REMnux Tutorial-2: Extraction and Decoding Artifacts

Rhydham Joshi

M.S. in Software Engineering, San Jose State University

Phone : (+1) 408-987-1991 | Email : rhydham.joshi@yahoo.com

Blog : malwareforensics1.blogspot.com | Linkedin : www.linkedin.com/in/rhydhamjoshi

Contents:

REMnux: A Linux Toolkit for Reverse-Engineering and Analyzing Malware

- **REMnux is a free, lightweight Linux (Ubuntu distribution) toolkit for reverse-engineering malicious software.**
- REMnux provides the collection of some of the most common and effective tools used for reverse engineering malwares in categories like:

    1) Investigate Linux malwares
    2) Statically analyze windows executable file
    3) Examine File properties and contents
    4) Multiple sample processing
    5) Memory Snapshot Examination
    6) Extract and decode artifacts
    7) Examine Documents
    8) Browser Malware Examination
    9) Network utilities

- For more information about REMnux, please visit my blog at: http://malwareforensics1.blogspot.com/2015/04/the-power-of-remnux-linux-toolkit-for.html

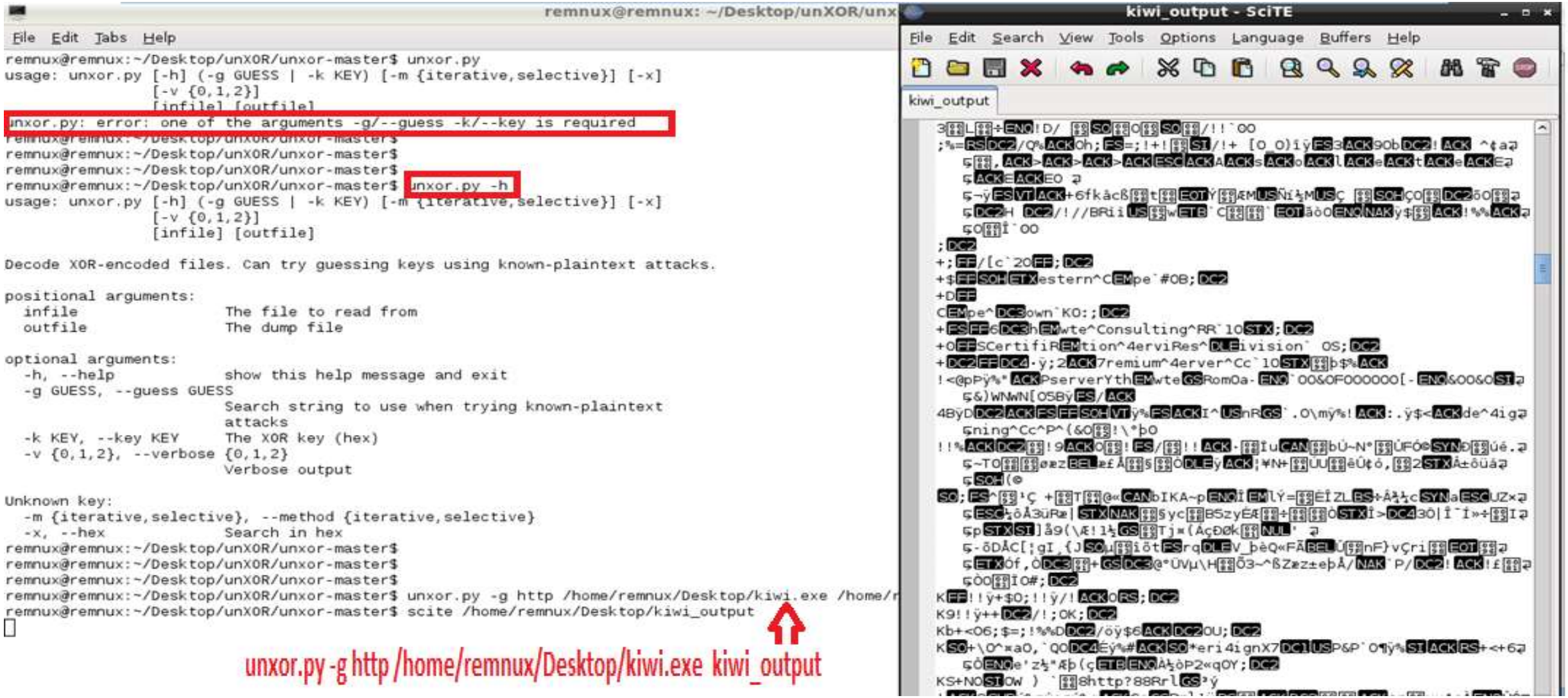
Artifacts:

- Artifact is something observed in a scientific investigation or experiment that is not naturally present but occurs as a result of the preparative or investigative procedure.
- Malwares usually embed themselves with USB devices, malicious JavaScript in HTML pages, a SQL injection attack, email attachment, PDF files, Document files, images etc.
- Malwares exhibits variety of behavior by thoroughly examining the system. I have listed some of the malware anomalies.

  Malware Anomalies:
  - Rogue Processes
  - Unknown services
  - Code injection and root kit behavior
  - Unusual OS artifacts
  - Suspicious network activity
  - Evidence of persistence
- For more information about REMnux, please visit my blog at: http://malwareforensics1.blogspot.com/2015/04/the-power-of-remnux-linux-toolkit-for.html

Deobfuscate and String extraction:-> unXOR

- unXOR try guessing keys until the known-plaintext is found. It requires either key or plaintext argument to keep guessing the keys



Deobfuscate and String extraction:-> xorsearch

XORSearch searches for a given string in an XOR, ROL, ROT or SHIFT encoded binary file.

An XOR encoded binary file is a file where some (or all) bytes have been XORed with a constant value (the key).
A ROL (or ROR) encoded file has its bytes rotated by a certain number of bits (the key).
A ROT encoded file has its alphabetic characters (A-Z and a-z) rotated by a certain number of positions.
A SHIFT encoded file has its bytes shifted left by a certain number of bits (the key):
Shifting is commonly used by malware programmers to obfuscate strings like URLs.

XORSearch will try all XOR keys (0 to 255), ROL keys (1 to 7), ROT keys (1 to 25) and SHIFT keys (1 to 7)

XORSearch does a bruteforce attack with 8-bit keys and smaller. A 32-bit key bruteforce attack would take too long. Option -k instructs XORSearch to do a 32-bit dictionary attack in stead of a 8-bit

-s here will save the XOR file at the default file location with .XOR.00/ff extention

```
remnux@remnux:~$ xorsearch
Usage: XORSearch [-siun] [-l length] [-n length] [-f search-file] file string
XORSearch V1.9.2, search for a XOR, ROL, ROT or SHIFT encoded string in a file
Use -s to save the XOR, ROL, ROT or SHIFT encoded file containing the string
Use -l length to limit the number of printed characters (50 by default)
Use -i to ignore the case when searching
Use -u to search for Unicode strings (limited support)
Use -f to provide a file with search strings
Use -n length to print the length neighbouring charaters (before & after the found keyword)
Use -h to search for hex strings
Use -k to decode with embedded keys
Options -l and -n are mutually exclusive
Options -u and -h are mutually exclusive
Source code put in the public domain by Didier Stevens, no Copyright
Use at your own risk
https://DidierStevens.com
remnux@remnux:~$
```

```
remnux@remnux:~$ xorsearch -s -i /home/remnux/Desktop/k1w1.exe http
Found XOR 00 position 2528: http?88Rrl....!.8..%q.;r.%;.Cc.Rrl.........ar...Ac
Found XOR 00 position A5D4: http://crl.thawte.com/ThawtePremiumServerCA.crl0..
Found XOR 00 position A921: http://cs-g2-crl.thawte.com/ThawteCSG2.crl0...U.%.
Found XOR 00 position A9A9: http://ocsp.thawte.com0...`.H...B........0...*.H..
Found XOR 00 position AB75: http://www.usertrust.com1.0...U....UTN-USERFirst-0
Found XOR 00 position ADE7: http://www.comodogroup.com/repository0B..U...;0907
Found XOR 00 position AE1F: http://crl.usertrust.com/UTN-USERFirst-Object.crl0
Found XOR 00 position B20C: http://www.usertrust.com1.0...U....UTN-USERFirst-0
Found XOR 0B position 8C41: hTTP.2*.beb..nyf..$f..........oa~x..mob..0..0.....
Found XOR 20 position 2528: HTTP...rRL=..*.&.:..Q..R...&cC=rRL..>.&2..&AR...aC
Found XOR 20 position A5D4: HTTP...CRL.THAWTE.COM.tHAWTEpREMIUMsERVERca.CRL.-&
Found XOR 20 position A921: HTTP...CS.G..CRL.THAWTE.COM.tHAWTEcsg..CRL.?&#u=.$
Found XOR 20 position A9A9: HTTP...OCSP.THAWTE.COM.1&)@.h!..b!!$$#"$0.-&)..h..
Found XOR 20 position AB75: HTTP...WWW.USERTRUST.COM.=.;&#u$#34utn.userfIRST.o
Found XOR 20 position ADE7: HTTP...WWW.COMODOGROUP.COM.REPOSITORY.b&#u=?$.....
Found XOR 20 position AE1F: HTTP...CRL.USERTRUST.COM.utn.userfIRST.oBJECT.CRL.
Found XOR 20 position B20C: HTTP...WWW.USERTRUST.COM.=.;&#u$#34utn.userfIRST.o
Found XOR 2B position 8C41: Http-..#BEB__NYF-..F2....-.-#20A^X_#M0B.....o.-..-
remnux@remnux:~$
```

Deobfuscate and String extraction:-> xorsearch

- XORStrings is best described as the combination of my XORSearch tool and the well-known strings command.
- XORStrings will search for strings in the (binary) file , using the same encodings as XORSearch (XOR, ROL, ROT and SHIFT).
- For every encoding/key, XORStrings will search for strings and report the number of strings found, the average string length and the maximum string length.
- Common used words : “HTML”, “This program” & “DOS”

```
remnux@remnux:~$ xorstrings
Usage: XORStrings [options] file
XORStrings V0.0.1, look for XOR, ROL or SHIFT encoded strings in a file
Use -s to save the XOR, ROL or SHIFT encoded file
Use -d to dump the longest string
Use -m sort by maximum string length
Use -l to set the minimum string length (default 5)
Use -t to set the string terminator character, accepts integer or hex number (default 0)
Use -c to output CSV
Use -o to select the operation (XOR, ROL or SHIFT) to perform (to be used together with -k)
Use -k to select the key for the operation to perform (to be used together with -o)
Source code put in the public domain by Didier Stevens, no Copyright
Use at your own risk
https://DidierStevens.com
remnux@remnux:~$
```

Deobfuscate and String extraction:-> xorstrings

Here, XORStrings will do XOR encoding to file with 0x55 key and will save it at the folder where that particular file belong.

```
remnux@remnux:~$ xorstrings /home/remnux/Desktop/kiwi.exe
Opr    Key   Count Avg    Max
  XOR 0xd8       1   5.0      5
  XOR 0xdd       1   8.0      8
  XOR 0x0e       1   6.0      6
  XOR 0xc8       1   5.0      5
  XOR 0x15       1  10.0     10
  XOR 0xbf       1   6.0      6
  XOR 0xbe       1   6.0      6
  XOR 0x18       1   7.0      7
  XOR 0xee       1   7.0      7
  XOR 0xb7       1   5.0      5
  XOR 0x1a       1   5.0      5
  XOR 0xb3       1   5.0      5
  XOR 0xb0       1   5.0      5
SHIFT 0x01       1   6.0      6
  XOR 0xa5       1   5.0      5
  XOR 0xea       1   5.0      5
  XOR 0x98       1   6.0      6
  XOR 0x97       1   5.0      5
  XOR 0xfb       1   5.0      5
  XOR 0x8c       1   6.0      6
  XOR 0xe7       1   5.0      5
  XOR 0x6b       1   5.0      5
  XOR 0x8a       1   5.0      5
  XOR 0x88       1   6.0      6
  XOR 0xf8       1   5.0      5
  XOR 0xfc       2   5.0      5
  XOR 0x9b       2   5.0      5
  XOR 0x32       2   7.5      9
```

OPR : XOR/Shift/ROT/ROL

KEY : Key Value

AVG : Average string length

Max : Maximux string length

```
remnux@remnux:~$ xorstrings -o XOR -k 0x55 -s /home/remnux/Desktop/kiwi.exe
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU
UUUUUUUUU
zVL<!U:'a<;290
d^e\SV
d@eFSV
4%0dGeESV
:";dHeNSV
:;& 9!<;2u66d}esSV
<#<&<:;dteJSV
d^e\SV
d@eFSV
;6{dqewSV
WTUe[SV
QQVWTSeHSV
HpQCeAS]~STPPRVWS]~STPPRVVe|SV
KeId0eMSV
xgxdeeHSV
d^e\SV
d@eFSV
;6{dqewSV
```

-o : selects XOR
-k : selects key
-s : saves it to place where that particular file is stored.

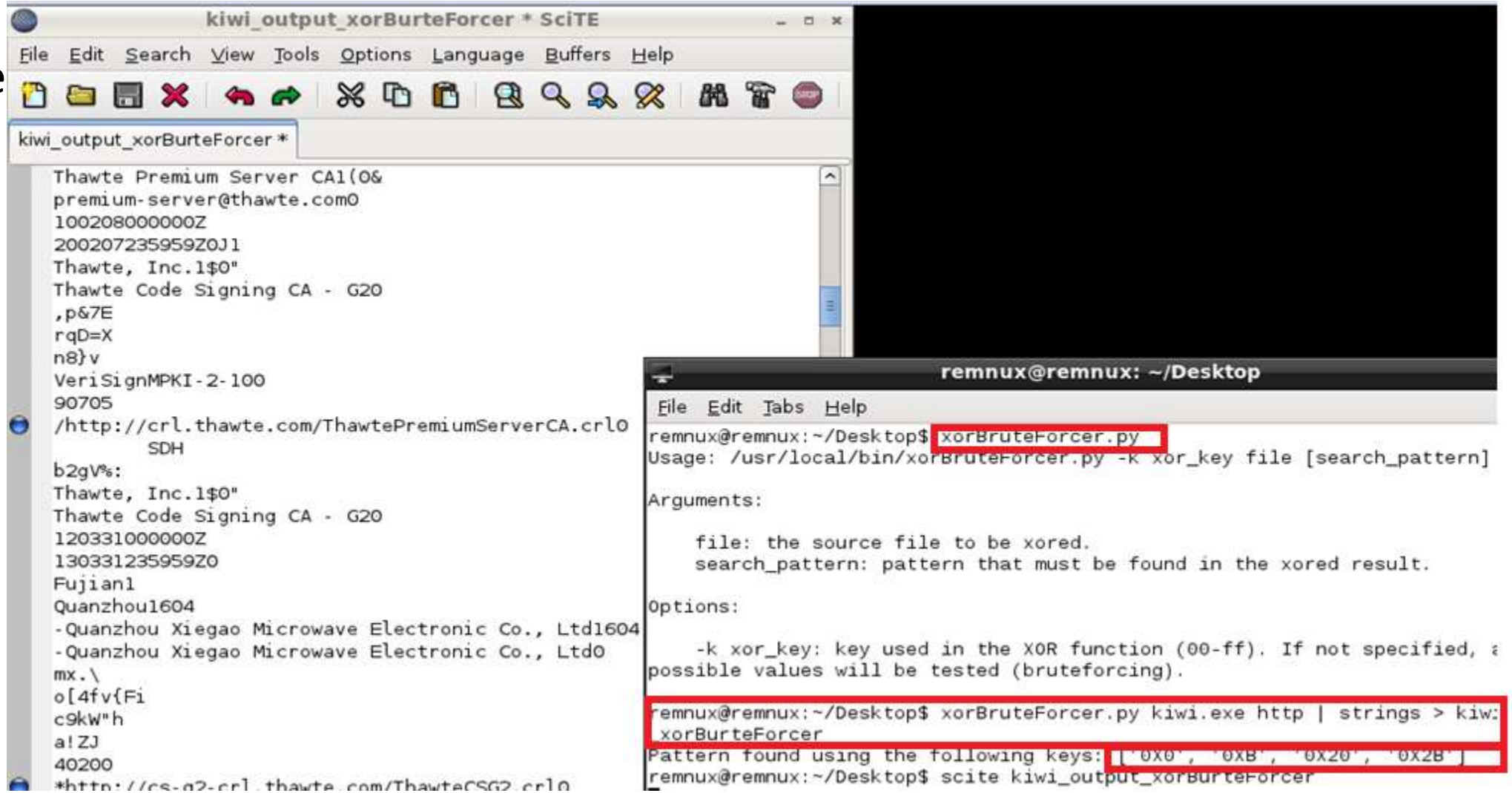
To read that file use below command :

strings /home/remnux/Desktop/kiwi.exe.XOR.55 > output

scite output

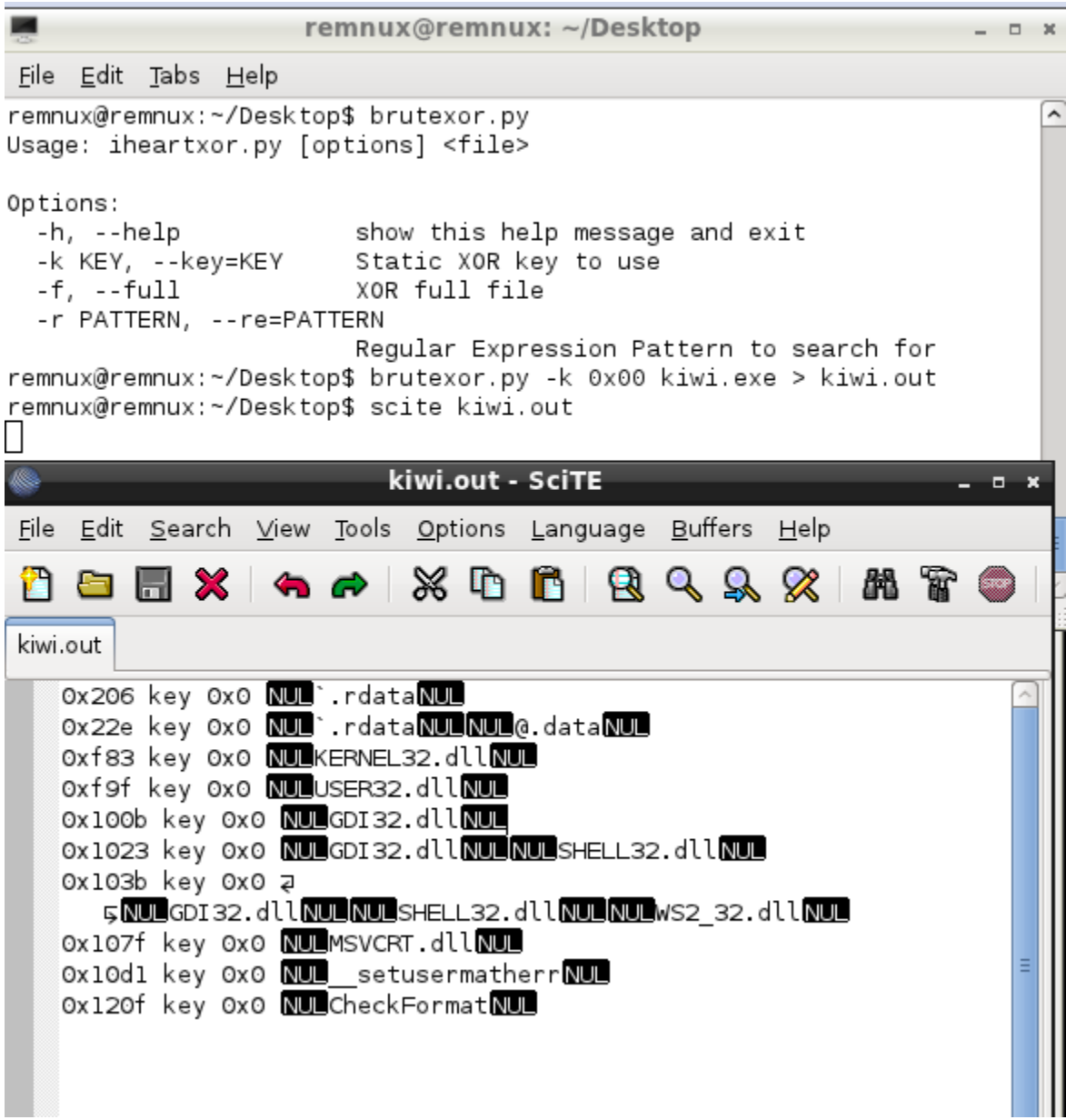
Deobfuscate and String extraction:-> xorBruteForcer

- xorBruteForcer decodes contents of a given file using all possible 1-byte XOR key values. The output of the tool contains lots of noise, xorBruteForcer shows potential string values for all possible 1-byte XOR key values.
- Now, we can use **–k key** to reduce noise



Deobfuscate and String extraction:-> brutexor

- Brutexor brute-forces all possible 1-byte XOR key values and examines the file for strings that might have been encoded with these keys.
- The brutexor tool provides a handy way to brute-force simple XOR keys without looking for any particular string.
- Brutexor shows ASCII data located between null bytes ("\x00") by default.
- Brutexor is very handy provided we know the key found using other tools like, xorsearch, xorstring, xorBruteForcer etc.
- It could do brutexor for full file too using **–f** option.



Deobfuscate and String extraction:-> xortool

- Xortool is very useful in determining key length and in certain cases encrypted key too.
- Here I have used xor.py program to encrypt one pdf file.
- Xortool returns the key length
- By proper combinations or by using trail and error method, (usually 00 for text and 20 for doc), we can even extract exact key

```
remnux@remnux:~/Desktop$ python xor.py -h
****************************************
* Xorpy v1.0                           *
****************************************
* Coded by: Shawn Evans                *
* Email: Shawn.Evans@knowledgeCG.com   *
****************************************

Usage: python xor.py "key" encrypted.txt
$ cat decryptMe.txt | python xor.py "key"
$ cat encryptMe.txt | python xor.py "$eCreTkey"
remnux@remnux:~/Desktop$ cat tiny_guide_to_x86_assembly.pdf | python xor.py "secretkey" > assembly.output
remnux@remnux:~/Desktop$ xortool assembly.output
The most probable key lengths:
   6:
  18:    10.7%
  27:    10.7%
  30:    8.6%
  36:    10.6%
  45:    10.7%
  51:    7.9%
  54:    10.6%
  60:    8.9%
  63:    10.5%
Key-length can be 3*n
Most possible char is needed to guess the key!
remnux@remnux:~/Desktop$ xortool -h
xortool.py
  A tool to do some xor analysis:
  - guess the key length (based on count of equal chars)
  - guess the key (base on knowledge of most probable char)
Usage:
  xortool [-h|--help] [OPTIONS] [<filename>]
Options:
  -l,--key-length      length of the key (integer)
  -c,--char            most possible char (one char or hex code)
  -m,--max-keylen=32   maximum key length to probe (integer)
  -x,--hex             input is hex-encoded str
remnux@remnux:~/Desktop$ xortool -l 9 -c 00 assembly.output
 possible key(s) of length
CESBUTIUY
CUSBUTIUY
```

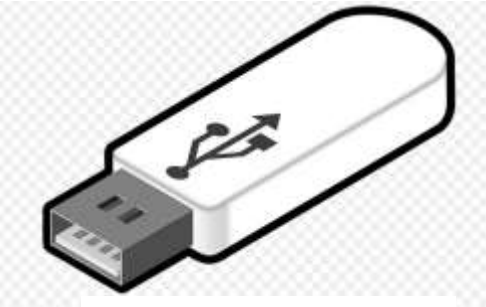
Deobfuscate and String extraction:-> NoMoreXor

- NoMoreXOR attempts to guess XOR 256-byte long XOR key values.
- It uses Yara signatures to determine whether a potential key value worked:
- If the decoded content matches one of the signatures in you file, then probably the key was guessed correctly.
- In that case, the tool deobfuscates corresponding contents and extracts them from the original file.
- NoMoreXOR extracted the deobfuscated contents into the files named **filename.0.unxored** that could be further examined.

```
remnux@remnux:~/Desktop/NoMoreXOR-master$ NoMoreXOR.py -h
usage: NoMoreXOR.py [-h] [-a] [-c] [-xor key] [-g] [-o outfile] [-y YARARULES]
                    Path

Tool to help guess a files 256 byte XOR key by using frequency analysis.

positional arguments:
  Path                  Path to file to be analyzed

optional arguments:
  -h, --help            show this help message and exit
  -a, --analyze         Auto analyze the specified file by looking for all
                        possible XOR keys then apply each of them & scan with
                        YARA to try and determine if it's the correct XOR key
                        (requires an output file)
  -c, --convert         Convert the input file to a hex_file (requires an
                        output file)
  -xor key              XOR the file with the supplied XOR key (requires an
                        output file)
  -g, --guess           Print out information from the hex_file including most
                        common characters and possible SHA256 keys
  -o outfile, --out outfile
                        Name of output file to create
  -y YARARULES, --yararules YARARULES
                        Path to YARA rules to be used during auto analysis if
                        different than what's hardcoded
```

Forensics Investigation practices-> Forensically imaging a Drive

- Forensic investigators often used to investigate devices like compact disk, hard disk, USB, etc.
- The first step is to perform forensic imaging of the drive start the investigation on them insuring no harm or any type of modification to evidence.
- **Make sure to use Write Blocker to perform imagining.**
- Since Linux is dominantly used for performing Investigations. I would be focusing on doing forensic imaging using Linux.
- “Recoverjpeg”, “Foremost”, “Scalpel” for commonly used for file carving.



Forensics Investigation practices-> Forensically imaging a Drive

- Consider we have a suspect USB drive to be forensically examined.
- Connect USB drive to Write blocker and then to Forensic Workstation to convert the suspect drive to *.dd image file.
- Find the drive using **dir /dev/sd*** command and **dmesg | grep sd*** (dmesg lists the Kernel messages including the information about drives)
- The *.dd extension (for archive images, not picture images) is not a single file, but rather an archive in the form of a file.
- Dcfldd (download it using **sudo apt-get install dcfldd**) is an enhanced version for imaging drives. It is an advance version of **dd** developed by the U.S. Department of Defense Computer Forensics Lab.
- It has some useful features for forensic investigators such as:
  - On-the-fly hashing of the transmitted data.
  - Progress bar of how much data has already been sent.
  - Wiping of disks with known patterns.
  - Verification that the image is identical to the original drive, bit-for-bit.
  - Simultaneous output to more than one file/disk is possible.
  - The output can be split into multiple files.
  - Logs and data can be piped into external applications.

Forensics Investigation practices-> Forensically imaging a Drive

```
remnux@remnux:~$ dir /dev/sd*
/dev/sda  /dev/sda1  /dev/sda2  /dev/sda5  /dev/sdb  /dev/sdd  /dev/sdd1
remnux@remnux:~$ dmesg | grep sdd
[47078.134958] sd 6:0:0:0: [sdd] 7900319 512-byte logical blocks: (4.04 GB/3.76 GiB)
[47078.146745] sd 6:0:0:0: [sdd] Write Protect is off
[47078.146759] sd 6:0:0:0: [sdd] Mode Sense: 0b 00 00 08
[47078.158458] sd 6:0:0:0: [sdd] No Caching mode page present
[47078.158472] sd 6:0:0:0: [sdd] Assuming drive cache: write through
[47078.213099] sd 6:0:0:0: [sdd] No Caching mode page present
[47078.213109] sd 6:0:0:0: [sdd] Assuming drive cache: write through
[47078.244369]  sdd: sdd1
[47078.314467] sd 6:0:0:0: [sdd] No Caching mode page present
[47078.314477] sd 6:0:0:0: [sdd] Assuming drive cache: write through
[47078.314483] sd 6:0:0:0: [sdd] Attached SCSI removable disk
remnux@remnux:~$ sudo mkdir /mnt/suspicious
remnux@remnux:~$ sudo mount /dev/sdd1 /mnt/suspicious
remnux@remnux:~$ ls -al /mnt/suspicious
total 30448
drwx------  3 remnux remnux     4096 1969-12-31 19:00 .
drwxr-xr-x 10 root   root       4096 2015-04-19 20:20 ..
-rwxr-xr-x  1 remnux remnux      167 2015-02-09 20:39 abc.bat
-rw-r--r--  1 remnux remnux        0 2015-04-12 19:07 abc.jpeg
-rwxr-xr-x  1 remnux remnux   120832 2014-07-02 12:20 Challenge1.exe
-rw-r--r--  1 remnux remnux  1153949 2015-03-19 22:27 CMPE287-Homework2-Spring-2015.pdf
-rw-r--r--  1 remnux remnux    25097 2015-02-04 19:26 hmk1.rar
-rw-r--r--  1 remnux remnux   111689 2015-01-03 23:04 IMG_1720.JPG
-rw-r--r--  1 remnux remnux   184359 2015-01-04 22:36 IMG_1723.JPG
-rw-r--r--  1 remnux remnux   672549 2015-01-04 22:36 IMG_1724.JPG
-rw-r--r--  1 remnux remnux    77155 2015-01-07 20:27 IMG_1748.JPG
-rw-r--r--  1 remnux remnux    77970 2015-01-09 00:00 IMG_1752.JPG
-rw-r--r--  1 remnux remnux    83014 2015-01-09 23:52 IMG_1763.JPG
-rw-r--r--  1 remnux remnux   245074 2015-03-17 23:56 IMG_2840.JPG
-rw-r--r--  1 remnux remnux 24155361 2014-09-03 21:39 Intro to REST.mp4
-rw-r--r--  1 remnux remnux   297845 2015-03-07 12:28 Manual Detection and Mitigation of Malwares in Windows OS.pdf
-rw-r--r--  1 remnux remnux      953 2014-12-30 00:30 MinGW Installer.lnk
-rw-r--r--  1 remnux remnux    97307 2015-03-09 00:24 Security_TestCases_MTP_5_x_12Aug14.xlsx
-rw-r--r--  1 remnux remnux    79298 2015-04-05 19:16 Sysinternals diagram.png
drwx------  2 remnux remnux     4096 2015-02-16 08:09 System Volume Information
```

To list all the devices attached

dmesg shows kernel messgaes | grep limits it to display messages containing SDD

sdd1 contains pen-drive

Mkdir creates a folder

mount mounts device to folder

umount /mnt/suspicous unmounts the connection

Forensics Investigation practices->Dcfldd

```
remnux@remnux:~/suspicious$ sudo dcfldd if=/dev/sdd1 hash=md5,sha256 hashwindow=10G md5log=md5.txt sha256log=sha256.txt  hash
conv=after bs=512 conv=noerror,sync split=10G splitformat=aa of=driveimage.dd
87552 blocks (42Mb) written.
```

- **man dcfldd** or **dcfldd --help** provides many options that can be used to format the output
- Here, Input file is /dev/sdd1 and output file is located at /suspicious
- This command will read ten Gigabytes from the source drive and write that to a file called driveimage.dd.aa
- It will then read the next ten gigs and name that driveimage.dd.ab.
- It will also calculate the MD5 hash and the sha256 hash of the ten Gigabyte chunk.
- The md5 hashes will be stored in a file called md5.txt and the sha256 hashes will be stored in a file called sha256.txt.
- The block size for transferring has been set to 512 bytes, and in the event of read errors, dcfldd will write zeros.

Forensics & File Carving -> Foremost

- Foremost is a console program to recover files based on their headers, footers, and internal data structures.
- Foremost can work on image files, such as those generated by dd, Safeback, Encase, etc, or directly on a drive. foremost can search through most any kind of data without worrying about the format.
- Foremost is designed to ignore the type of underlying filesystem and directly read and copy portions of the drive into the computer's memory.
- It takes these portions one segment at a time, and using a process known as file carving searches this memory for a file header type that matches the ones found in Foremost's configuration file.
- When a match is found, it writes that header and the data following it into a file, stopping when either a footer is found, or until the file size limit is reached.
- The headers and footers can be specified by a configuration file or you can use command line switches to specify built-in file types.
- These built-in types look at the data structures of a given file format allowing for a more reliable and faster recovery.
- Foremost served as the basis for Scalpel, a significantly faster program to also recover deleted files & to perform forensic investigation on device images.

Forensics & File Carving -> Foremost

```
remnux@remnux:~/suspicious/foremostfiles$ foremost -h
foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus.
$ foremost [-v|-V|-h|-T|-Q|-q|-a|-w-d] [-t <type>] [-s <blocks>] [-k <size>]
        [-b <size>] [-c <file>] [-o <dir>] [-i <file]

  -V  - display copyright information and exit
  -t  - specify file type.  (-t jpeg,pdf ...)
  -d  - turn on indirect block detection (for UNIX file-systems)
  -i  - specify input file (default is stdin)
  -a  - Write all headers, perform no error detection (corrupted files)
  -w  - Only write the audit file, do not write any detected files to the disk
  -o  - set output directory (defaults to output)
  -c  - set configuration file to use (defaults to foremost.conf)
  -q  - enables quick mode. Search are performed on 512 byte boundaries.
  -Q  - enables quiet mode. Suppress output messages.
  -v  - verbose mode. Logs all messages to screen
remnux@remnux:~/suspicious/foremostfiles$ scite /etc/foremost.conf
remnux@remnux:~/suspicious/foremostfiles$ scite /etc/foremost.conf
remnux@remnux:~/suspicious/foremostfiles$ foremost -t jpeg,pdf,exe,doc,wav,docx,rar,png,xlsx
-i /home/remnux/suspicious/driveimage.dd.aa -o /home/remnux/suspiciousfiles
```

```
remnux@remnux:~/suspicious/foremostfiles$ sudo chmod -R 777 /home/remnux/suspiciousfile
```

Note: Please go through foremost.conf file located at /etc/foremost.conf

Folders created by foremost :

Forensics & File Carving -> Foremost

Audit.txt

```
Foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus
Audit File

Foremost started at Sun Apr 19 21:30:30 2015
Invocation: foremost -i driveimage.dd.aa -o foremostfiles/
Output directory: /home/remnux/suspicious/foremostfiles
Configuration file: /etc/foremost.conf
-------------------|------------------------------------------
File: driveimage.dd.aa
Start: Sun Apr 19 21:30:30 2015
Length: 3 GB (4044922368 bytes)

Num      Name (bs=512)         Size      File Offset     Comment

0:      00259434.exe          96 KB       132830208      02/03/2009 07:21:06
1:      03161162.jpg          81 KB      1618514944
2:      03161330.jpg          76 KB      1618600960
3:      03161490.jpg          75 KB      1618682880
4:      03161642.jpg         180 KB      1618760704
5:      03162010.jpg         656 KB      1618949120
6:      03163330.jpg         109 KB      1619624960
7:      03163562.jpg         186 KB      1619743980
8:      03164004.jpg         134 KB      1619970187
9:      03164282.jpg           5 KB      1620112474
10:     03164325.jpg          25 KB      1620134866
11:     03164382.jpg          60 KB      1620163989
12:     03165053.jpg         380 KB      1620507366
117:    04526501.png           3 KB      2317568758      (30 x 30)
118:    04526507.png           3 KB      2317571988      (42 x 42)
119:    04526514.png           3 KB      2317575352      (54 x 54)
120:    04526521.png           3 KB      2317578884      (24 x 24)
121:    04526697.png           3 KB      2317669294      (150 x 150)
122:    04526705.png           4 KB      2317673344      (210 x 210)
123:    04526725.png           3 KB      2317683252      (120 x 120)
124:    04527364.png           3 KB      2318010557      (70 x 70)
125:    04527371.png           3 KB      2318014019      (98 x 98)
126:    04527378.png           3 KB      2318017714      (126 x 126)
127:    04527386.png           3 KB      2318021691      (56 x 56)
128:    04527411.png           4 KB      2318034896      (310 x 150)
129:    04527440.png           4 KB      2318049672      (248 x 120)
Finish: Sun Apr 19 21:35:31 2015

130 FILES EXTRACTED

jpg:= 16
rif:= 1
zip:= 2
rar:= 1
exe:= 2
png:= 105
pdf:= 3
-----------------------------------------------------------------

Foremost finished at Sun Apr 19 21:35:31 2015
```

Foremost parses the entire image and those files which left their traces(not wiped out securely) will get detected

Forensics & File Carving -> Scalpel

- Scalpel is an open source program for recovering deleted data originally based on foremost, although significantly more efficient. It runs on Linux and Windows.
- The tool visits the block database storage and identifies the deleted files from it and recover them instantly.
- Apart from file recovery it is also useful for digital forensics investigation.
- By default scalpel utility has its own configuration file in ‘/etc‘ directory and full path is **“/etc/scalpel/scalpel.conf**” or “**/etc/scalpel.conf**“.
- **Everything is commented out (#) by default**. So before running scalpel one needs to uncomment the file format that one want to recover.
- However uncomment the entire file is time consuming and will generate a huge false results.
- Installation comand: **sudo apt-get install scalpel**

Forensics & File Carving -> Scalpel

Default Parameters :

```
Scalpel version 1.60
Written by Golden G. Richard III, based on Foremost 0.69.
Carves files from a disk image based on file headers and footers.

Usage: scalpel [-b] [-c <config file>] [-d] [-h|V] [-i <file>]
               [-m blocksize] [-n] [-o <outputdir>] [-0 num] [-q clustersize]
               [-r] [-s num] [-t <blockmap file>] [-u] [-v]
               <imgfile> [<imgfile>] ...

-b  Carve files even if defined footers aren't discovered within
    maximum carve size for file type [foremost 0.69 compat mode].
-c  Choose configuration file.
-d  Generate header/footer database; will bypass certain optimizations
    and discover all footers, so performance suffers.  Doesn't affect
    the set of files carved.  **EXPERIMENTAL**
-h  Print this help message and exit.
-i  Read names of disk images from specified file.
-m  Generate/update carve coverage blockmap file.  The first 32bit
    unsigned int in the file identifies the block size. Thereafter
    each 32bit unsigned int entry in the blockmap file corresponds
    to one block in the image file.  Each entry counts how many
    carved files contain this block. Requires more memory and
    disk.  **EXPERIMENTAL**
-n  Don't add extensions to extracted files.
-o  Set output directory for carved files.
-0  Don't organize carved files by type. Default is to organize carved files
    into subdirectories.
-p  Perform image file preview; audit log indicates which files
    would have been carved, but no files are actually carved.
-q  Carve only when header is cluster-aligned.
-r  Find only first of overlapping headers/footers [foremost 0.69 compat mode].
-s  Skip n bytes in each disk image before carving.
-t  Set directory for coverage blockmap.  **EXPERIMENTAL**
-u  Use carve coverage blockmap when carving.  Carve only sections
    of the image whose entries in the blockmap are 0.  These areas
    are treated as contiguous regions.  **EXPERIMENTAL**
-V  Print copyright information and exit.
-v  Verbose mode.
remnux@remnux:~/suspicious$
```

This is not the option to pass drive parameter

Forensics & File Carving -> Scalpel

```
remnux@remnux:~$ sudo dcfldd if=/dev/sdb1 of=/home/remnux/suspicious/pen_image.dd
57856 blocks (1808Mb) written.
```

- Here, I uncommented for Java, zip, .dat, wav, rpm, pdf, html, doc, avi, bmp, png file formats.

```
remnux@remnux:~$ cd /home/remnux/suspicious/
remnux@remnux:~/suspicious$ ls
pen_image.dd  scalpel.conf
remnux@remnux:~/suspicious$ scite scalpel.conf
remnux@remnux:~/suspicious$ scalpel -c /home/remnux/suspicious/scalpel.conf -o /
home/remnux/suspicious/scalpel_extract  /home/remnux/suspicious/pen_image.dd
Scalpel version 1.60
Written by Golden G. Richard III, based on Foremost 0.69.

Opening target "/home/remnux/suspicious/pen_image.dd"

Image file pass 1/2.
/home/remnux/suspicious/pen_image.dd: 100.0% |*****************************************************************************************
Allocating work queues...
Work queues allocation complete. Building carve lists...
Carve lists built.  Workload:
jpg with header "\xff\xd8\xff\xe0\x00\x10" and footer "\xff\xd9" --> 10 files
png with header "\x50\x4e\x47\x3f" and footer "\xff\xfc\xfd\xfe" --> 6 files
pdf with header "\x25\x50\x44\x46" and footer "\x25\x45\x4f\x46\x0d" --> 3 files
pdf with header "\x25\x50\x44\x46" and footer "\x25\x45\x4f\x46\x0a" --> 0 files
rpm with header "\xed\xab" and footer "" --> 62526 files
dat with header "\x72\x65\x67\x66" and footer "" --> 3 files
dat with header "\x43\x52\x45\x47" and footer "" --> 0 files
zip with header "\x50\x4b\x03\x04" and footer "\x3c\xac" --> 53 files
java with header "\xca\xfe\xba\xbe" and footer "" --> 0 files
Carving files from image.
Image file pass 2/2.
/home/remnux/suspicious/pen_image.dd:  13.7% |************
/home/remnux/suspicious/pen_image.dd:  15.6% |********                                                    |  600.0 MB    12:58 ETA
```

No argument for passing drive

We need to give recursive permission to folder to access the content

```
remnux@remnux:~/suspicious$ sudo chmod -R 777 /home/remnux/suspicious/scalpel_extract/
[sudo] password for remnux:
remnux@remnux:~/suspicious$
```

Forensics & File Carving -> Scalpel

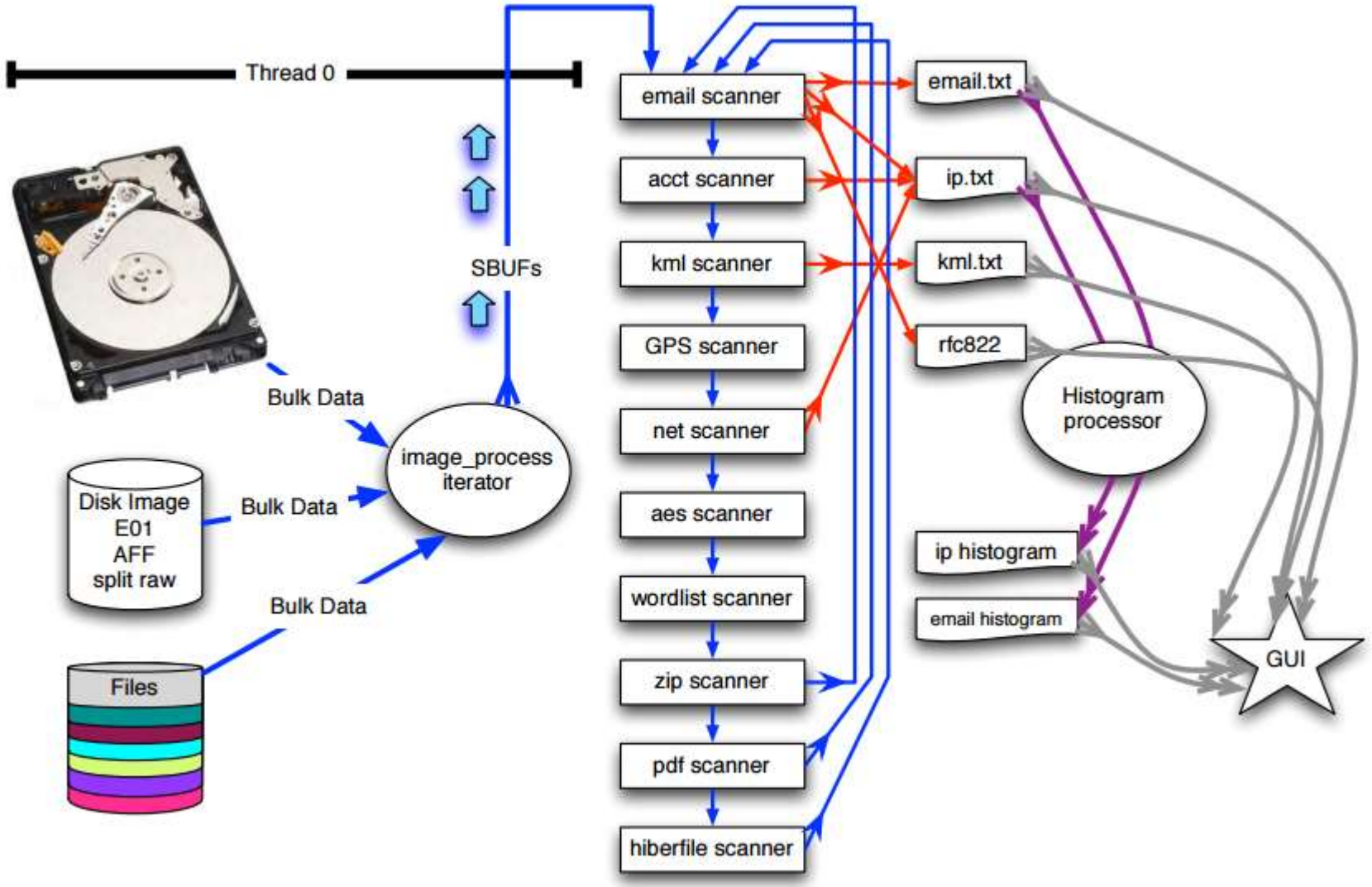
audit.txt

```
Scalpel version 1.60 audit file
Started at Mon Apr 20 05:23:02 2015
Command line:
scalpel -c /home/remnux/suspicious/scalpel.conf -o /home/remnux/suspicious/scalpel_extract /home/remnux/
suspicious/pen_image.dd

Output directory: /home/remnux/suspicious/scalpel_extract
Configuration file: /home/remnux/suspicious/scalpel.conf

Opening target "/home/remnux/suspicious/pen_image.dd"

The following files were carved:
File              Start                Chop          Length          Extracted From
00000035.rpm      9445522              YES           1000000         pen_image.dd
00000034.rpm      9353266              YES           1000000         pen_image.dd
00000033.rpm      9346463              YES           1000000         pen_image.dd
00000032.rpm      9265472              YES           1000000         pen_image.dd
00000031.rpm      8946419              YES           1000000         pen_image.dd
00000030.rpm      8753802              YES           1000000         pen_image.dd
00000029.rpm      8641591              YES           1000000         pen_image.dd
00000028.rpm      8500473              YES           1000000         pen_image.dd
00000027.rpm      8457263              YES           1000000         pen_image.dd
00000026.rpm      8456585              YES           1000000         pen_image.dd
00000025.rpm      8421906              YES           1000000         pen_image.dd
```

Forensics & File Carving -> Bulk_Extractor

- bulk_extractor is a multi-threaded program that extracts features such as email addresses, credit card numbers, URLs, and other types of information from digital evidence files, disk image, or directory of files.
- It is a useful forensic investigation tool for many tasks such as malware and intrusion investigations, identity investigations and cyber investigations, as well as analyzing imagery and password cracking.
- The results can be easily inspected, parsed, or processed with automated tools.
- bulk_extractor automatically detects, decompresses, and recursively re-processes compressed data that is compressed with a variety of algorithms.
- It can process compressed data (like ZIP, PDF and GZIP files) and incomplete or partially corrupted data.
- It can carve JPEGs, office documents and other kinds of files out of fragments of compressed data. It will detect and carve encrypted RAR files, XOR files etc.
- It builds word lists based on all of the words found within the data, even those in compressed files that are in unallocated space. Those word lists can be useful for password cracking.
- It creates histograms showing the most common email addresses, URLs, domains, search terms and other kinds of information on the drive.
- In addition to the capabilities described above, bulk_extractor also includes:
  - A graphical user interface, Bulk Extractor Viewer, for browsing features stored in feature files and for launching bulk_extractor scans
  - A small number of python programs for performing additional analysis on feature files

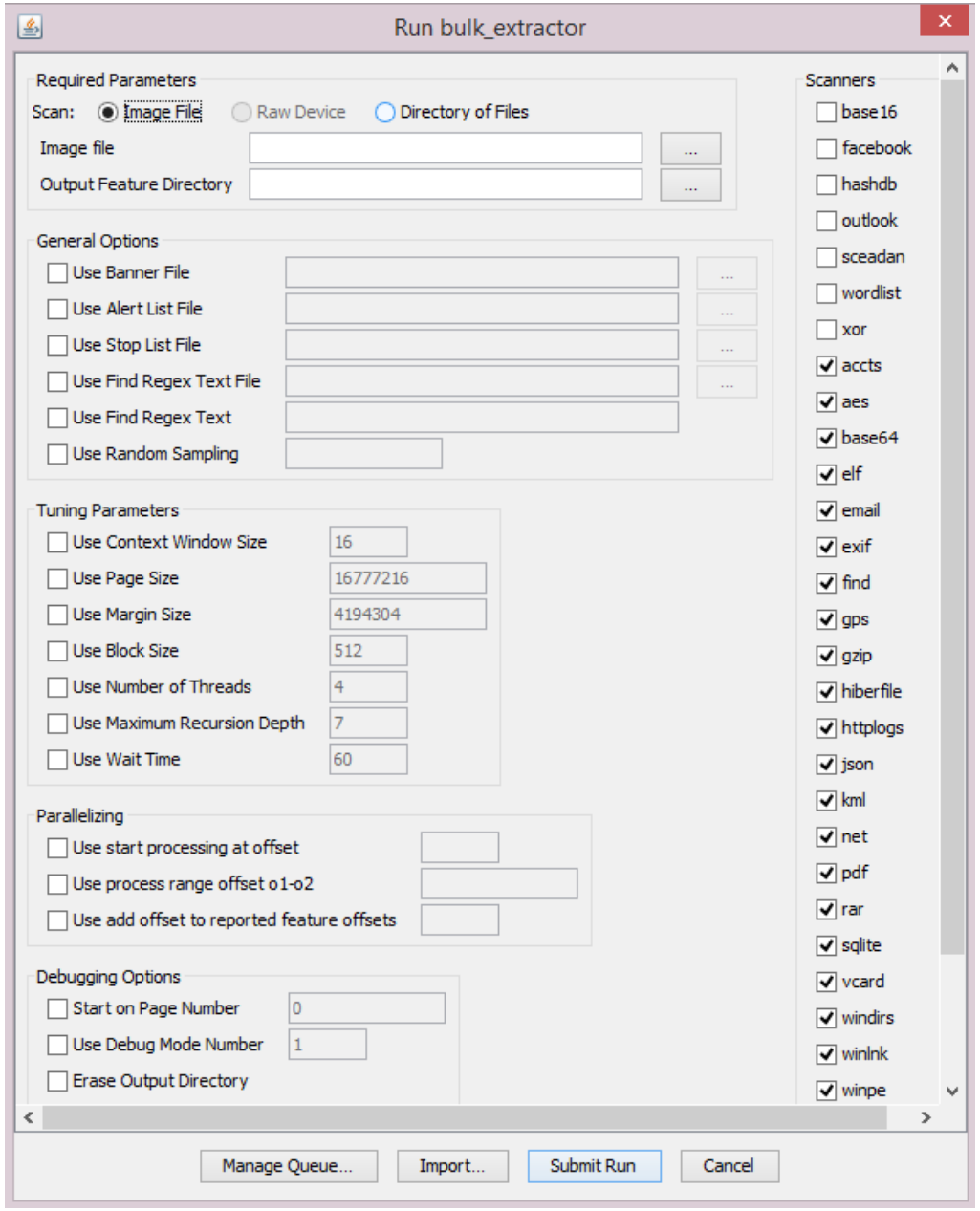
Forensics & File Carving -> Bulk_Extractor



Forensics & File Carving -> Bulk Extractor

```
$ bulk_extractor -h
bulk_extractor version 1.3b6 $Rev: 10046 $
Usage: src/bulk_extractor [options] imagefile
  runs bulk extractor and outputs to stdout a summary of what was found where

Required parameters:
   imagefile     - the file to extract
 or  -R filedir  - recurse through a directory of files
                 SUPPORT FOR E01 FILES COMPILED IN
                 SUPPORT FOR AFF FILES COMPILED IN
                 EXIV2 COMPILED IN
   -o outdir    - specifies output directory. Must not exist.
                 bulk_extractor creates this directory.
Options:
   -b banner.txt- Add banner.txt contents to the top of every output file.
   -r alert_list.txt  - a file containing the alert list of features to alert
                      (can be a feature file or a list of globs)
                      (can be repeated.)
   -w stop_list.txt   - a file containing the stop list of features (white list
                      (can be a feature file or a list of globs)s
                      (can be repeated.)
   -F <rfile>   - Read a list of regular expressions from <rfile> to find
   -f <regex>   - find occurances of <regex>; may be repeated.
                 results go into find.txt
   -q nn        - Quiet Rate; only print every nn status reports. Default 0; -1 for no status

Tuning parameters:
   -C NN        - specifies the size of the context window (default 16)
   -G NN        - specify the page size (default 4194304)
   -g NN        - specify margin (default 4194304)
   -W n1:n2      - Specifies minimum and maximum word size
                (default is -w6:14)
   -B NN        - Specify the blocksize for bulk data analysis (default 512)
   -j NN        - Number of threads to run (default 8)
   -M nn        - sets max recursion depth (default 5)

Path Processing Mode:
   -p <path>/f  - print the value of <path> with a given format.
                 formats: r = raw; h = hex.
                 Specify -p - for interactive mode.
                 Specify -p -http for HTTP mode.

Parallelizing:
   -Y <o1>      - Start processing at o1 (o1 may be 1, 1K, 1M or 1G)
   -Y <o1>-<o2> - Process o1-o2
   -A <off>     - Add <off> to all reported feature offsets

Debugging:
   -h           - print this message
   -H           - print detailed info on the scanners
   -V           - print version number
   -z nn        - start on page nn
   -dN          - debug mode (see source code
   -Z           - zap (erase) output directory

Control of Scanners:
   -P <dir>     - Specifies a plugin directory
   -E scanner   - turn off all scanners except scanner
   -m <max>     - maximum number of minutes to wait for memory starvation
                default is 60
   -s name=value - sets a bulk extractor option name to be value
   -e bulk - enable scanner bulk
   -e exiv2 - enable scanner exiv2
   -e wordlist - enable scanner wordlist
   -x accts - disable scanner accts
   -x aes - disable scanner aes
   -x base16 - disable scanner base16
   -x base64 - disable scanner base64
   -x elf - disable scanner elf
   -x email - disable scanner email
   -x exif - disable scanner exif
   -x gps - disable scanner gps
   -x gzip - disable scanner gzip
   -x hiber - disable scanner hiber
   -x json - disable scanner json
   -x kml - disable scanner kml
   -x net - disable scanner net
   -x pdf - disable scanner pdf
   -x vcard - disable scanner vcard
   -x windirs - disable scanner windirs
   -x winpe - disable scanner winpe
   -x winprefetch - disable scanner winprefetch
   -x zip - disable scanner zip
$
```

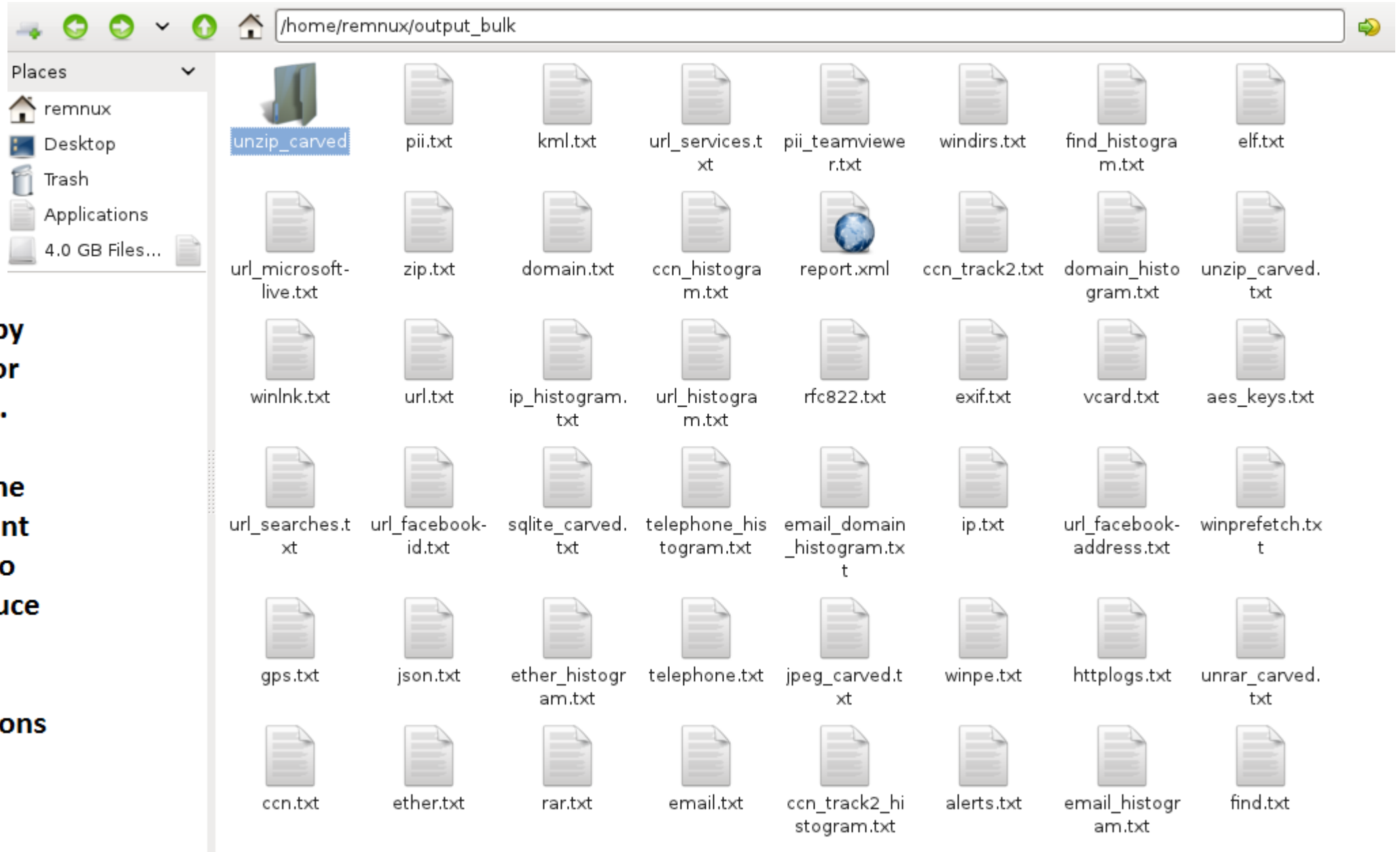
Forensics & File Carving -> Bulk_Extractor



Bulk-extractor is available in GUI and command line mode. Using Bulk-Extractor with command line is equally easy like GUI

```
remnux@remnux:~$ sudo bulk_extractor -o output_bulk /dev/sdb1
bulk_extractor version: 1.5.5
Hostname: remnux
Input file: /dev/sdb1
Output directory: output_bulk
Disk Size: 4044922368
Threads: 1
Attempt to open /dev/sdb1
20:46:46 Offset 67MB (1.66%) Done in  1:03:30 at 21:50:16
20:47:59 Offset 150MB (3.73%) Done in  0:58:47 at 21:46:46
20:49:08 Offset 234MB (5.81%) Done in  0:55:46 at 21:44:54
20:50:22 Offset 318MB (7.88%) Done in  0:54:33 at 21:44:55
20:51:38 Offset 402MB (9.95%) Done in  0:53:36 at 21:45:14
20:52:51 Offset 486MB (12.03%) Done in  0:52:16 at 21:45:07
21:39:26 Offset 3338MB (82.54%) Done in  0:11:21 at 21:50:47
21:41:01 Offset 3422MB (84.61%) Done in  0:10:03 at 21:51:04
21:42:35 Offset 3506MB (86.69%) Done in  0:08:44 at 21:51:19
21:44:07 Offset 3590MB (88.76%) Done in  0:07:23 at 21:51:30
21:45:38 Offset 3674MB (90.84%) Done in  0:06:02 at 21:51:40
21:47:11 Offset 3758MB (92.91%) Done in  0:04:41 at 21:51:52
21:48:43 Offset 3841MB (94.98%) Done in  0:03:19 at 21:52:02
21:50:19 Offset 3925MB (97.06%) Done in  0:01:57 at 21:52:16
21:51:53 Offset 4009MB (99.13%) Done in  0:00:34 at 21:52:27
All data are read; waiting for threads to finish...
Time elapsed waiting for 1 thread to finish:
    1 sec (timeout in 59 min59 sec.)
All Threads Finished!
Producer time spent waiting: 3355.41 sec.
Average consumer time spent waiting: 5.27024 sec.
*********************************************
** bulk_extractor is probably CPU bound.  **
**    Run on a computer with more cores    **
**      to get better performance.         **
*********************************************
MD5 of Disk Image: f7b0960d98203168764402c05aea1e9c
Phase 2. Shutting down scanners
Phase 3. Creating Histograms
Elapsed time: 3997.53 sec.
Total MB processed: 4044
Overall performance: 1.01186 MBytes/sec (1.01186 MBytes/sec/thread)
Total email features found: 0
remnux@remnux:~$ ls
```

Forensics & File Carving -> Bulk_Extractor



Files generated by Byte-Extractor for enabled options.

For command line interface, relevant arguments has to be used to produce desired results.

Most of the options are self-explanatory.

Forensics & File Carving -> Bulk_Extractor

url_services.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: url
# Filename: /dev/sdb1
# Histogram-File-Version: 1.1
n=199   schemas.openxmlformats.org
n=107   schemas.microsoft.com
n=27    www.w3.org
n=16    purl.org
n=6     github.com
n=6     scholar.google.com
n=6     www.microsoft.com
n=6     www.verisign.com
n=4     dl.acm.org
n=4     hostname
n=4     www.example.host
n=4     www.flare-on.com
n=4     www.google.com.sg
n=3     +:80
n=3     capec.mitre.org
n=3     www.apple.com
n=2     +:443
n=2     developer.android.com
n=2     dublincore.org
n=2     evil
n=2     evil.org
n=2     example.com
n=2     google.com
n=2     grahamhosking.blogspot.ru
n=2     ocsp.verisign.com
n=2     projects.webappsec.org
n=2     schemas
```

zip.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: zip
# Filename: /dev/sdb1
# Feature-File-Version: 1.1
1622778880      [Content_Types].xml     <zipinfo><name>[Content_Types].xml</name><version>20</
version><general>6</general><compression_method>8</compression_method><uncompr_size>1495</
uncompr_size><compr_size>368</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>1560748519</
crc32><extra_field_len>520</extra_field_len><disposition bytes='1495'>decompressed</disposition></zipinfo>
1622779817      _rels/.rels     <zipinfo><name>_rels/.rels</name><version>20</version><general>6</
general><compression_method>8</compression_method><uncompr_size>590</uncompr_size><compr_size>239</
compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>3071971614</crc32><extra_field_len>520</
extra_field_len><disposition bytes='590'>decompressed</disposition></zipinfo>
1622780617      word/_rels/document.xml.rels    <zipinfo><name>word/_rels/document.xml.rels</
name><version>20</version><general>6</general><compression_method>8</compression_method><uncompr_size>1479</
uncompr_size><compr_size>303</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>3432752878</
crc32><extra_field_len>264</extra_field_len><disposition bytes='1479'>decompressed</disposition></zipinfo>
1622781242      word/document.xml       <zipinfo><name>word/document.xml</name><version>20</
version><general>6</general><compression_method>8</compression_method><uncompr_size>19289</
uncompr_size><compr_size>2686</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>1313273469</
crc32><extra_field_len>0</extra_field_len><disposition bytes='19289'>decompressed</disposition></zipinfo>
1622783975      word/theme/theme1.xml   <zipinfo><name>word/theme/theme1.xml</name><version>20</
version><general>6</general><compression_method>8</compression_method><uncompr_size>6795</
uncompr_size><compr_size>1571</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>3743765162</
crc32><extra_field_len>0</extra_field_len><disposition bytes='6795'>decompressed</disposition></zipinfo>
1625086630      word/settings.xml       <zipinfo><name>word/settings.xml</name><version>20</
version><general>6</general><compression_method>8</compression_method><uncompr_size>2638</
uncompr_size><compr_size>990</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>1576088243</
crc32><extra_field_len>0</extra_field_len><disposition bytes='2638'>decompressed</disposition></zipinfo>
1625087667      word/webSettings.xml    <zipinfo><name>word/webSettings.xml</name><version>20</
version><general>6</general><compression_method>8</compression_method><uncompr_size>511</
uncompr_size><compr_size>275</compr_size><mtime>1980-01-01T00:00:00</mtime><crc32>1152839460</
```

Forensics & File Carving -> Bulk_Extractor

domain.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: domain
# Filename: /dev/sdb1
# Feature-File-Version: 1.1
239576164       www.apple.com   .0//EN" "http://www.apple.com/DTDs/PropertyLi
240358500       www.apple.com   .0//EN" "http://www.apple.com/DTDs/PropertyLi
240370788       www.apple.com   .0//EN" "http://www.apple.com/DTDs/PropertyLi
507972537       ocsp.thawte.com \x06\x01\x05\x05\x070\x01\x86\x16http://ocsp.thawte.com0\x12\x06\x03U\x1D\x13
\x01\x01\xFF\x04\x080\x06\x01\x01
507972598       crl.thawte.com  604\xA02\xA00\x86.http://crl.thawte.com/ThawteTimestamp
507973530       ts-ocsp.ws.symantec.com \x06\x01\x05\x05\x070\x01\x86\x1Ehttp://ts-ocsp.ws.symantec.com07\x06
\x08+\x06\x01\x05\x05\x070\x02\x86+ht
507973574       ts-aia.ws.symantec.com  \x06\x01\x05\x05\x070\x02\x86+http://ts-aia.ws.symantec.com/tss-ca-
g2.cer0<
507973636       ts-crl.ws.symantec.com  301\xA0/\xA0-\x86+http://ts-crl.ws.symantec.com/tss-ca-g2.crl0(
507974209       www.verisign.com         use at https://www.verisign.com/rpa (c)101.0,\x06\x03
507974837       csc3-2010-crl.verisign.com      705\xA03\xA01\x86/http://csc3-2010-crl.verisign.com/
CSC3-2010.crl0D
507974927       www.verisign.com        \x01\x05\x05\x07\x02\x01\x16\x1Chttps://www.verisign.com/rpa0\x13\x06
\x03U\x1D%\x04\x0C0\x0A\x06
507975005       ocsp.verisign.com       \x06\x01\x05\x05\x070\x01\x86\x18http://ocsp.verisign.com0;\x06\x08+
\x06\x01\x05\x05\x070\x02\x86/ht
507975043       csc3-2010-aia.verisign.com      \x06\x01\x05\x05\x070\x02\x86/http://csc3-2010-
aia.verisign.com/CSC3-2010.cer0\x1F
507976280       crl.microsoft.com       L0J\xA0H\xA0F\x86Dhttp://crl.microsoft.com/pki/crl/product
507977265       www.verisign.com         use at https://www.verisign.com/rpa (c)101.0,\x06\x03
507977711       www.verisign.com        \x01\x05\x05\x07\x02\x01\x16\x1Chttps://www.verisign.com/cps0*\x06\x08
+\x06\x01\x05\x05\x07\x02\x02
507977755       www.verisign.com        \x05\x07\x02\x020\x1E\x1A\x1Chttps://www.verisign.com/rpa0\x0E\x06
\x03U\x1D\x0F\x01\x01\xFF\x04\x04
507977874       logo.verisign.com       \x18,{\x19.0%\x16#http://logo.verisign.com/vslogo.gif04\x06\x03U
507977928       crl.verisign.com        +0)\xA0'\xA0%\x86#http://crl.verisign.com/pca3-g5.crl04\x06\x08
```

Forensics & File Carving -> Bulk_Extractor

url_histogram.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: url
# Filename: /dev/sdb1
# Histogram-File-Version: 1.1
n=26    http://schemas.openxmlformats.org/markup-compatibility/2006
n=25    http://schemas.openxmlformats.org/officeDocument/2006/relationships
n=15    http://schemas.microsoft.com/office/word
n=15    http://schemas.microsoft.com/office/word/2010/wordml
n=15    http://schemas.microsoft.com/office/word/2012/wordml
n=15    http://schemas.openxmlformats.org/wordprocessingml/2006/main
n=14    http://www.w3.org/2001/XMLSchema-instance
n=12    http://schemas.openxmlformats.org/package/2006/relationships
n=12    http://schemas.openxmlformats.org/spreadsheetml/2006/main
n=11    http://schemas.openxmlformats.org/drawingml/2006/main
n=10    http://schemas.microsoft.com/office/spreadsheetml/2009/9/ac
n=9     http://schemas.openxmlformats.org/officeDocument/2006/relationships/worksheet
n=8     http://schemas.openxmlformats.org/drawingml/2006/picture
n=8     http://schemas.openxmlformats.org/officeDocument/2006/math
n=6     http://purl.org/dc/elements/1.1/
```

telephone_histogram.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: telephone
# Filename: /dev/sdb1
# Histogram-File-Version: 1.1
n=2     8888208908
```

rar.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: rar
# Filename: /dev/sdb1
# Feature-File-Version: 1.1
1622623232      <volume>        <rar_volume><encrypted>false</encrypted></rar_volume>
1622623252      hmk1\Key points of reading 1, 2 & 3(for reference only).docx    <rar_component><name>hmk1\Key
points of reading 1, 2 & 3(for reference only).docx</name><flags>0x0000</flags><version>29</
version><compression_method>normal</compression_method><uncompr_size>17585</uncompr_size><compr_size>14798</
compr_size><file_attr>0x20</file_attr><lastmoddate>2015-02-04T19:16:50Z</lastmoddate><host_os>Windows</
host_os><crc32>0x04CE1871</crc32></rar_component>
1622638147      hmk1\Summary of Reading 1, 2 & 3.docx   <rar_component><name>hmk1\Summary of Reading 1, 2 &
3.docx</name><flags>0x0000</flags><version>29</version><compression_method>normal</
compression_method><uncompr_size>12771</uncompr_size><compr_size>10060</compr_size><file_attr>0x20</
file_attr><lastmoddate>2015-02-04T19:25:56Z</lastmoddate><host_os>Windows</host_os><crc32>0x82DAD834</crc32></
rar_component>
```

Forensics & File Carving -> Bulk_Extractor

domain.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: domain
# Filename: /dev/sdb1
# Feature-File-Version: 1.1
239576164       www.apple.com    .0//EN" "http://www.apple.com/DTDs/PropertyLi
240358500       www.apple.com    .0//EN" "http://www.apple.com/DTDs/PropertyLi
240370788       www.apple.com    .0//EN" "http://www.apple.com/DTDs/PropertyLi
507972537       ocsp.thawte.com \x06\x01\x05\x05\x070\x01\x86\x16http://ocsp.thawte.com0\x12\x06\x03U\x1D\x13
\x01\x01\xFF\x04\x080\x06\x01\x01
507972598       crl.thawte.com  604\xA02\xA00\x86.http://crl.thawte.com/ThawteTimestamp
507973530       ts-ocsp.ws.symantec.com \x06\x01\x05\x05\x070\x01\x86\x1Ehttp://ts-ocsp.ws.symantec.com07\x06
\x08+\x06\x01\x05\x05\x070\x02\x86+ht
507973574       ts-aia.ws.symantec.com  \x06\x01\x05\x05\x070\x02\x86+http://ts-aia.ws.symantec.com/tss-ca-
g2.cer0<
507973636       ts-crl.ws.symantec.com  301\xA0/\xA0-\x86+http://ts-crl.ws.symantec.com/tss-ca-g2.crl0(
507974209       www.verisign.com         use at https://www.verisign.com/rpa (c)101.0,\x06\x03
507974837       csc3-2010-crl.verisign.com      705\xA03\xA01\x86/http://csc3-2010-crl.verisign.com/
CSC3-2010.crl0D
507974927       www.verisign.com        \x01\x05\x05\x07\x02\x01\x16\x1Chttps://www.verisign.com/rpa0\x13\x06
\x03U\x1D%\x04\x0C0\x0A\x06
507975005       ocsp.verisign.com       \x06\x01\x05\x05\x070\x01\x86\x18http://ocsp.verisign.com0;\x06\x08+
\x06\x01\x05\x05\x070\x02\x86/ht
```

unzip_carved.txt

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 1.5.5 ($Rev: 10844 $)
# Feature-Recorder: unzip_carved
# Filename: /dev/sdb1
# Feature-File-Version: 1.1
1622623252-RAR-0-ZIP-0  unzip_carved/000/1622623252-RAR-0-ZIP-0__Content_Types_.xml
<fileobject><filename>output_bulk/unzip_carved/000/1622623252-RAR-0-ZIP-0__Content_Types_.xml</
filename><filesize>1445</filesize><hashdigest type='md5'>0c68c545558c401710918137fdb65b84</hashdigest></
fileobject>
1622623252-RAR-927-ZIP-0        unzip_carved/000/1622623252-RAR-927-ZIP-0__rels_.rels
<fileobject><filename>output_bulk/unzip_carved/000/1622623252-RAR-927-ZIP-0__rels_.rels</
filename><filesize>590</filesize><hashdigest type='md5'>77bf61733a633ea617a4db76ef769a4d</hashdigest></
fileobject>
1622623252-RAR-1727-ZIP-0       unzip_carved/000/1622623252-RAR-1727-ZIP-0_word__rels_document.xml.rels
<fileobject><filename>output_bulk/unzip_carved/000/1622623252-RAR-1727-ZIP-0_word__rels_document.xml.rels</
filename><filesize>950</filesize><hashdigest type='md5'>71a16b3d2402c5001d7ea35765da86b5</hashdigest></
fileobject>
1622623252-RAR-2310-ZIP-0       unzip_carved/000/1622623252-RAR-2310-ZIP-0_word_document.xml
<fileobject><filename>output_bulk/unzip_carved/000/1622623252-RAR-2310-ZIP-0_word_document.xml</
filename><filesize>22279</filesize><hashdigest type='md5'>90e71d638b65b77e3bfd8f3bd7e17107</hashdigest></
fileobject>
```

Bulk_Extractor is very powerful tool for extracting contents from file.For more information about Bulk_Extractor, go through below urls :

http://tools.kali.org/forensics/bulk-extractor

http://www.basistech.com/wp-content/uploads/2014/04/osdf-2011-garfinkel-bulk-extractor.pdf

http://wiki.bitcurator.net/index.php?title=Using_Bulk_Extractor_Viewer_to_Find_Potentially_Sensitive_Information_on_a_Disk_Image

https://github.com/simsong/bulk_extractor/wiki/Installing-bulk_extractor

Forensics & File Carving -> Hachoir-metadata

- Hachoir metadata can extract metadata even from invalid/truncated files, remove duplicate values, Set priority to value, so it's possible to filter metadata (option --level).
- It archives (bzip2, gzip, zip, tar), audio (MPEG audio/MP3, WAV, Sun/NeXT audio, Ogg/Vorbis, MIDI, AIFF, AIFC, Real Audio), images (BMP, CUR, EMF, ICO, GIF, JPEG, PCX, PNG, TGA, TIFF, WMF, XCF), and video (ASF/WMV, AVI, Matroska, Quicktime, Ogg/Theora, Real Media).

```
remnux@remnux:~$ hachoir-metadata /home/remnux/Desktop/kroker.exe
Metadata:
- Creation date: 2009-09-12 13:02:31
- Comment: CPU: Intel 80386
- Comment: Subsystem: Windows GUI
- Format version: Portable Executable: Windows application
- MIME type: application/x-dosexec
- Endianness: Little endian
remnux@remnux:~$
```

```
remnux@remnux:~$ hachoir-metadata --help
Usage: hachoir-metadata [options] files

Options:
  -h, --help            show this help message and exit
  --type                Only display file type (description)
  --mime                Only display MIME type
  --level=LEVEL         Quantity of information to display from 1 to 9 (9 is
                        the maximum)
  --raw                 Raw output
  --bench               Run benchmark
  --force-parser=FORCE_PARSER
                        List all parsers then exit
  --parser-list         List all parsers then exit
  --profiler            Run profiler
  --version             Display version and exit
  --quality=QUALITY     Information quality (0.0=fastest, 1.0=best, and
                        default is 0.5)
  --maxlen=MAXLEN       Maximum string length in characters, 0 means unlimited
                        (default: 300)
  --verbose             Verbose mode
  --debug               Debug mode
```

Forensics & File Carving -> Hachoir-urwid

- Hachoir urwid project is a binary file explorer that uses the Hachoir library to parse files.
- Using this tool, we can know the exact meaning of each bit/byte of files.

```
remnux@remnux:~$ hachoir-urwid --help
Usage: hachoir-urwid [options] filename

Options:
  -h, --help            show this help message and exit

  Urwid:
    Option of urwid explorer

    --preload=PRELOAD   Number of fields to preload at each read
    --path=PATH         Initial path to focus on
    --parser=PARSER     Use the specified parser (use its identifier)
    --offset=OFFSET     Skip first bytes of input file
    --parser-list       List all parsers then exit
    --profiler          Run profiler
    --profile-display   Force update of the screen beetween each event
    --size=SIZE         Maximum size of bytes of input file
    --hide-value        Don't display value
    --hide-size         Don't display size
    --version           Display version and exit

  Hachoir library:
    Configure Hachoir library

    --verbose           Verbose mode
    --log=LOG           Write log in a file
    --quiet             Quiet mode (don't display warning)
    --debug             Debug mode
remnux@remnux:~$
```

```
0) file:/home/remnux/Desktop/kroker.exe: Microsoft Windows Portable Executable: Intel 80386, Windows GUI (168.0 KB)
 + 0) msdos: MS-DOS program header (64 bytes)
 + 248) pe_header (24 bytes)
 + 272) pe_opt_header: PE optional header: Windows GUI, entry point 0x000073a0 (224 bytes)
 + 496) section_hdr[0]: Section ".text": rva=0x00001000..0x0000bb6c, size=42.9 KB, exec, read (40 bytes)
 + 536) section_hdr[1]: Section ".data": rva=0x0000c000..0x0000ce74, size=3700 bytes, read (40 bytes)
 + 576) section_hdr[2]: Section "BSS": rva=0x0000d000..0x0004608d, size=228.1 KB, read (40 bytes)
 + 616) section_hdr[3]: Section "DATA": rva=0x00047000..0x00047384, size=900 bytes, read (40 bytes)
 + 656) section_hdr[4]: Section ".rdata": rva=0x00048000..0x000486ee, size=1774 bytes, read, write (40 bytes)
   1536) section_text= "\0\0\0\0\0\0\0\0\0\0\0\0\0\0(...)": Raw data (43.0 KB)
   45568) section_data= "3\xc6\0\0D\xc6\0\0T\xc6\0\0h\xc6(...)": Raw data (4096 bytes)
   49664) section_BSS= "\0\0\0\0\0\0\0\0\0\0\0\0\0\0(...)": Raw data (116.5 KB)
   168960) section_DATA= "\0\0\0\0\0\0\0\0\0\0\0\0\0\0(...)": Raw data (1024 bytes)
   169984) section_rdata= "ih\0YwIfN\x000kd\0\0(...)": Raw data (2048 bytes)
```

References:

- Remnux(https://remnux.org/)
- Artifacts and Malwares(http://windowsir.blogspot.com/p/malware.html)
- Unxor(https://github.com/tomchop/unxor/)
- Bulk_Extractor(http://tools.kali.org/forensics/bulk-extractor)
- Bulk_Extractor(http://www.basistech.com/wp-content/uploads/2014/04/osdf-2011-garfinkel-bulk-extractor.pdf)
- Bulk_Extractor(http://wiki.bitcurator.net/index.php?title=Using_Bulk_Extractor_Viewer_to_Find_Potentially_Sensitive_Information_on_a_Disk_Image)
- Bulk_Extractor(https://github.com/simsong/bulk_extractor/wiki/Installing-bulk_extractor)
- Scalpel(https://www.youtube.com/watch?v=5Z9JsBazOdw)
- Foremost(https://www.youtube.com/watch?v=OGlRKz2PECg)
- Dcfldd(http://www.forensicswiki.org/wiki/Dcfldd)
- Xortool(https://github.com/hellman/xortool)
- Hachoir(http://www.forensicswiki.org/wiki/Hachoir)
- XorBruteForcer(http://digital-forensics.sans.org/blog/2013/05/14/tools-for-examining-xor-obfuscation-for-malware-analysis)

Thank you..!