'application, ce qui est un peu étrange (comment sait-il quel compte requiert le reset, à quelle adresse mail envoyer le reset, ...).

Une implémentation plus standard d'une fonctionnalité de password reset (c'est celle généralement adoptée, mais peut-être pénible à simuler ici) est la suivante :

1. The user enters their username or email address and submits a password reset request.
2. The website checks that this user exists and then generates a temporary, unique, high-entropy token, which it associates with the user's account on the back-end.
3. The website sends an email to the user that contains a link for resetting their password. The user's unique reset token is included as a query parameter in the corresponding URL: `https://normal-website.com/reset?token=0a1b2c3d4e5f6g7h8i9j`
4. When the user visits this URL, the website checks whether the provided token is valid and uses it to determine which account is being reset. If everything is as expected, the user is given the option to enter a new password. Finally, the token is destroyed.

Source : portswigger

Elle peut être vulnérable à certaines attaques, mais reste relativement sécurisée.

Quoi qu'il en soit, nous sommes, dans le cas de figure de Darkly, capable de contrôler l'adresse mail de destination de la demande de password reset. Si l'adresse mail n'est pas proprement parsée, on peut tout à fait la manipuler afin de recevoir l'email de reset password sur notre adresse mail :
Voir ici différentes techniques (se mettre en cc / bcc, ajouter son email...).

Bref, dans la requête de notre application vulnérable présentée ci-dessus, on modifie l'adresse mail de destination, ce qui nous révèle bien un **flag**.

RESSOURCES
Un script python dans Ressources d'envoyer la requête modifiée, et affiche la page avec le flag.

MITIGATION

> Implémenter le système avec tokens décrit ci-dessus.
> Valider le *user-input* sans se reposer ou faire confiance à des mécanismes de front-end.