# Computer forensics

Présenté par : Maxime Goyette

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, strings, head, tail...
4. Bless (hex editor)

## ExifTool - Lire ou écrire les métadonnées d'un fichier

Les logiciels de caméra de téléphones intelligents ont tendance à ajouter une énorme quantité de métadonnées aux photos prises.

Par exemple, IMG_20180729_135730.jpg est une photo que j'ai prise avec mon téléphone.

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, strings, head, tail...
4. Bless (hex editor)

```
$ exiftool IMG_20180729_135730.jpg

ExifTool Version Number         : 10.10
File Name                       : IMG_20180729_135730.jpg
Directory                       : .
File Size                       : 5.7 MB
File Modification Date/Time     : 2018:07:29 14:57:08-04:00
File Access Date/Time           : 2018:07:29 14:57:06-04:00
File Inode Change Date/Time     : 2018:07:29 14:57:34-04:00
File Permissions                : rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : ZTE
Camera Model Name               : ZTE B2017G
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Software                        : P852A15-user 6.0.1 MMB29M 20180222.102149 release-keys
Modify Date                     : 2018:07:29 13:57:30
Y Cb Cr Positioning             : Centered
Exposure Time                   : 1/405
F Number                        : 1.9
Exposure Program                : Not Defined
ISO                             : 100
Exif Version                    : 0220
Date/Time Original              : 2018:07:29 13:57:30
Create Date                     : 2018:07:29 13:57:30
Components Configuration        : Y, Cb, Cr, -
Shutter Speed Value             : 1/405
Aperture Value                  : 1.9
Brightness Value                : 0
Metering Mode                   : Center-weighted average
Flash                           : Off, Did not fire
Focal Length                    : 3.7 mm
Maker Note Unknown Text         : auto
Sub Sec Time                    : 297067
Sub Sec Time Original           : 297067
Sub Sec Time Digitized          : 297067
Flashpix Version                : 0100
Color Space                     : sRGB
Exif Image Width                : 4608
Exif Image Height               : 3456
Interoperability Index          : R98 - DCF basic file (sRGB)
Interoperability Version        : 0100
```

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, strings, head, tail...
4. Bless (hex editor)

```
Sensing Method                  : One-chip color area
Scene Type                      : Directly photographed
Exposure Mode                   : Auto
White Balance                   : Auto
Focal Length In 35mm Format     : 4 mm
Scene Capture Type              : Standard
GPS Latitude Ref                : North
GPS Longitude Ref               : West
GPS Altitude Ref                : Above Sea Level
GPS Time Stamp                  : 17:57:28
GPS Processing Method           : ASCII
GPS Date Stamp                  : 2018:07:29
Compression                     : JPEG (old-style)
Thumbnail Offset                : 1082
Thumbnail Length                : 16436
Image Width                     : 4608
Image Height                    : 3456
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Aperture                        : 1.9
GPS Altitude                    : 0 m Above Sea Level
GPS Date/Time                   : 2018:07:29 17:57:28Z
GPS Latitude                    : 45 deg 29' 42.95" N
GPS Longitude                   : 73 deg 33' 49.64" W
GPS Position                    : 45 deg 29' 42.95" N, 73 deg 33' 49.64" W
Image Size                      : 4608x3456
Megapixels                      : 15.9
Scale Factor To 35 mm Equivalent: 1.1
Shutter Speed                   : 1/405
Create Date                     : 2018:07:29 13:57:30.297067
Date/Time Original              : 2018:07:29 13:57:30.297067
Modify Date                     : 2018:07:29 13:57:30.297067
Thumbnail Image                 : (Binary data 16436 bytes, use -b option to extract)
Circle Of Confusion             : 0.028 mm
Field Of View                   : 154.9 deg
Focal Length                    : 3.7 mm (35 mm equivalent: 4.0 mm)
Hyperfocal Distance             : 0.26 m
Light Value                     : 10.5
```

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, strings, head, tail...
4. Bless (hex editor)

## Binwalk et Foremost

Ils servent à extraire des fichiers en identifiant les «magic bytes» de ceux-ci.

Par exemple, une image de type PNG commence toujours par les bytes suivants: 0x89 0x50 0x4E 0x47 0x0D 0x0A 0x1A 0x0A

**Foremost** a moins tendance à trouver les fichiers que **Binwalk**, mais lorsqu'il les extrait, on est certain que c'est bien fait. Au contraire, **Binwalk** trouve à peu près tout, mais c'est souvent du «garbage» et/ou il n'arrive pas bien l'extraire.

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, strings, head, tail...
4. Bless (hex editor)

```
$ binwalk BestPresident.jpg -e

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
85413         0x14DA5         Zip archive data, at least v2.0 to extract, uncompressed size:
556557, name: ToRussia.pdf
631069        0x9A11D         End of Zip archive

$ tree

.
├── BestPresident.jpg
└── _BestPresident.jpg.extracted
    ├── 14DA5.zip
    └── ToRussia.pdf

1 directory, 3 files
```

```
$ foremost BestPresident.jpg

Processing: BestPresident.jpg
|foundat=ToRussia.pdfUT
*|

$ tree

.
├── BestPresident.jpg
└── output
    ├── audit.txt
    ├── jpg
    │   └── 00000000.jpg
    └── zip
        └── 00000166.zip

3 directories, 4 files
```

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, cat, strings, head, tail...
4. Bless (hex editor)

## ls

Liste le contenu d'un dossier.

```
$ ls
$ ls -a

.  ..  .hidden_file  .hidden_folder
```

## cat

Affiche le contenu d'un fichier.

```
$ cat image.png

PNG
♂
IHDR◁◁wS
         IDATx`c```▷◁8UIEND`B`
```

## strings

Affiche les séquences de caractères affichables de longueur 4 ou plus d'un fichier.

```
$ strings image.png

IHDR
IDATx
c```
IEND
```

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, cat, strings, head, tail...
4. Bless (hex editor)

head

Va chercher les lignes ou les caractères au début d'un texte.

```
$ cat /usr/share/dict/words | head -n 5

A
A's
AA's
AB's
ABM's

$ cat /usr/share/dict/words | head -c 5

A
A's
```

tail

Va chercher les lignes ou les caractères à la fin d'un texte.

```
$ cat /usr/share/dict/words | tail -n 5

épée's
épées
étude
étude's
études

$ cat /usr/share/dict/words | tail -c 5

udes
```

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, cat, strings, head, tail...
4. Bless (hex editor)

grep

Affiche les lignes contenant un certain mot.

```
$ cat /usr/share/dict/words | grep hacker

Thackeray
bushwhacker
bushwhacker's
bushwhackers
hacker
hacker's
hackers
```

less

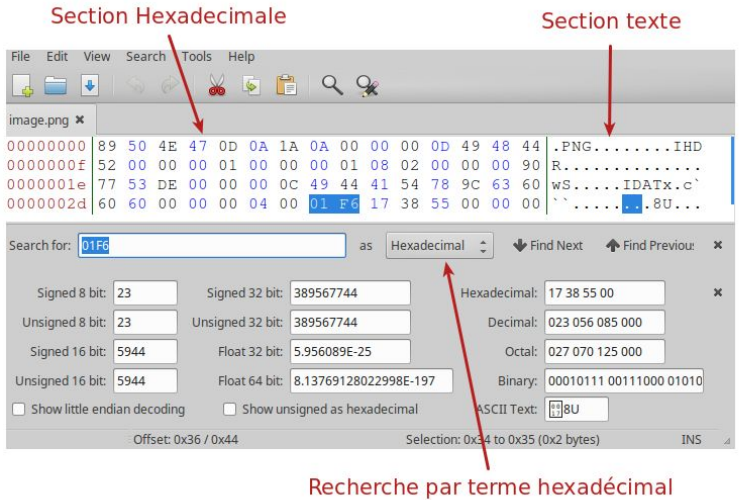
Permet de «scroll» à travers un text dans la console.

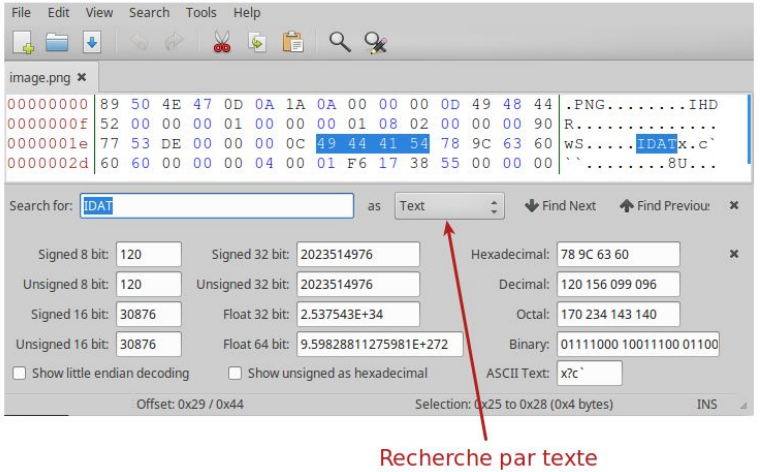
```
$ cat /usr/share/dict/words | less
```

Le «scrolling» se contrôle avec les flèches du clavier.

# Analyse de fichiers

1. ExifTool
2. Binwalk / Foremost
3. ls, cat, strings, head, tail...
4. Bless (hex editor)



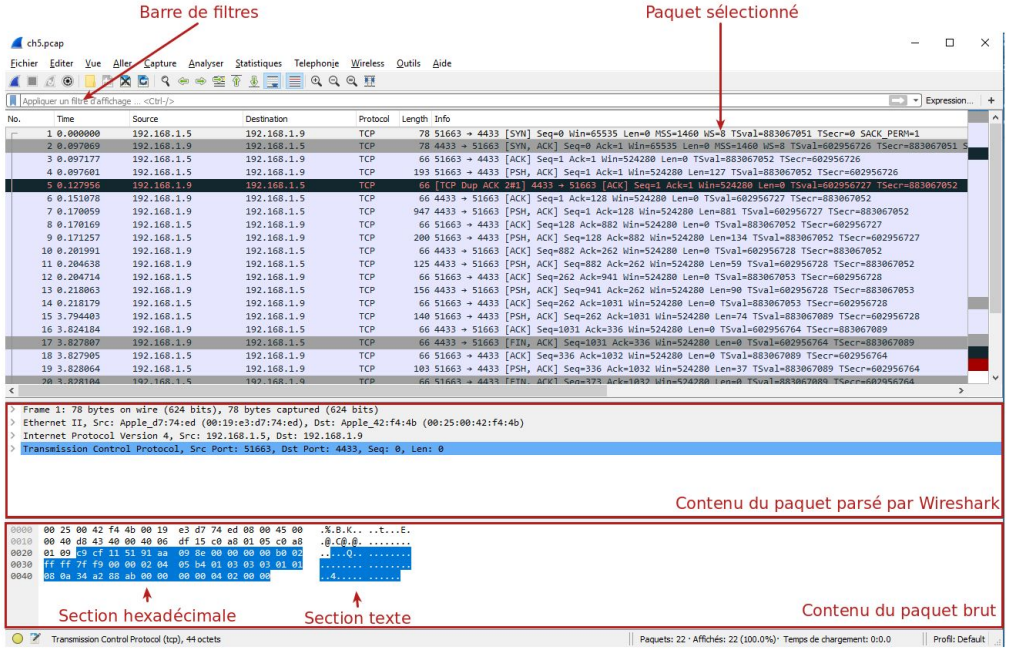



# Analyse de captures réseau

1. Wireshark
2. Tshark
3. PyShark / Scapy

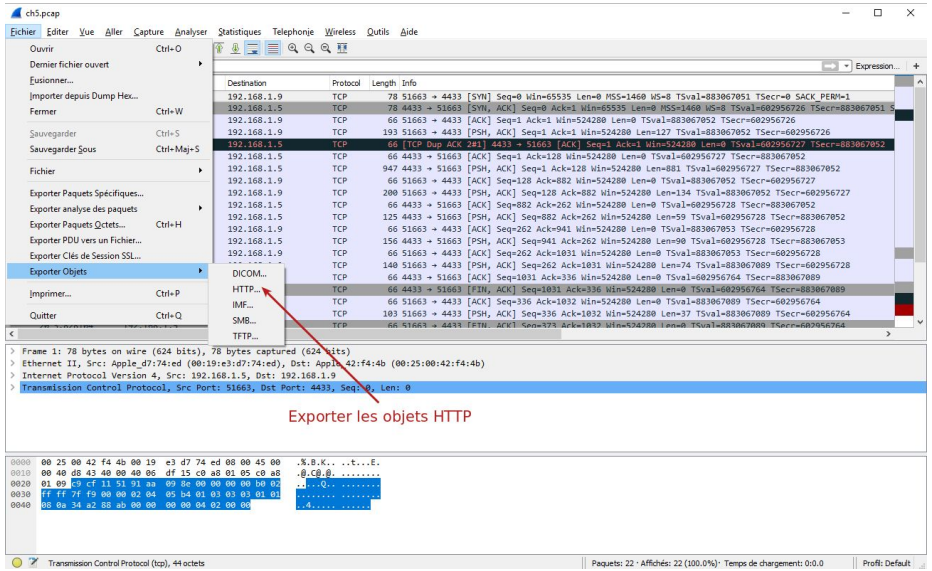
## Wireshark

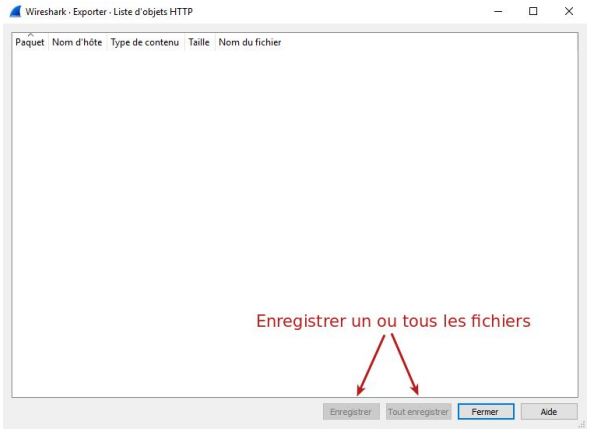
**Wireshark** est un analyseur de paquets libre et gratuit. Il est utilisé dans le dépannage et l'analyse de réseaux informatiques, le développement de protocoles, l'éducation et la rétro-ingénierie.



# Analyse de captures réseau

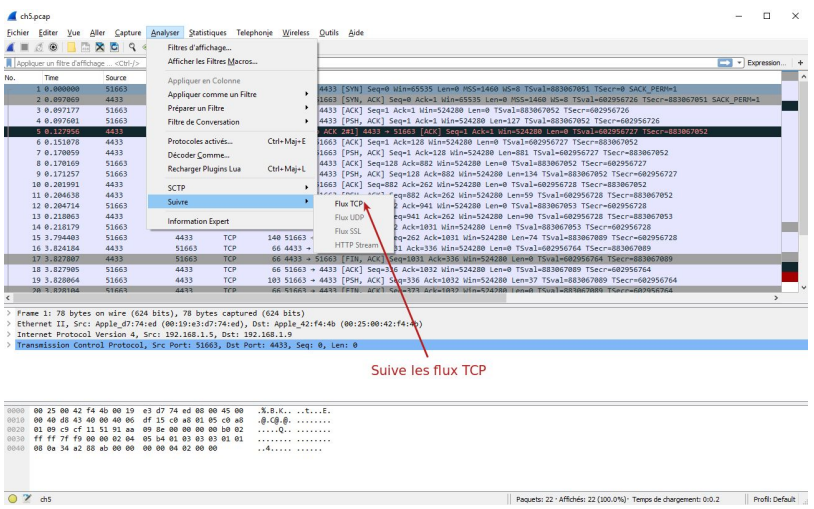
1. Wireshark
2. Tshark
3. PyShark / Scapy

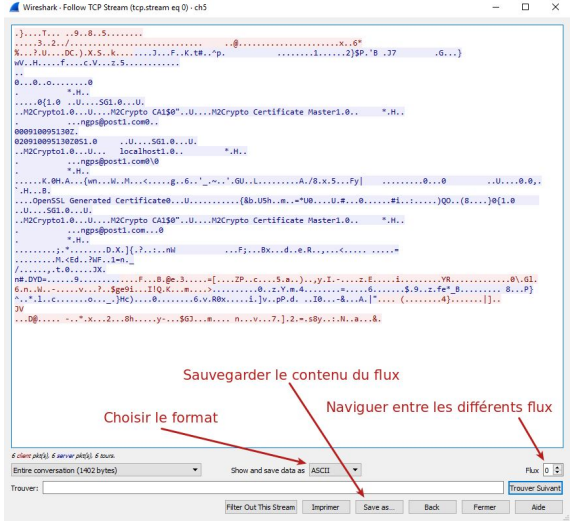


# Analyse de captures réseau

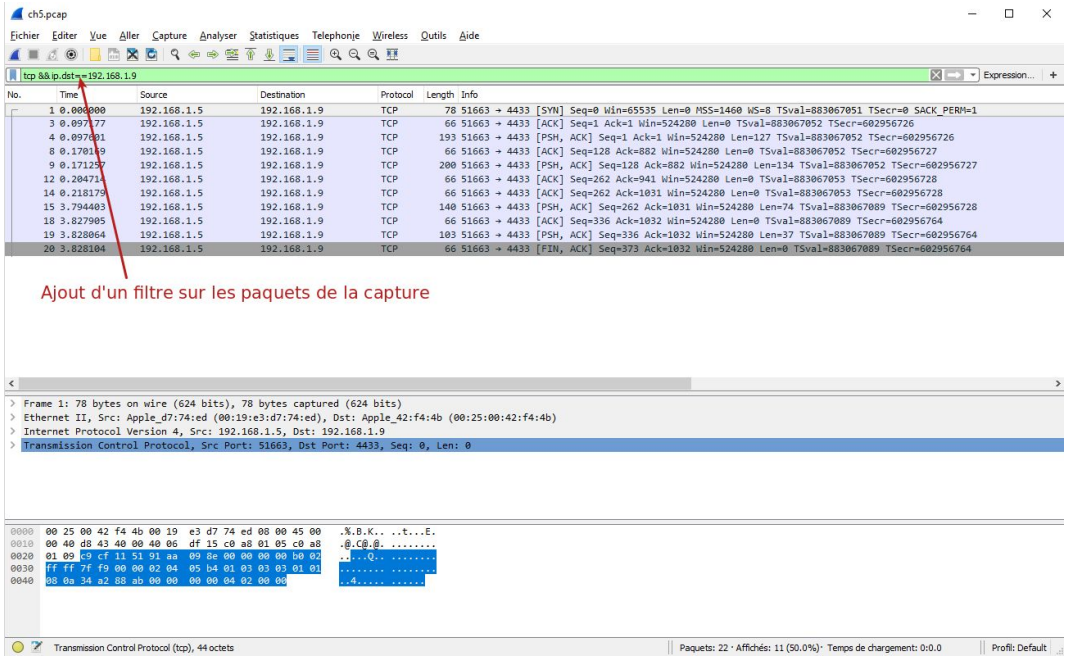
1. Wireshark
2. Tshark
3. PyShark / Scapy



# Analyse de captures réseau

1. Wireshark
2. Tshark
3. PyShark / Scapy



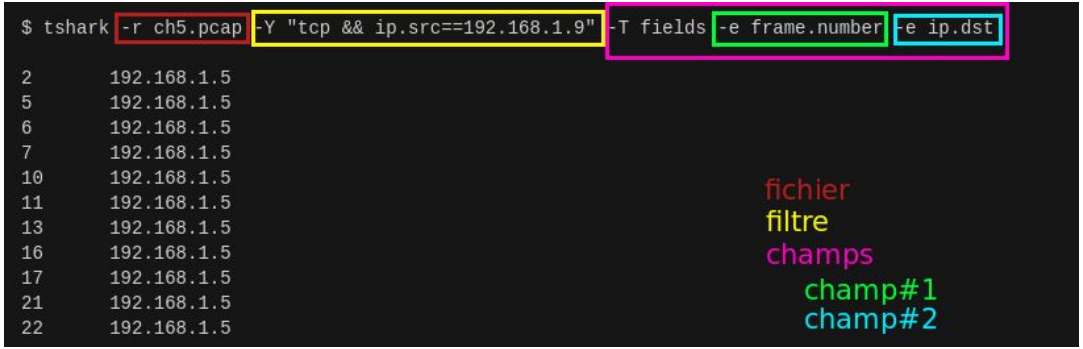
# Analyse de captures réseau

1. Wireshark
2. Tshark
3. PyShark / Scapy

## Tshark

**Tshark** est la version console de **Wireshark**.
Le principal avantage d'utiliser **Tshark** au lieu de **Wireshark** est qu'on peut scripter le travail qu'on veut accomplir.

Utilisation:



# Analyse de captures réseau

1. Wireshark
2. Tshark
3. PyShark / Scapy

Il est parfois difficile de trouver les noms des champs que l'on veut afficher. Puisque la documentation en ligne est assez mauvaise, il est souvent plus rapide de juste afficher les paquets désirés en utilisant le pdml (Packet Details Markup Language).

```
$ tshark -r ch5.pcap -Y "frame.number==1" -T pdml

<?xml version="1.0"?>
...
<pdml>
...
  <proto name="ip" showname="Internet Protocol Version 4, Src: 192.168.1.5, Dst: 192.168.1.9"
size="20" pos="14">
    ...
    <field name="ip.dst" showname="Destination: 192.168.1.9" size="4" pos="30"
show="192.168.1.9" value="c0a80109"/>
  ...
  </proto>
  ...
</packet>

</pdml>
```

Par exemple, ici on peut voir que si on veut avoir l'ip du destinataire, on doit utiliser ip.dst

# Analyse de captures réseau

1. Wireshark
2. Tshark
3. PyShark / Scapy

## Pyshark et Scapy

Pyshark et Scapy sont des librairies python permettant d'analyser les paquets des captures réseaux.

```python
# Ce script affiche les paquets ayant une couche TCP
import pyshark

cap = pyshark.FileCapture('ch5.pcap')

for packet in cap:
    if 'tcp' in packet:
        print(packet)
```

```python
# Ce script affiche les paquets ayant une couche TCP
from scapy.all import *

cap = rdpcap('ch5.pcap')

for packet in cap:
    if packet.haslayer(TCP):
        print(packet.show())
```

- Documentation de Pyshark
- Documentation de Scapy

# Crackage de mots de passe

1. John The Ripper

John The Ripper est utilisé pour effectuer des attaques «brute-force» et «dictionary» sur des hash de mots de passe.

Voici un exemple d'un «dictionary attack» sur un hash SHA1:

```
$ echo -n password | sha1sum | awk '{print $1}' > password_sha1_hash
$ /tools/JohnTheRipper/run/john --wordlist=/tools/rockyou.txt password_sha1_hash

...
Press 'q' or Ctrl-C to abort, almost any other key for status
password         (?)
1g 0:00:00:00 DONE (2018-07-29 22:28) 50.00g/s 400.0p/s 400.0c/s 400.0C/s 123456..rockyou
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Il est aussi possible d'extraire le hash de mot de passe de plusieurs types de fichiers différents

Voici un exemple de crackage de mot de passe d'un fichier ZIP:

```
$ /tools/JohnTheRipper/run/zip2john secrets.zip > secrets.john
$ /tools/JohnTheRipper/run/john --wordlist=/tools/rockyou.txt secrets.john

Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 256/256 AVX2 8x])
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
1234             (secrets.zip)
1g 0:00:00:00 DONE (2018-07-29 22:34) 5.555g/s 91022p/s 91022c/s 91022C/s 123456..christal
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Pour avoir une liste des type de fichiers crackable avec JtR, veuillez vous référer au dossier "/tools/JohnTheRipper/run" sur le Docker.

# Analyse de «dump» de mémoire

1. Volatility Framework

Volatility est un framework implémenté en python sous la GNU General Public License. Il est utilisé pour l'extraction d'artefacts numériques à partir d'un échantillon de mémoire volatile (RAM dump, memory dump, core dump...)

Avant de pouvoir extraire les artefacts numériques, il faut définir le type de profil du memory dump.

```
$ volatility -f memdump imageinfo

Volatility Foundation Volatility Framework 2.5
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win2008R2SP0x64, Win7SP1x64, Win7SP0x64, Win2008R2SP1x64
                     AS Layer1 : AMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/challenges/memdump)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf80002a410a0L
          Number of Processors : 1
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff80002a42d00L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2018-07-24 17:36:38 UTC+0000
     Image local date and time : 2018-07-24 13:36:38 -0400
```

On peut ensuite utiliser les autres commandes en spécifiant le profil :

```
$ volatility -f memdump --profile=Win7SP1x64 <commande>
```

Pour la liste de toutes les commandes disponibles et de leur utilisation, veuillez vous référer à la documentation.