# Windows Forensics Project : Analyzer

Made by : daniel kovalevsky
Class : 7736/21
Teacher : arel regev

## *Intro :*

The script performing memory analysis , with carvers such  as binwalk foremost and bulk_extractor while also checking if the User have the carvers installed , if not it will install then collecting the data extracted and saves it into a folder of the report ,
While also if the file is .mem it will perform malware analysis With volatility and extract from the memory process that were running , the connections there were and etcetera
The script also perform user check to see if the user is root
At the end the script saves the extracted data to a zip.

---

The Beginning of the script with the some of the variable's that used, also info incase of installation's problems that might occur

```
#if u have problems with the installations
#please make sure u've updated the apt-get

#remove the # from the following command if u want to run it from the script.
#apt-get update -y

#variable to show the run time
start=$(date +%s.%N)

#variable to indcate the user
whoami=$(whoami)

#variable to show location
pwd=$(pwd)

#variable to show the date
date=$(date)
```

```
#variable for colors
G="\e[32m"
E="\e[0m"
R="\e[31m"
O="\e[33m"
LC="\e[1;36m"
LR="\e[1;31m"
LG="\e[1;32m"
LP="\e[1;35m"

printf ${G}
figlet Memory Analyzer
printf ${E}
```

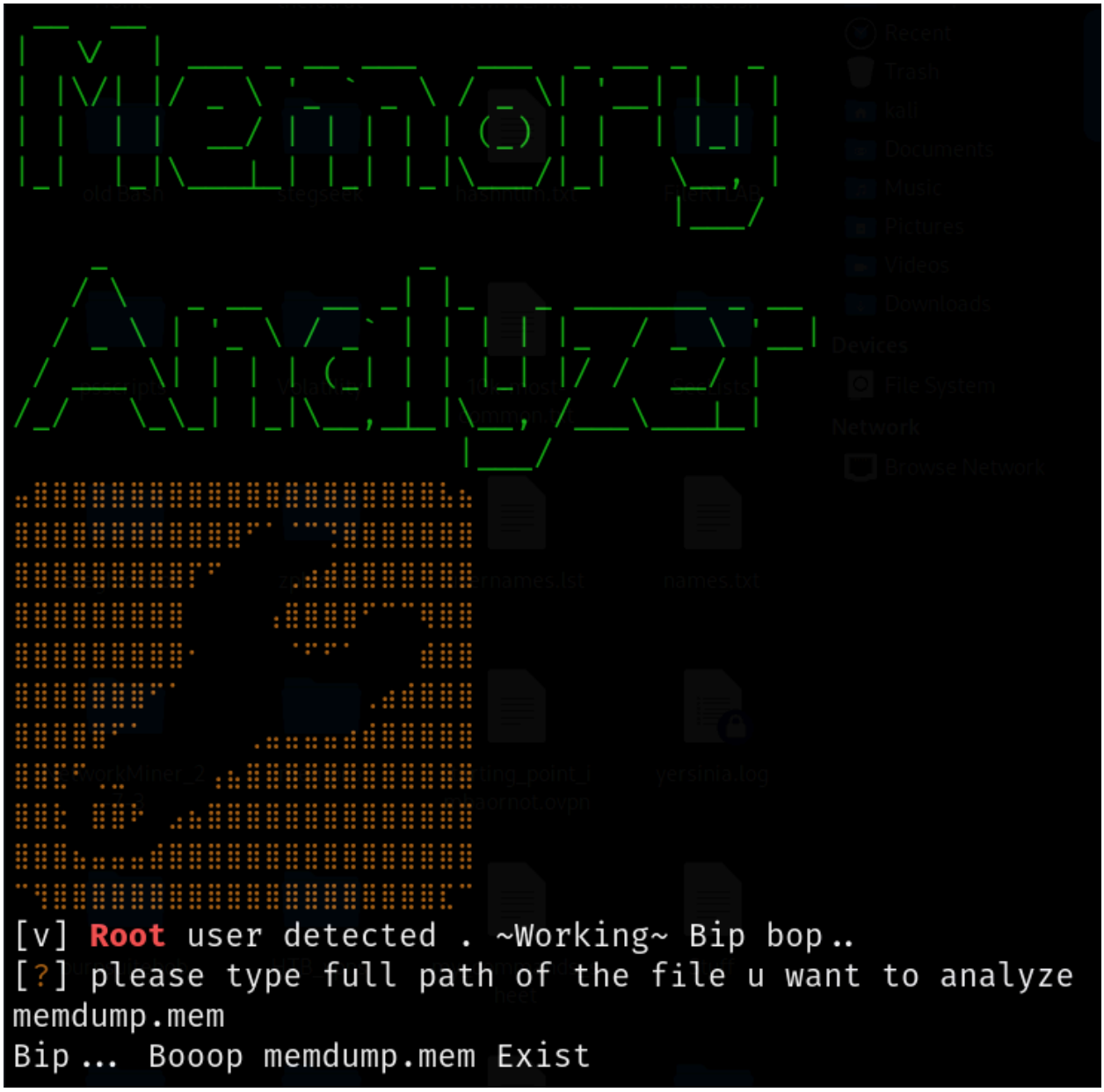
## **Function :** Usercheck

```
    #making the user to activate this script only with root previliges.
function usercheck()
{
if [ $whoami = root ]
then
printf ${O}
echo    "▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▄▄▄▄▄▀▀ ▀▀▀▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▄▄▀▀    ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▄▄     ▄▄▄▄▄▀▀▀▀▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▄▄▄     ▀▀▀▀    ▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▀▀           ▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▀▀      ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▀▀ ▄▄    ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄  ▄▄▀  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
echo    "▀▀▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▀▀"
printf ${E}

    #informing if the user is root
    echo -e "[v] ${LR}Root${E} user detected . ~Working~ Bip bop.."

    #creates directory
    mkdir /$pwd/Analyze_Results &> /dev/null

else
    #informs and exiting if user not root
    echo -e "[${LR}!${E}] you are not *${LR}ROOT${E}* Exiting. please run as root"
    exit 1
fi
}
#calling a function
usercheck
```

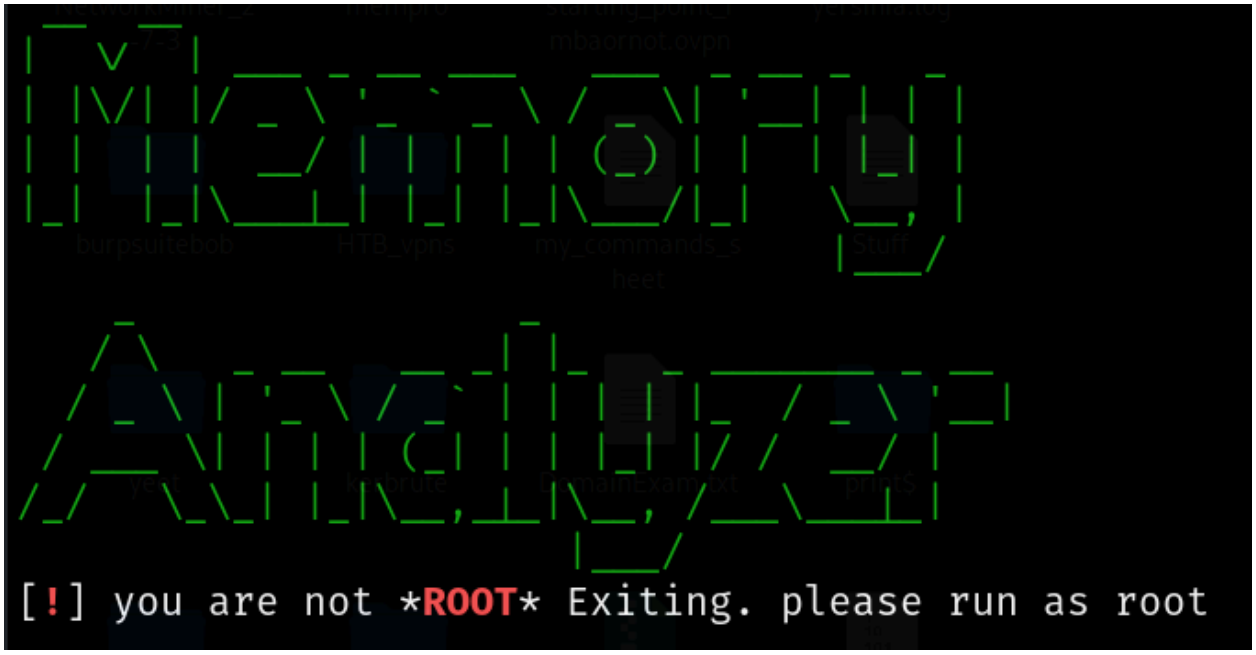
This function checks if the user is root , if the user is root the script will start with asking the user which file to analyze

```
    #asks the user a question than using his answer as a var
echo -e "[${O}?${E}] please type full path of the file u want to analyze"
    read FILE
```

Which will store the file into variable

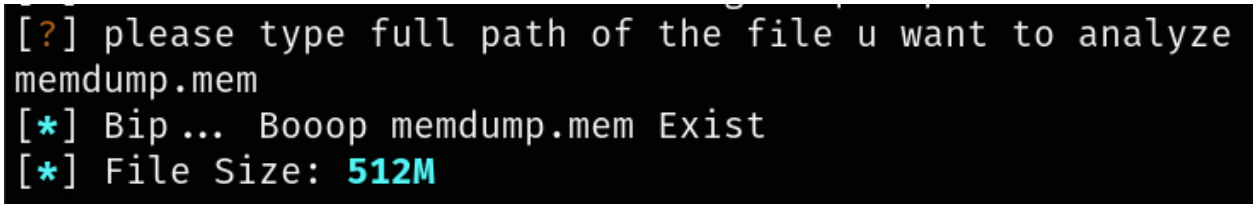



If not the script will stop working



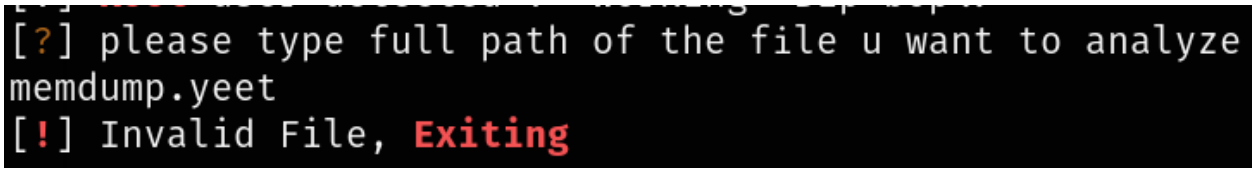

After activating the script with root
The script will call
Function : Verify
To check if the the file is .mem or .dd , and to tell the file size

```
#variable to user the file basename
NAME=$(basename $FILE)

#function that verify's that its a file.
function Verify()
{

    #checks if the file given is file
    if [ -f $FILE ] && [ $NAME == *.mem ] || [ $NAME == *.dd ]
    then
        echo -e "[${LC}*${E}] Bip... Booop $NAME Exist"
        echo -e "[${LC}*${E}] File Size: ${LC}$(ls -h -l $FILE | awk '{print $5}')${E}"
        echo ""
    else
        echo -e "[${LR}!${E}] Invalid File, ${LR}Exiting${E}"
        exit
fi
}
#calling a function
Verify
```




If the file is .mem or .dd the function will print $file exists. And will tell the size

If not the user will get




After that the script will call for the installation functions.
Such as stringsinstall , Binwalkin, foremostinstall , bulk_extractorinstall which will check if the user has the carvers installed. If not, the functions will run the installations .

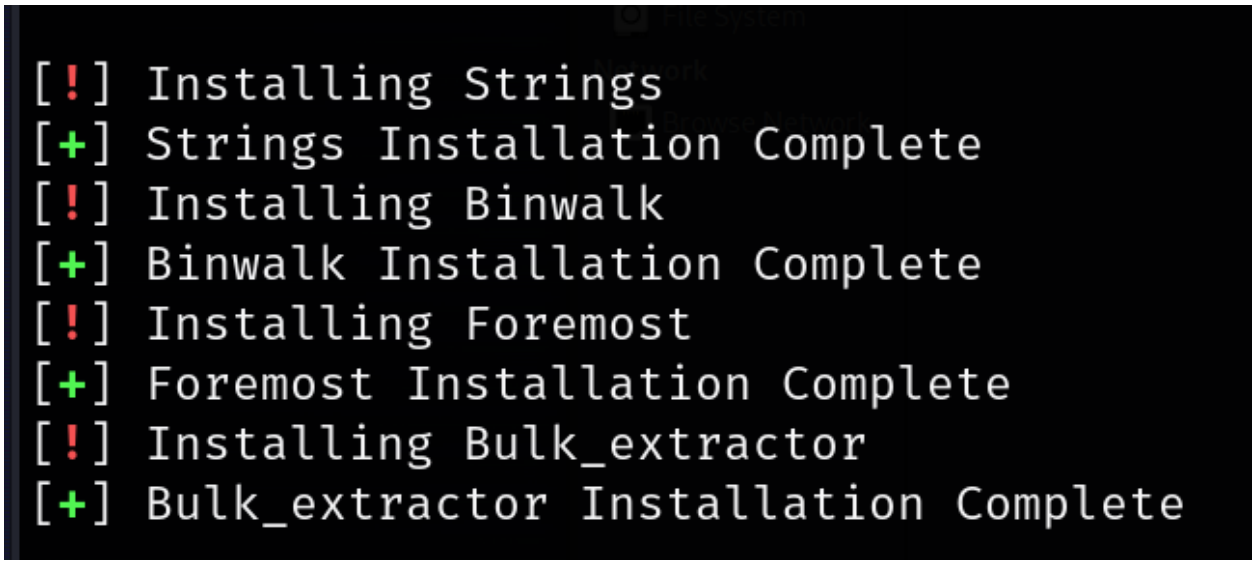
```
    #installing calling a function with option's to find if a program is installed or not , if not that it will install
function stringsinstall()
{
    if [ -e "/usr/bin/strings" ]
        then
            echo -e "[${LG}+${E}] strings is installed"
    else
        echo -e "[${LR}!${E}] Installing Strings"
            apt-get install binutils -y &> /dev/null
            sleep 1
        echo -e "[${LG}+${E}] Strings Installation Complete"
fi
}
#calling a function
stringsinstall
```

```
    #installing calling a function with option's to find if a program is installed or not , if not that it will install
function Binwalkin()
{
    if [ -e "/usr/bin/binwalk" ]
        then
            echo -e "[${LG}+${E}] Binwalk is installed"
    else
        echo -e "[${LR}!${E}] Installing Binwalk"
            apt-get install binwalk -y &> /dev/null
            sleep 1
        echo -e "[${LG}+${E}] Binwalk Installation Complete"
fi
}
#calling a function
Binwalkin
```

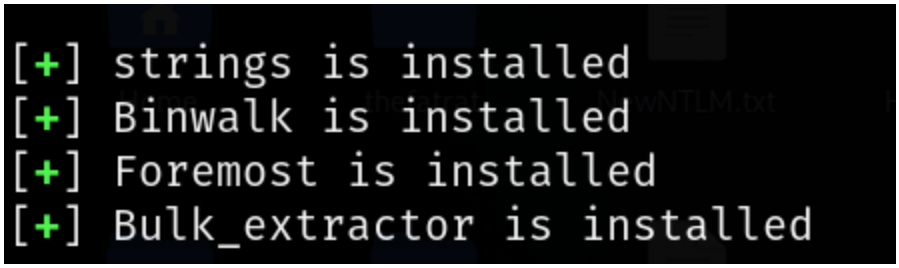
```
    #installing calling a function with option's to find if a program is installed or not , if not that it will install
function Foremostinstall()
{
    if [ -e "/usr/bin/foremost" ]
        then
            echo -e "[${LG}+${E}] Foremost is installed"
    else
        echo -e "[${LR}!${E}] Installing Foremost"
            apt-get install foremost -y &> /dev/null
            sleep 1
        echo -e "[${LG}+${E}] Foremost Installation Complete"
fi
}
#calling a function
Foremostinstall
```

```
    #installing calling a function with option's to find if a program is installed or not , if not that it will install
function bulk_extractorInstall()
{
    if [ -e "/usr/bin/bulk_extractor" ]
        then
            echo -e "[${LG}+${E}] Bulk_extractor is installed"
    else
        echo -e "[${LR}!${E}] Installing Bulk_extractor"
            apt-get install bulk-extractor -y &> /dev/null
            sleep 1
        echo -e "[${LG}+${E}] Bulk_extractor Installation Complete"
fi
}
#calling a function
bulk_extractorInstall

printf ${G}
figlet Installations Complete
printf ${E}
```

If the carvers are not install the user will get this output




If the carvers are installed the user will get this output




Function Menu :

This function will display the user menu with which of the following carving tools to be selected.

1 for Binwalk , 2 for Foremost , 3 for Bulk_extractor , 4 for All

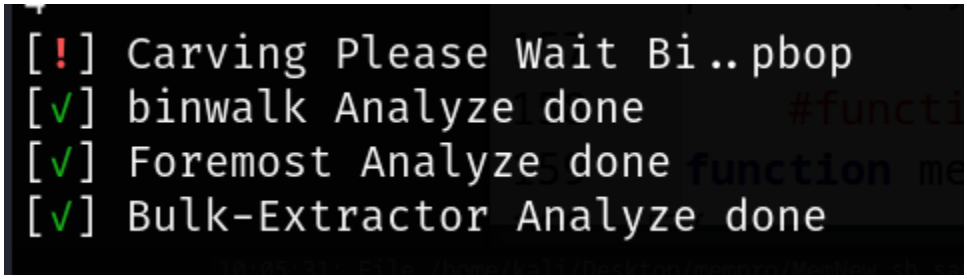
```
    #function that contains menu for carving options.
function menu()
{
    echo""
    echo -e "Choose an option to use:\n [${LG}+${E}] 1 - Binwalk\n [${LG}+${E}] 2 - Foresmost\n [${LG}+${E}] 3 - Bulk_Extractor\n [${LG}+${E}] 4 - A
        read answer
        #the menu options
case $answer in

    1)
        echo -e "[${LR}!${E}] Carving Please Wait Bi..pbop"
        #runs binwalk
        binwalk --run-as=root $FILE -e --directory $pwd/Analyze_Results &> /dev/null
        #stores offsets
        binwalk $FILE >> /$pwd/Analyze_Results/binwalkoffsets.txt
        echo -e "[${G}✓${E}] binwalk Analyze done"
    ;;

    2)
        echo -e "[${LR}!${E}] Carving Please Wait Bi..pbop"
        #runs foremost
        foremost $FILE -o $pwd/Analyze_Results/ForeOutput &> /dev/null
        echo -e "[${G}✓${E}] Foremost Analyze done"
    ;;
```

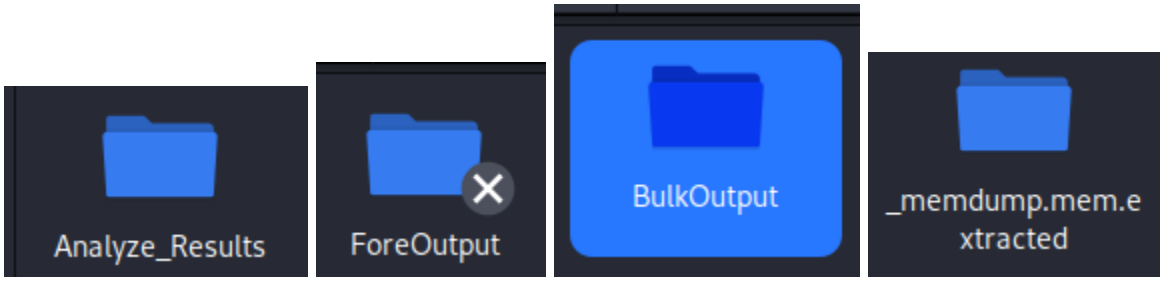
```
    3)
        echo -e "[${LR}!${E}] Carving Please Wait Bi..pbop"
        #runs bulk_extractor
        bulk_extractor $FILE -o $pwd/Analyze_Results/BulkOutput &> /dev/null
        echo -e "[${G}✓${E}] Bulk-Extractor Analyze done"
    ;;

    4)
        echo -e "[${LR}!${E}] Carving Please Wait Bi..pbop"
            #runs binwalk
        binwalk --run-as=root $FILE -e --directory $pwd/Analyze_Results &> /dev/null
            #stores offset
            binwalk $FILE >> /$pwd/Analyze_Results/binwalkoffsets.txt

            echo -e "[${G}✓${E}] binwalk Analyze done"
            #runs foremost
        foremost $FILE -o $pwd/Analyze_Results/ForeOutput &> /dev/null

            echo -e "[${G}✓${E}] Foremost Analyze done"
            #runs bulk_extractor
        bulk_extractor $FILE -o $pwd/Analyze_Results/BulkOutput &> /dev/null
            echo -e "[${G}✓${E}] Bulk-Extractor Analyze done"
    ;;

    *)
        echo -e "[${LR}!${E}] Wrong input please pick again"
            #calling menu function
            menu
    ;;
```

```
esac
}
#calling a function
menu
```

```
Choose an option to use:
 [+] 1 - Binwalk
 [+] 2 - Foresmost
 [+] 3 - Bulk_Extractor
 [+] 4 - All [!] Warning might take time
```

After selecting the the tool , the carver will extract data and store it in the folder that was created

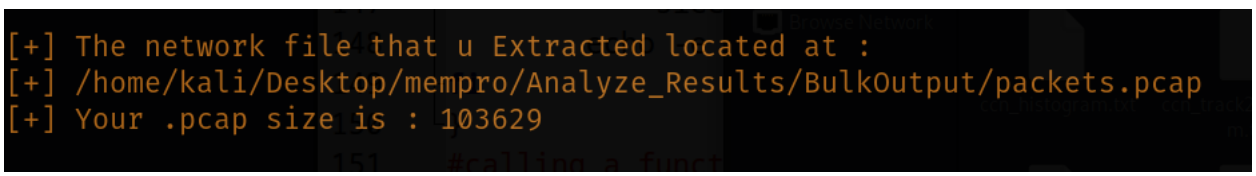



After that the script will run pcap check to see if pcap file was extracted

```
#if pcap file was extracted than show user location,size
if [[ -f "$pwd/Analyze_Results/BulkOutput/packets.pcap" ]]
    then
        printf ${O}
        echo  "[+] The network file that u Extracted located at :"
        echo  "[+] $(ls $pwd/Analyze_Results/BulkOutput/packets.pcap) "
        echo  "[+] Your .pcap size is : $(ls -l $pwd/Analyze_Results/BulkOutput/packets.pcap | awk '{print $5}') "
        printf ${E}
    else
        echo -e "[${LR}!${E}] Pcap file was not extracted"
fi
```

If the file was extracted the user will receive the following output

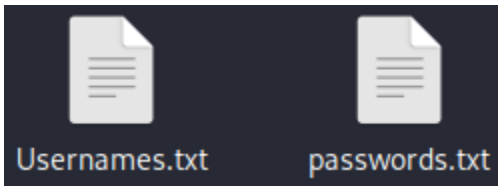


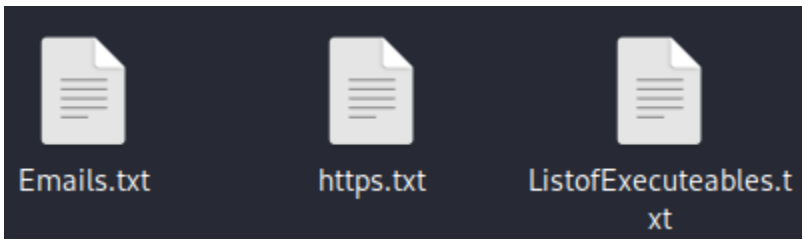
If not, the user will receive the following output.


After that the following function will be called

```
#function that strings the file to extract specific keywords
function string()
{
#strings to get more info like usernames/email/passwords/exe/http
strings $FILE | grep "USERNAME" | grep = |sort | uniq | sort -n  >> /$pwd/Analyze_Results/Usernames.txt
strings $FILE | grep -i "password" | sort | uniq | sort -n  >> /$pwd/Analyze_Results/passwords.txt
strings $FILE | grep "@" | grep "\.com"  >> /$pwd/Analyze_Results/Emails.txt
strings $FILE | grep -i "http" | sort | uniq | sort -n  >> /$pwd/Analyze_Results/https.txt
strings $FILE | grep -F ".exe" | sort | uniq | sort -n  >> /$pwd/Analyze_Results/ListofExecuteables.txt

echo -e "[${LP}^${E}] Used Strings To Extract Keywords."
}
string
```

Which will use strings to extract keywords from the memory file


At this point if the file is .dd then function report and fin will be called

```
#if the file is dd than it will show the results and exit the script.
if [[ $FILE == *.dd ]]
then
    #calling a function
    Report
    #calling a function
    fin
fi
```

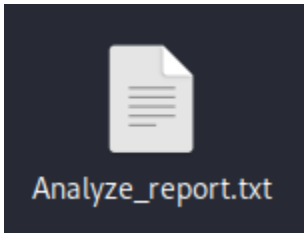
```
function Report()
{

printf ${O}

#prints when the file was analyized
echo "[?] Analyzed at : $date $FILE" | tee -a $pwd/Analyze_Results/Analyze_report.txt

#prints how long the script took
echo "[*] This script took $SECONDS seconds to execute" | tee -a $pwd/Analyze_Results/Analyze_report.txt

#prints the amount of files that were extracted
echo "[*] Files Extracted From The Memory Analysis : $(ls -R /$pwd/Analyze_Results | wc -l) " | tee -a $pwd/Analyze_Results/Analyze_report.txt

printf ${E}

    #if folder exists use variable , if not use other variable
    if  [ -d "$pwd/Analyze_Results/_$NAME.extracted" ]  ;then
        BIN=$(dir -l /$pwd/Analyze_Results/_$NAME.extracted | wc -l)
        else
        BIN=$(echo 0)
    fi

    #if folder exists use variable , if not use other variable
    if  [ -d "$pwd/Analyze_Results/ForeOutput" ]    ;then
        FORE=$(dir -l /$pwd/Analyze_Results/ForeOutput | wc -l)
        else
        FORE=$(echo 0)
    fi

    #if folder exists use variable , if not use other variable
    if  [ -d "$pwd/Analyze_Results/BulkOutput" ]    ;then
        BULK=$(dir -l /$pwd/Analyze_Results/BulkOutput | wc -l)
        else
        BULK=$(echo 0)
    fi

    #if folder exists use variable , if not use other variable
    if  [ -d "$pwd/Analyze_Results/$NAME" ] ;then
        VOL=$(dir -l /$pwd/Analyze_Results/$NAME | wc -l)
        else
        VOL=$(echo 0)
    fi


    #echos the results of the extracter files from the carver
    echo -e "[Bulk extracted : ${LC}$BULK${E} Files ] [Foremost Extracted : ${LC}$FORE${E} Files ] [Binwalk extracted : ${LC}$BIN${E} Files ] [Volat
}
```

# Function Report :

# This function will collect data  on how much the script ran , How much files were extracted , and at which date the script ran

```
[?] Analyzed at : Sat Mar 30 01:31:15 PM EDT 2024 memdump.mem
[*] This script took 30 seconds to execute
[*] Files Extracted From The Memory Analysis : 222
[Bulk extracted : 0 Files ] [Foremost Extracted : 9 Files ] [Binwalk extracted : 0 Files ] [Volatility extra
cted : 7 Files]
```

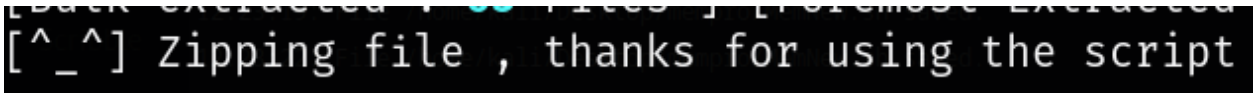
If the folder of the carver was created than it will count how much files were extracted from it , if not then it will use the variable 0

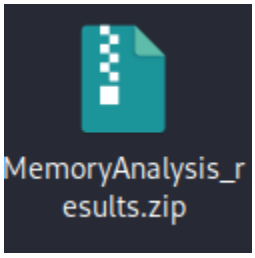


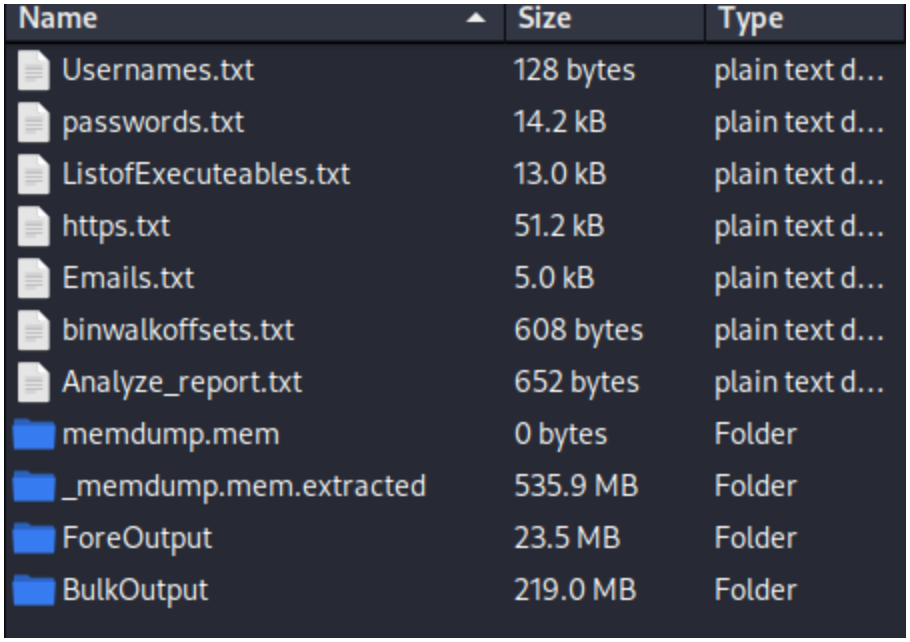

Function Fin :

```
#function that end's the script and zip's the results
function fin()
{
     echo "[^_^] Zipping file , thanks for using the script"
    zip -r MemoryAnalysis_results /$pwd/Analyze_Results &> /dev/null
        sleep 2
            #can activate if u also want to remove the folder just remove the # in the following command
    #rm -r /$pwd/Analyze_Results &> /dev/null
updatedb
    #exits the script
    exit 1
}
```

This function will zip all the data that were collected from the script





| Name | Size | Type |
|---|---|---|
| Usernames.txt | 128 bytes | plain text d... |
| passwords.txt | 14.2 kB | plain text d... |
| ListofExecuteables.txt | 13.0 kB | plain text d... |
| https.txt | 51.2 kB | plain text d... |
| Emails.txt | 5.0 kB | plain text d... |
| binwalkoffsets.txt | 608 bytes | plain text d... |
| Analyze_report.txt | 652 bytes | plain text d... |
| memdump.mem | 0 bytes | Folder |
| _memdump.mem.extracted | 535.9 MB | Folder |
| ForeOutput | 23.5 MB | Folder |
| BulkOutput | 219.0 MB | Folder |

If the file is .mem than it will make a dir with the basename of the file

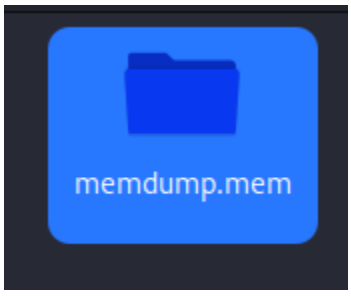
```
#check's if a file is a mem file if not exists
if [[ $FILE == *.mem ]]
    then
    echo -e "[${LG}+${E}] Analyzing : $NAME"
        mkdir -p $pwd/Analyze_Results/$NAME > /dev/null 2>&1
    #calling Volatility fuction
        Volatility
    #calling a function
        Report
else
    echo -e "[${LR}!${E}]Cant Use Vol Analysis File Not *.mem* File,"
    sleep 1
    echo "Exiting"
    exit
fi
#calling Fin Fuction
fin
```

And will call the function Volatility.

Function Volatility:
The function will get the profile from running volatility store it into variable and will loop the volatility with plugins from the PLUGINS variable array . will tell the user the profile name then will store the data to txt file

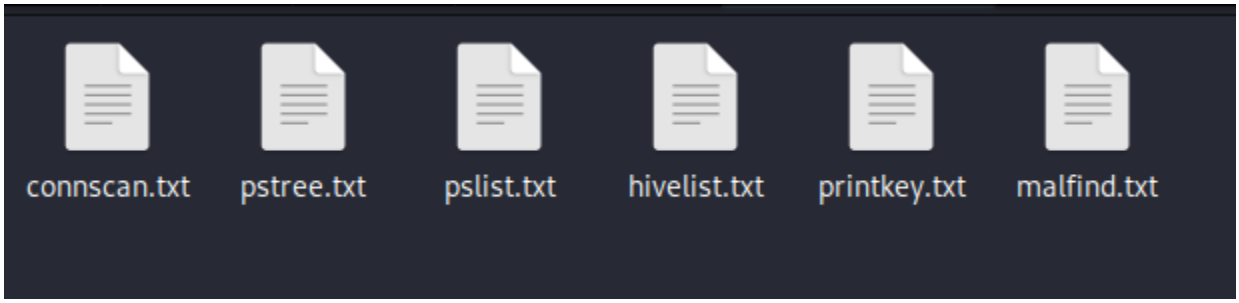
```
#function that using the Volatility carver
function Volatility()
{

#var that gets the profile of the memory file
PROFILE=$(./Vol -f $FILE imageinfo  2> /dev/null | grep "Suggested" | awk '{print $4}' | awk -F "," '{print $1}' )

#array variable for the loop
PLUGINS="pstree connscan pslist hivelist printkey malfind"

#prints the profile
echo -e "[${LC}*${E}] $NAME file Profile = ${LC}$PROFILE${E}"

#creates a loop that runs all the plugins we used ^ and saves it to txt files with the name of the plugin.
for X in $PLUGINS
    do
        echo -e "[${LG}+${E}] Plugin being used: [-] $X"
    ./Vol -f $FILE --profile=$PROFILE $X > $pwd/Analyze_Results/$NAME/$res_$X.txt 2> /dev/null

#stops the loop
done
echo""
}
```

```
[+] Analyzing : memdump.mem
[*] memdump.mem file Profile = WinXPSP2x86
[+] Plugin being used: [-] pstree
[+] Plugin being used: [-] connscan
[+] Plugin being used: [-] pslist
[+] Plugin being used: [-] hivelist
[+] Plugin being used: [-] printkey
[+] Plugin being used: [-] malfind
```

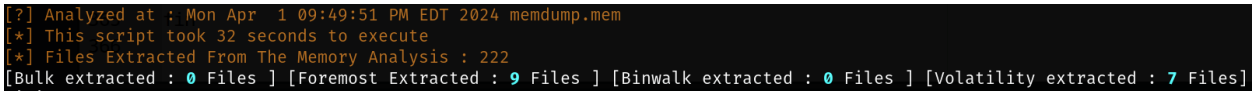


the folder that’s created after the volatility

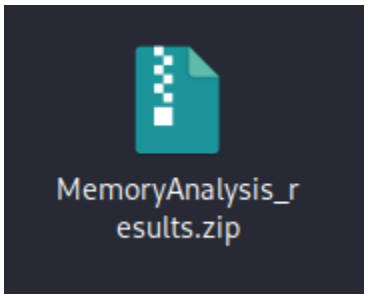
The files that’s created after the volatility




Then the script will call function report as above




After that the script will can the fin function that will zip the files and will store them into rar




```
#function that end's the script and zip's the results
function fin()
{
     echo "[^_^] Zipping file , thanks for using the script"
    zip -r MemoryAnalysis_results /$pwd/Analyze_Results &> /dev/null
        sleep 2
            #can activate if u also want to remove the folder just remove the # in the following command
    #rm -r /$pwd/Analyze_Results &> /dev/null
updatedb
    #exits the script
    exit 1
}
```

The user also can remove the folder if needed by removing the # From the

```
#rm -r /$pwd/Analyze_Results &> /dev/null
```

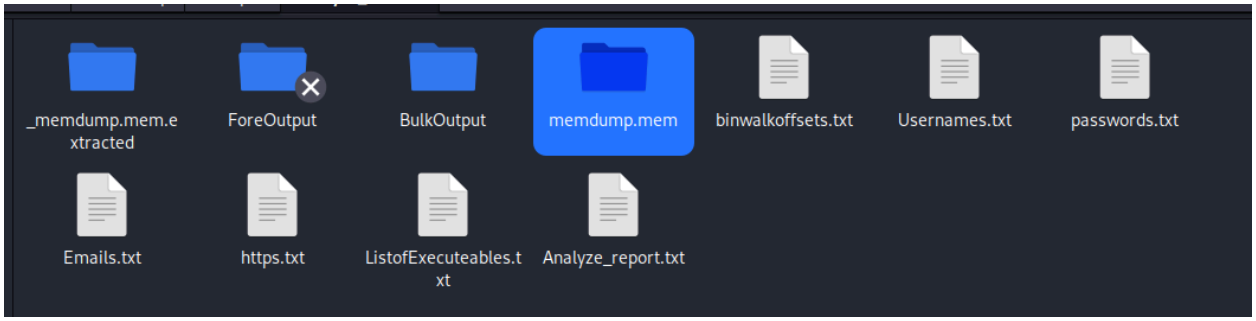
---

Full run in one go :

```
┌──(root㉿kali)-[/home/kali/Desktop/mempro]
└─# ./Analyzer.sh
```

```
[v] Root user detected . ~Working~ Bip bop..
[?] please type full path of the file u want to analyze
memdump.mem
[*] Bip... Booop memdump.mem Exist
[*] File Size: 512M

[+] strings is installed
[+] Binwalk is installed
[+] Foremost is installed
[+] Bulk_extractor is installed
```

```
Choose an option to use:
 [+] 1 - Binwalk
 [+] 2 - Foresmost
 [+] 3 - Bulk_Extractor
 [+] 4 - All [!] Warning might take time

4
[!] Carving Please Wait Bi..pbop
[√] binwalk Analyze done
[√] Foremost Analyze done
[√] Bulk-Extractor Analyze done

[+] The network file that u Extracted located at :
[+] /home/kali/Desktop/mempro/Analyze_Results/BulkOutput/packets.pcap
[+] Your .pcap size is : 103629

[^] Used Strings To Extract Keywords.

[+] Analyzing : memdump.mem
[*] memdump.mem file Profile = WinXPSP2x86
[+] Plugin being used: [-] pstree
[+] Plugin being used: [-] connscan
[+] Plugin being used: [-] pslist
[+] Plugin being used: [-] hivelist
[+] Plugin being used: [-] printkey
[+] Plugin being used: [-] malfind

[?] Analyzed at : Mon Apr  1 10:09:00 PM EDT 2024 memdump.mem
[*] This script took 89 seconds to execute
[*] Files Extracted From The Memory Analysis : 3446
[Bulk extracted : 65 Files ] [Foremost Extracted : 9 Files ] [Binwalk extracted : 2 Files ] [Volatility extracted : 7 Files]
[^_^] Zipping file , thanks for using the script

┌──(root㉿kali)-[/home/kali/Desktop/mempro]
└─# 
```



```
  327 Apr  1 22:10 Analyze_report.txt
  608 Apr  1 22:09 binwalkoffsets.txt
 4096 Apr  1 22:10 BulkOutput
 2523 Apr  1 22:10 Emails.txt
 4096 Apr  1 22:09 ForeOutput
25606 Apr  1 22:10 https.txt
 6482 Apr  1 22:10 ListofExecuteables.txt
 4096 Apr  1 22:10 memdump.mem
 4096 Apr  1 22:09 _memdump.mem.extracted
 7115 Apr  1 22:10 passwords.txt
   64 Apr  1 22:10 Usernames.txt
```

🙂