Cracking the htpasswd hash

EXPLICATION

Comme relevé au cours de l'énumération (voir Enumeration – partie 9), le directory fuzzing de l'application web nous a révélé des emplacements intéressants, et notamment un fichier htpasswd publiquement accessible au path `/whatever/htpasswd`.

Le hash est le suivant :

```
root:8621ffdbc5698829397d97767ac13db3
```

Ce hash a une longueur de 32 bits, et il ressemble assez fortement à du **MD5**. **Hash-identifier** nous le confirme d'ailleurs.

L'utilisation de MD5 dans le cadre d'une encryption n'est plus aujourd'hui considérée comme sécurisée :
"*Cinq ans plus tard, en 1996, une faille qualifiée de « grave » (possibilité de créer des collisions à la demande) est découverte et indique que MD5 devrait être mis de côté au profit de fonctions plus robustes comme SHA-1. En 2004, une équipe chinoise découvre des collisions complètes. MD5 n'est donc plus considéré comme sûr au sens cryptographique.*" - source wikipedia

Ainsi, on peut tout à fait essayer de cracker ce hash afin de révéler un mot de passe. On a plusieurs options pour cela :

> Utiliser une **rainbow table MD5** en ligne. Ces tables sont simplement d'énormes bases de données associant un texte en clair et son hash MD5. En entrant le hash MD5, ce dernier est comparé aux hash présents dans la base afin d'en déduire le texte en clair associé. Ces rainbow tables semblent assez complètes :

- https://crackstation.net/
- https://md5decrypt.net/

> Utiliser **John The Ripper** ou **Hashcat**, une option classique lorsqu'on essaie d'attaquer un hash. On utilise généralement des dictionnaires, **rockyou.txt** est la base de mots de passe la plus réputée. Typiquement, John a cracké le hash en un peu plus d'une seconde :

```
john --format=Raw-MD5 --wordlist=/usr/share/wordlists/rockyou.txt hash
```

> Utiliser un script d'attaque par dictionnaire personnalisé. On en a crée un très simple, en multithread :

```python
from multiprocessing import Pool
import hashlib
import sys

if (len(sys.argv) != 3) :
    print ('Usage : python3 md5decrypt.py <MD5 HASH> <WORDLIST>')
    sys.exit(1)

user_hash   = sys.argv[1]
filename    = sys.argv[2]
```

```python
def bruteforce_hash(password, md5_hash = user_hash) :
    password = password.rstrip()
    guess = hashlib.md5(password.encode()).hexdigest()
    if (guess == md5_hash) :
        print(f"[+] Found working password : {password}")

with open(filename, 'r', errors='replace') as wordlist :
    passwords = wordlist.readlines()

    with Pool(4) as pool :
        results = pool.map(bruteforce_hash, passwords)
```

Comme indiqué dans le script, il suffit d'indiquer le hash LD5 qu'on souhaite attaquer, et la wordlist qu'on veut utiliser :

```
→  Ressources git:(master) ✗ python3 md5decrypt.py 8621ffdbc5698829397d97767ac13db3 /usr/share/wordlists/rockyou.txt
[+] Found working password : dragon
→  Ressources git:(master) ✗
```

Ce qui fonctionne, puisque le mot de passe est **dragon**. Ce mot de passe peut être utilisé sur le path `/admin/` ; ce qui nous permet d'obtenir un **flag**.

RESSOURCES

Un fichier avec le fichier htpasswd qui contient le hash, et le script **md5decrypt.py** expliqué ci-dessus.

MITIGATION

> Être vigilant sur les politiques de contrôle d'accès des dossiers de l'application, même lorsqu'ils semblent difficilement atteignables.
> Utiliser une encryption sécurisée pour le stockage des mots de passe.