Forensics Project

Ziv Kaufman

Script Breakdown

1. Shebang and Function Definition

- **Shebang**: The script starts with `#!/bin/bash`, indicating it should be executed using the Bash shell.

2. START Function

Purpose: Checks if the script is run as root and verifies the existence of a file specified by the user.

- **Root Check**: Verifies if the user has root privileges. If not, the script exits.
- **File Check**: Prompts the user for a file name and checks its existence, notifying the user if it's missing.
- **User Experience**: Introduces a 3-second delay between operations for readability.

```
function START ()

{

# Check the current user; exit if not 'root'

    if [ "$(whoami)" = "root" ]; then
        echo "You are root"
    else
        echo "[*] You are not root!!! exiting ..."
        exit
    fi

    sleep 3

# Allow the user to specify the filename; check if the file exists.
    read -p "[*] Enter the file you want to analyze: " FILE # this condition checks if the file exists
    if [ -f "$FILE" ]
    then
    echo "[*] File exists"
else
    echo "[*] File does not exist"

    fi

    sleep 3

}
```

3. TOOLS Function

Purpose: Ensures the presence of essential forensic tools, installing them if missing.

- **Tool Checks**: Uses `dpkg -l` to verify installation of `bulk_extractor`, `binwalk`, and `foremost`.
- **Installation**: Installs missing tools using `apt-get`.

```
function TOOLS ()

{

# Create a function to install the forensics tools if missing.

# Checking to see if bulk_extractor is installed

    BULK_EXTRACTOR=$(dpkg -l | grep 'bulk-extractor' | awk '{print $2}')

    # Check if the BULK_EXTRACTOR variable is empty
    if [ -z "$BULK_EXTRACTOR" ]; then
        echo "installing bulk_extractor"

        sudo apt-get install -y bulk-extractor
    else
        echo "[*] Bulk_extractor is already installed"

    fi

    sleep 3
```

```
  # Checking to see if binwalk is installed

  BINWALK=$(dpkg -l | grep "binwalk" | head -n1 | awk '{print $2}')

  # Check if the BINWALK variable is empty
  if [ -z "$BINWALK" ]; then
      echo "installing binwalk"

      sudo apt-get install -y binwalk
  else
      echo "[*] Binwalk is already installed"

  fi

  sleep 3

   # Checking to see if foremost is installed
```

```
     # Checking to see if foremost is installed

    FOREMOST=$(dpkg -l | grep "foremost" | head -n1 | awk '{print $2}')

    # Check if the FOREMOST variable is empty
    if [ -z "$FOREMOST" ]
    then
        echo "installing foremost"

        sudo apt-get install -y foremost
    else
        echo "[*] Foremost is already installed"

    fi

    sleep 3
}
```

4. CARVERS Function

Purpose: Utilizes forensic carvers to extract data from the specified file.

- **Binwalk**: Extracts files into the `BINWALK` directory.
- **Foremost**: Extracts files into the `FOREMOST` directory.

```
function CARVERS ()

# Use different carvers to automatically extract data

# Data should be saved into a directory.

{

    rm -rf BINWALK
    echo "[*] Files extracted with binwalk being saved to BINWALK directory"
    binwalk -e --run-as=root -C BINWALK $FILE # --run-as=root is being added to this command because some memory files require this comamand when using Binwalk


    sleep 3


    rm -rf FOREMOST
    echo "[*] Files extracted with foremost being saved to FOREMOST directory"
    foremost -o FOREMOST $FILE


    sleep 3

}
```

5. ANALYZE Function

Purpose: Analyzes the file for network traffic and other data.

- **Bulk Extractor**: Extracts data into the `BULK` directory.
- **Network File Check**: Searches for `packets.pcap` in the `BULK` directory, copying it to `/tmp/`. Reports its existence and size if found.

```
function ANALYZE ()

{



# Attempt to extract network traffic; if found, display to the user the location and size

rm -rf BULK
echo "[*] Files extracted with bulk_extractor being saved to BULK directory"
    sleep 3
bulk_extractor -o BULK $FILE
cp BULK/packets.pcap /tmp/

    sleep 3

if [ -f /tmp/packets.pcap ] # check if packets.pcap exists
then
    echo "[*] Found Network File > Saved into /tmp/packets1.pcap1 [Size:$(ls -lh /tmp/ | grep packets.pcap | awk '{print $5}')]"
    else
    echo "[*] No Network files found"
fi

# binwalk and foremost do not typically extract network files

    sleep 3

}
```

6. STRINGS Function

Purpose: Extracts human-readable strings and checks for specific evidence.

- **String Extraction**: Uses `strings` to identify and categorize text within the file.
- **Evidence Check**: Searches for mentions of "Syria" and "Iran" to flag potential relevance.

```
function STRINGS () {

    # Check for human-readable (exe files, passwords, usernames, etc.).
    rm Strings_*

    echo "[*] human readable files being saved to current directory"

    strings $FILE | grep -i 'exe' > Strings_exe
    strings $FILE | grep -i 'password' > Strings_password
    strings $FILE | grep -i 'user' > Strings_username
    strings $FILE | grep -i 'http' > Strings_http
    strings $FILE | grep -i '@' > Strings_email
```

```
# Checking for any evidence file went through Syria at any point
echo "[*] Checking for any evidence file went through Syria at any point"
sleep 3
SYRIA=$(strings $FILE | grep -i "Syria")
if [ -z "$SYRIA" ]; then
   echo "[*] No evidence found that file has been in Syria at any point"
else
   echo "[*] Please note that there is suspicion file may have been in Syria at some point that must be further investigated."
   strings $FILE | grep -i 'syria'
fi


sleep 3
```

```
  sleep 3

  # Checking for any evidence file went through Iran at any point
  echo "[*] Checking for any evidence file went through Iran at any point"
  sleep 3
  IRAN=$(strings $FILE | grep -i "Iran")
  if [ -z "$IRAN" ]; then
     echo "[*] No evidence found that file has been in Iran at any point"
  else
     echo "[*] Please note that there is suspicion file may have been in Iran at some point that must be further investigated."
     strings $FILE | grep -i  'iran'
  fi

}

sleep 3
```

7. VOLATILITY Function

Purpose: Analyzes memory files using Volatility.

- **Memory File Check**: Uses `imageinfo` to verify if the file is a memory image.
- **Profile and Analysis**: Suggests profiles and conducts analysis of processes, network connections, registry hives, browser history, and user activity.

```
function VOLATILITY ()

{

# Check if the file can be analyzed in Volatility; if yes, run Volatility


# Check if the file can be analyzed in Volatility; if yes, run Volatility

# Capture the output of the imageinfo command

    OUTPUT=$(./vol -f "$FILE" imageinfo 2>/dev/null)

# Check if "No suggestion" is present in the output
    if echo "$OUTPUT" | grep -q "No suggestion"
    then
    echo "[*] This is not a memory file ... exiting"
    exit 1
else
    echo "[*] This is a memory file"
fi

# Find the memory profile and save it into a variable
     PROFILES=$(./vol -f $FILE imageinfo 2>/dev/null | grep "Suggested Profile(s)" | awk -F': ' '{print $2}' | sed 's/(.*)//g' | sed 's/,//g')
     sleep 6
     echo "[*] Suggested profiles found: $PROFILES"
```

```
#  Display the running processes

    echo "[*] Displaying running processes of the file"
    sleep 6
    ./vol -f $FILE pslist 2>/dev/null



# Display network connections

    echo "[*] The memeory file's suggested profiles have the following network coneections"
    sleep 6
    ./vol -f $FILE --profile=$PROFILES connscan 2>/dev/null


# Attempt to extract registry information

# Displaying the registry files
    echo "[*] Identifying and displaying the registry hives 0f the memory file's suggested profiles"
    sleep 6
    ./vol -f $FILE --profile=$PROFILES hivelist 2>/dev/null
```

```
# Displaying Internet Explorer history stores in the memory file
    echo "[*] Identifying and displaying Internet Explorer history stored in the memory file"
    sleep 6
    ./vol -f $FILE --profile=$PROFILES iehistory 2>/dev/null



# Displaying activity and frequently used applications stored in the memory file
    echo "[*] Identifying and dispplaying activity and frequently used applications stored in the memory file"
    sleep 6
    ./vol -f $FILE --profile=$PROFILES userassist 2>/dev/null


    sleep 3


}
```

8. STATISTICS Function

Purpose: Summarizes statistical information and generates a compressed report.

- **Metrics**:
  - Counts directories, `.txt` files, and XML files, recording results in `analysis_report.txt`.
  - Identifies and quantifies files matching the `Strings*` pattern.
- **Report Handling**:
  - Removes existing `analysis_report.txt` and `analysis_report.zip`.
  - Creates a new report and compresses it into `analysis_report.zip`.

```
function STATISTICS ()

{

# Counting the number of directories created

# Save all the results into a report (name, files extracted, etc.).

    rm analysis_report.txt
    rm -rf analysis_report.zip

    DIRECTORIES=$(find -type d | wc -l)
    echo "[*] total number of directories created and extracted is $DIRECTORIES" >> analysis_report.txt

    sleep 3

# Counting the number of text files created
    TEXT=$(find -type f -name "*.txt" | wc -l)
    echo "[*] total number of text files extracted is $TEXT" >> analysis_report.txt

    sleep 3

# Counting the number of string files created

    STRING=$(find -type f -name "Strings*" | wc -l)
    echo "[*] total number of string files extracted is $STRING" >> analysis_report.txt

    sleep 3
```

```
    sleep 3

# Counting the number of xml files created
    XML=$(find -type f -name "*.xml" | wc -l)
    echo "[*] total number of xml files extracted is $XML" >> analysis_report.txt

    sleep 3


#  Zip the extracted files and the report file.

    rm -f analysis_report.zip

    zip analysis_report.zip analysis_report.txt

    echo "Results have been saved to analysis_report.txt and compressed to a zip file"


}

START
TOOLS
CARVERS
ANALYZE
STRINGS
VOLATILITY
STATISTICS
```