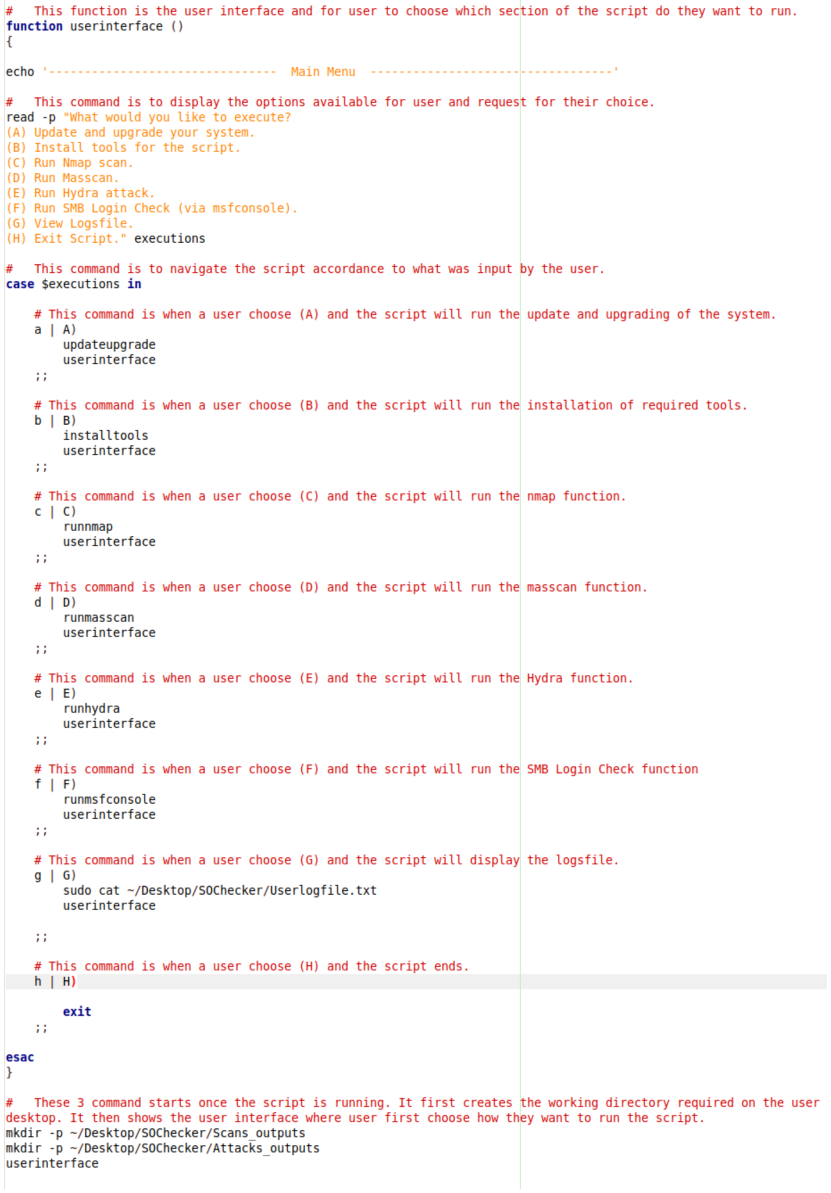
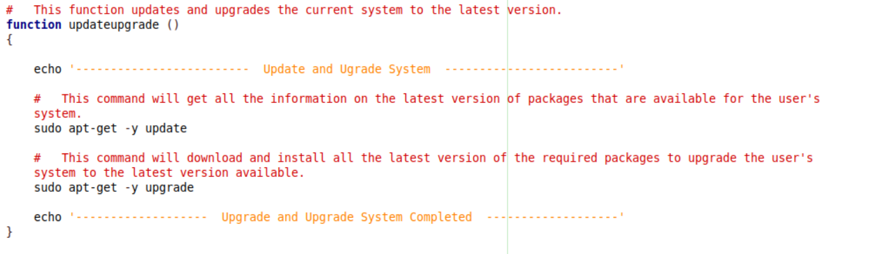
S/N	CODES	FUNCTIONS
1		• When a user runs the script, this is the user interface that the user will first see. • This interface allows user to decide which part of the script do they want to run. a) Update and Upgrade System b) Install relevant tools c) Run Nmap d) Run Masscan e) Run Hydra f) Run Msfconsole g) View Logfile h) Exit Script • A working folder “SOChecker will also be created on the user’s desktop. It will also create 2 folders inside for the scans and attacks.

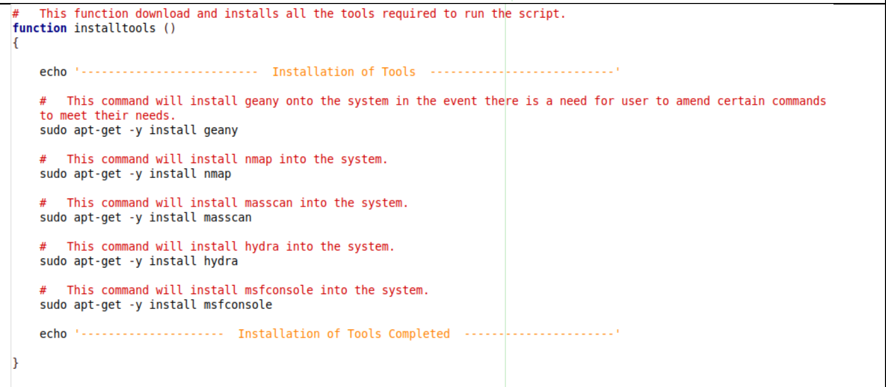
```
#   This function is the user interface and for user to choose which section of the script do they want to run.
function userinterface ()
{

echo '-------------------------------  Main Menu  ----------------------------------'

#   This command is to display the options available for user and request for their choice.
read -p "What would you like to execute?
(A) Update and upgrade your system.
(B) Install tools for the script.
(C) Run Nmap scan.
(D) Run Masscan.
(E) Run Hydra attack.
(F) Run SMB Login Check (via msfconsole).
(G) View Logsfile.
(H) Exit Script." executions

#   This command is to navigate the script accordance to what was input by the user.
case $executions in

    # This command is when a user choose (A) and the script will run the update and upgrading of the system.
    a | A)
        updateupgrade
        userinterface
    ;;

    # This command is when a user choose (B) and the script will run the installation of required tools.
    b | B)
        installtools
        userinterface
    ;;

    # This command is when a user choose (C) and the script will run the nmap function.
    c | C)
        runnmap
        userinterface
    ;;

    # This command is when a user choose (D) and the script will run the masscan function.
    d | D)
        runmasscan
        userinterface
    ;;

    # This command is when a user choose (E) and the script will run the Hydra function.
    e | E)
        runhydra
        userinterface
    ;;

    # This command is when a user choose (F) and the script will run the SMB Login Check function
    f | F)
        runmsfconsole
        userinterface
    ;;

    # This command is when a user choose (G) and the script will display the logsfile.
    g | G)
        sudo cat ~/Desktop/SOChecker/Userlogfile.txt
        userinterface

    ;;

    # This command is when a user choose (H) and the script ends.
    h | H)

        exit
    ;;

esac
}

#   These 3 command starts once the script is running. It first creates the working directory required on the user
desktop. It then shows the user interface where user first choose how they want to run the script.
mkdir -p ~/Desktop/SOChecker/Scans_outputs
mkdir -p ~/Desktop/SOChecker/Attacks_outputs
userinterface
```

<table>
<tr><th>S/N</th><th>CODES</th><th>FUNCTIONS</th></tr>
<tr><td>2</td><td>



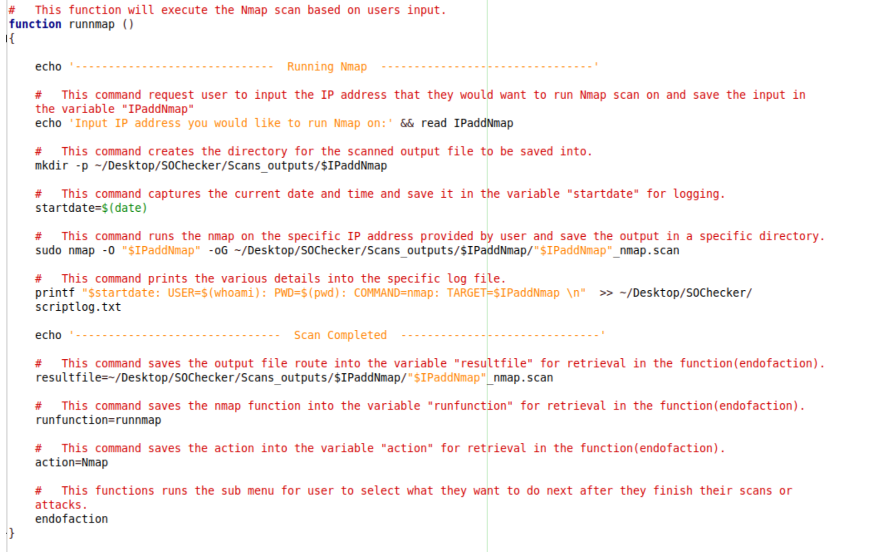
```
#   This function updates and upgrades the current system to the latest version.
function updateupgrade ()
{

    echo '------------------------  Update and Ugrade System  ------------------------'

    #   This command will get all the information on the latest version of packages that are available for the user's
    system.
    sudo apt-get -y update

    #   This command will download and install all the latest version of the required packages to upgrade the user's
    system to the latest version available.
    sudo apt-get -y upgrade

    echo '-------------------  Upgrade and Upgrade System Completed  ------------------'
}
```

</td><td>This function is to update and upgrade the user’s system to the latest version possible to run the script and its tools.</td></tr>
<tr><td>3</td><td>



```
#   This function download and installs all the tools required to run the script.
function installtools ()
{

    echo '-------------------------  Installation of Tools  --------------------------'

    #   This command will install geany onto the system in the event there is a need for user to amend certain commands
    to meet their needs.
    sudo apt-get -y install geany

    #   This command will install nmap into the system.
    sudo apt-get -y install nmap

    #   This command will install masscan into the system.
    sudo apt-get -y install masscan

    #   This command will install hydra into the system.
    sudo apt-get -y install hydra

    #   This command will install msfconsole into the system.
    sudo apt-get -y install msfconsole

    echo '--------------------  Installation of Tools Completed  ---------------------'

}
```

</td><td>This function installs the required tools that is needed for the script.</td></tr>
<tr><td>4</td><td>



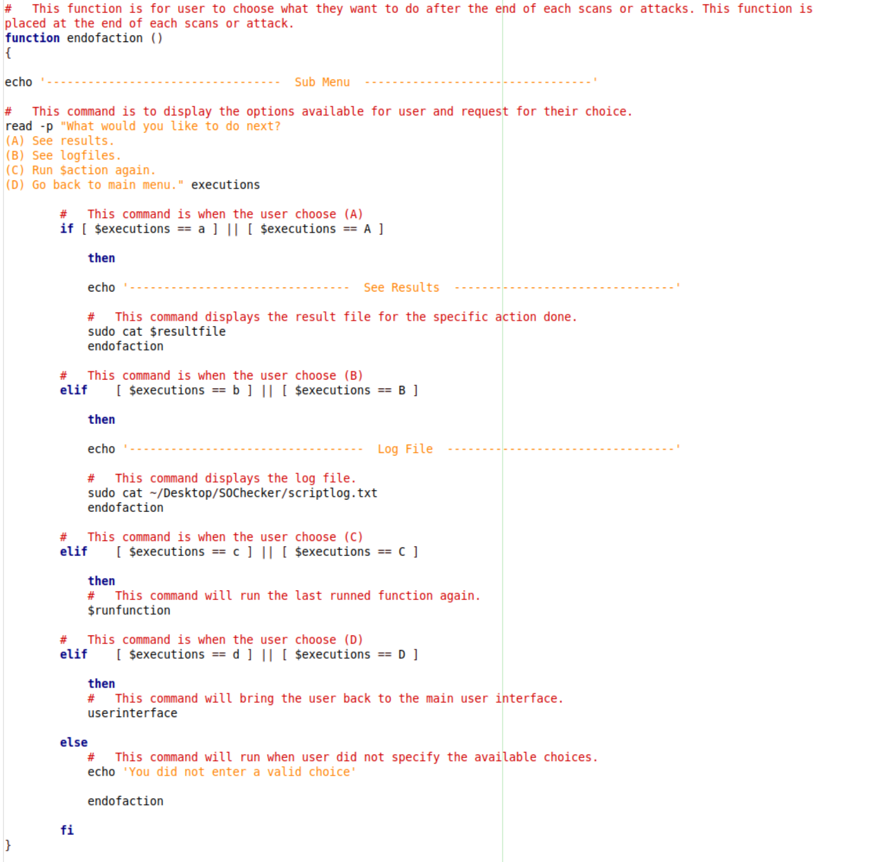
```
#   This function will execute the Nmap scan based on users input.
function runnmap ()
{

    echo '----------------------------  Running Nmap  ------------------------------'

    #   This command request user to input the IP address that they would want to run Nmap scan on and save the input in
    the variable "IPaddNmap"
    echo 'Input IP address you would like to run Nmap on:' && read IPaddNmap

    #   This command creates the directory for the scanned output file to be saved into.
    mkdir -p ~/Desktop/SOChecker/Scans_outputs/$IPaddNmap

    #   This command captures the current date and time and save it in the variable "startdate" for logging.
    startdate=$(date)

    #   This command runs the nmap on the specific IP address provided by user and save the output in a specific directory.
    sudo nmap -O "$IPaddNmap" -oG ~/Desktop/SOChecker/Scans_outputs/$IPaddNmap/"$IPaddNmap"_nmap.scan

    #   This command prints the various details into the specific log file.
    printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=nmap: TARGET=$IPaddNmap \n"  >> ~/Desktop/SOChecker/
    scriptlog.txt

    echo '-----------------------------  Scan Completed  ----------------------------'

    #   This command saves the output file route into the variable "resultfile" for retrieval in the function(endofaction).
    resultfile=~/Desktop/SOChecker/Scans_outputs/$IPaddNmap/"$IPaddNmap"_nmap.scan

    #   This command saves the nmap function into the variable "runfunction" for retrieval in the function(endofaction).
    runfunction=runnmap

    #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
    action=Nmap

    #   This functions runs the sub menu for user to select what they want to do next after they finish their scans or
    attacks.
    endofaction
}
```

</td><td>

- This function runs Nmap after it request the user for the target IP address.
- The results file is saved in the specific folder with the IP address as its name inside the “Scans_outputs” folder.
- It also creates a log entry inside the logsfile.
- After the scan is completed, the sub menu will appear to request for user’s choice:
  a) To see the results of the scan.
  b) To see the script logfile.
  c) Run Nmap again
  d) Exit to main menu

</td></tr>
</table>

<table>
<tr><th>S/N</th><th>CODES</th><th>FUNCTIONS</th></tr>
<tr><td>5</td><td>

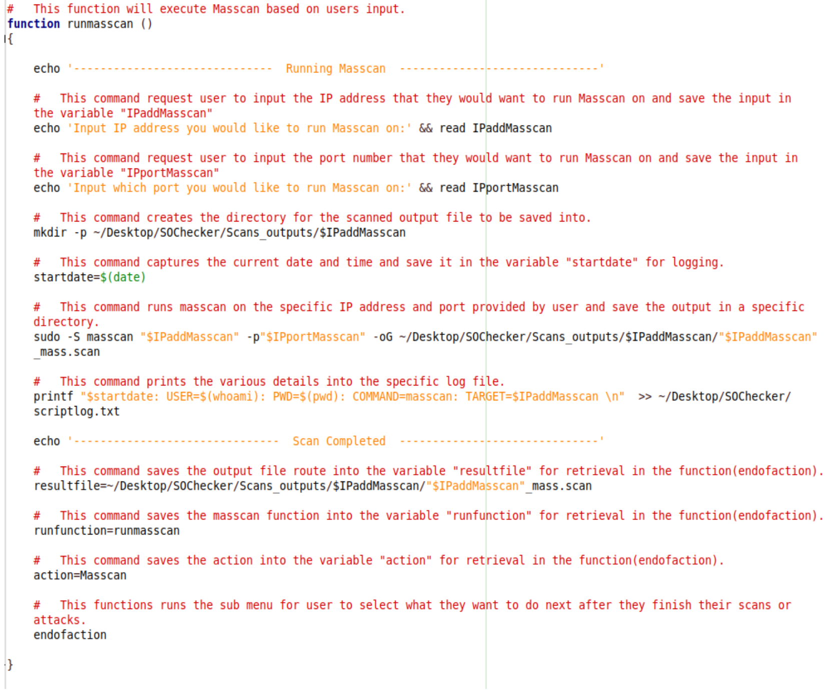


```
#   This function is for user to choose what they want to do after the end of each scans or attacks. This function is
placed at the end of each scans or attack.
function endofaction ()
{

echo '---------------------------------  Sub Menu  ---------------------------------'

#   This command is to display the options available for user and request for their choice.
read -p "What would you like to do next?
(A) See results.
(B) See logfiles.
(C) Run $action again.
(D) Go back to main menu." executions

        #   This command is when the user choose (A)
        if [ $executions == a ] || [ $executions == A ]

            then

            echo '--------------------------------  See Results  --------------------------------'

            #   This command displays the result file for the specific action done.
            sudo cat $resultfile
            endofaction

        #   This command is when the user choose (B)
        elif    [ $executions == b ] || [ $executions == B ]

            then

            echo '----------------------------------  Log File  ---------------------------------'

            #   This command displays the log file.
            sudo cat ~/Desktop/SOChecker/scriptlog.txt
            endofaction

        #   This command is when the user choose (C)
        elif    [ $executions == c ] || [ $executions == C ]

            then
            #   This command will run the last runned function again.
            $runfunction

        #   This command is when the user choose (D)
        elif    [ $executions == d ] || [ $executions == D ]

            then
            #   This command will bring the user back to the main user interface.
            userinterface

        else
            #   This command will run when user did not specify the available choices.
            echo 'You did not enter a valid choice'

            endofaction

        fi
}
```

</td><td>

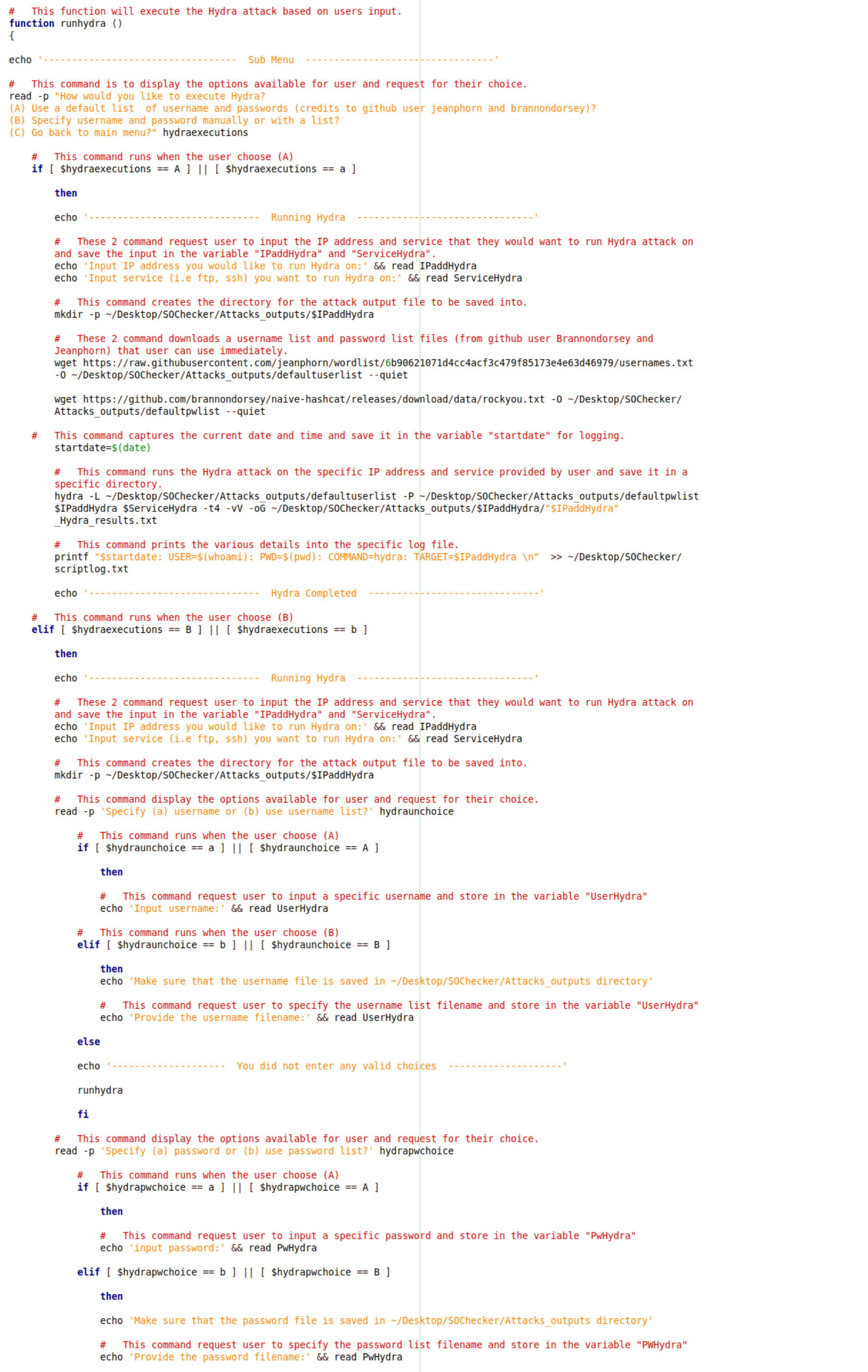
- This function is placed at the end of all attacks and scans function so that user can choose if they want to see the results or the log file.

</td></tr>
</table>

S/N	CODES	FUNCTIONS
6		• This function runs Masscan after it request the user for the target IP address and port number. • The results file is saved in the specific folder with the IP address as its name inside the “Scans_outputs” folder. • It also creates a log entry inside the logsfile. • After the scan is completed, the sub menu will appear to request for user’s choice: e) To see the results of the scan. f) To see the script logfile. g) Run Nmap again h) Exit to main menu

```
#   This function will execute Masscan based on users input.
function runmasscan ()
{

    echo '-----------------------------  Running Masscan  -----------------------------'

    #   This command request user to input the IP address that they would want to run Masscan on and save the input in
    the variable "IPaddMasscan"
    echo 'Input IP address you would like to run Masscan on:' && read IPaddMasscan

    #   This command request user to input the port number that they would want to run Masscan on and save the input in
    the variable "IPportMasscan"
    echo 'Input which port you would like to run Masscan on:' && read IPportMasscan

    #   This command creates the directory for the scanned output file to be saved into.
    mkdir -p ~/Desktop/SOChecker/Scans_outputs/$IPaddMasscan

    #   This command captures the current date and time and save it in the variable "startdate" for logging.
    startdate=$(date)

    #   This command runs masscan on the specific IP address and port provided by user and save the output in a specific
    directory.
    sudo -S masscan "$IPaddMasscan" -p"$IPportMasscan" -oG ~/Desktop/SOChecker/Scans_outputs/$IPaddMasscan/"$IPaddMasscan"
    _mass.scan

    #   This command prints the various details into the specific log file.
    printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=masscan: TARGET=$IPaddMasscan \n"  >> ~/Desktop/SOChecker/
    scriptlog.txt

    echo '-----------------------------  Scan Completed  -----------------------------'

    #   This command saves the output file route into the variable "resultfile" for retrieval in the function(endofaction).
    resultfile=~/Desktop/SOChecker/Scans_outputs/$IPaddMasscan/"$IPaddMasscan"_mass.scan

    #   This command saves the masscan function into the variable "runfunction" for retrieval in the function(endofaction).
    runfunction=runmasscan

    #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
    action=Masscan

    #   This functions runs the sub menu for user to select what they want to do next after they finish their scans or
    attacks.
    endofaction

}
```

7



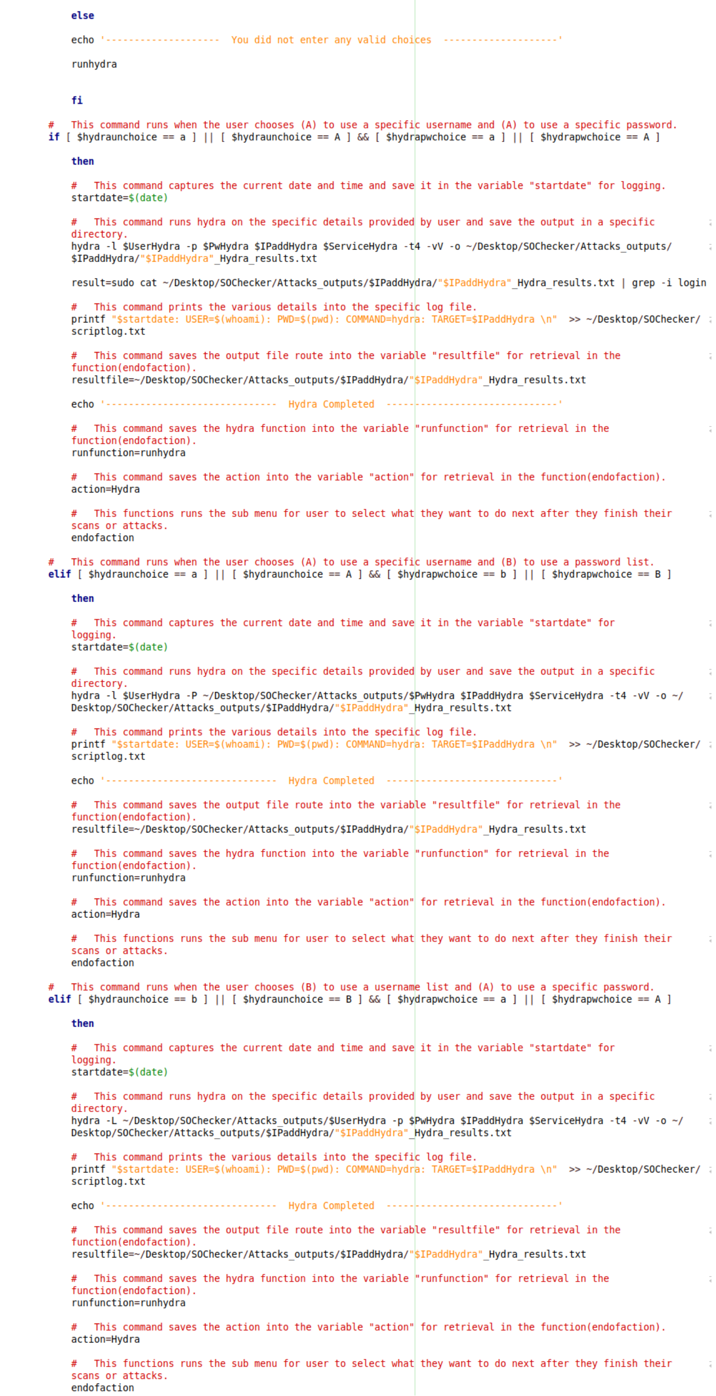
```
#   This function will execute the Hydra attack based on users input.
function runhydra ()
{

echo '--------------------------------  Sub Menu  --------------------------------'

#   This command is to display the options available for user and request for their choice.
read -p "How would you like to execute Hydra?
(A) Use a default list  of username and passwords (credits to github user jeanphorn and brannondorsey)?
(B) Specify username and password manually or with a list?
(C) Go back to main menu?" hydraexecutions

    #   This command runs when the user choose (A)
    if [ $hydraexecutions == A ] || [ $hydraexecutions == a ]

        then

        echo '----------------------------  Running Hydra  --------------------------------'

        #   These 2 command request user to input the IP address and service that they would want to run Hydra attack on
        and save the input in the variable "IPaddHydra" and "ServiceHydra".
        echo 'Input IP address you would like to run Hydra on:' && read IPaddHydra
        echo 'Input service (i.e ftp, ssh) you want to run Hydra on:' && read ServiceHydra

        #   This command creates the directory for the attack output file to be saved into.
        mkdir -p ~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra

        #   These 2 command downloads a username list and password list files (from github user Brannondorsey and
        Jeanphorn) that user can use immediately.
        wget https://raw.githubusercontent.com/jeanphorn/wordlist/6b90621071d4cc4acf3c479f85173e4e63d46979/usernames.txt
        -O ~/Desktop/SOChecker/Attacks_outputs/defaultuserlist --quiet

        wget https://github.com/brannondorsey/naive-hashcat/releases/download/data/rockyou.txt -O ~/Desktop/SOChecker/
        Attacks_outputs/defaultpwlist --quiet

    #   This command captures the current date and time and save it in the variable "startdate" for logging.
        startdate=$(date)

        #   This command runs the Hydra attack on the specific IP address and service provided by user and save it in a
        specific directory.
        hydra -L ~/Desktop/SOChecker/Attacks_outputs/defaultuserlist -P ~/Desktop/SOChecker/Attacks_outputs/defaultpwlist
        $IPaddHydra $ServiceHydra -t4 -vV -oG ~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"
        _Hydra_results.txt

        #   This command prints the various details into the specific log file.
        printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddHydra \n"  >> ~/Desktop/SOChecker/
        scriptlog.txt

        echo '----------------------------  Hydra Completed  ----------------------------'

    #   This command runs when the user choose (B)
    elif [ $hydraexecutions == B ] || [ $hydraexecutions == b ]

        then

        echo '----------------------------  Running Hydra  --------------------------------'

        #   These 2 command request user to input the IP address and service that they would want to run Hydra attack on
        and save the input in the variable "IPaddHydra" and "ServiceHydra".
        echo 'Input IP address you would like to run Hydra on:' && read IPaddHydra
        echo 'Input service (i.e ftp, ssh) you want to run Hydra on:' && read ServiceHydra

        #   This command creates the directory for the attack output file to be saved into.
        mkdir -p ~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra

        #   This command display the options available for user and request for their choice.
        read -p 'Specify (a) username or (b) use username list?' hydrauchoice

            #   This command runs when the user choose (A)
            if [ $hydrauchoice == a ] || [ $hydrauchoice == A ]

                then

                #   This command request user to input a specific username and store in the variable "UserHydra"
                echo 'Input username:' && read UserHydra

            #   This command runs when the user choose (B)
            elif [ $hydrauchoice == b ] || [ $hydrauchoice == B ]

                then
                echo 'Make sure that the username file is saved in ~/Desktop/SOChecker/Attacks_outputs directory'

                #   This command request user to specify the username list filename and store in the variable "UserHydra"
                echo 'Provide the username filename:' && read UserHydra

            else

            echo '-------------------  You did not enter any valid choices  -------------------'

            runhydra

            fi

        #   This command display the options available for user and request for their choice.
        read -p 'Specify (a) password or (b) use password list?' hydrapwchoice

            #   This command runs when the user choose (A)
            if [ $hydrapwchoice == a ] || [ $hydrapwchoice == A ]

                then

                #   This command request user to input a specific password and store in the variable "PwHydra"
                echo 'input password:' && read PwHydra

            elif [ $hydrapwchoice == b ] || [ $hydrapwchoice == B ]

                then

                echo 'Make sure that the password file is saved in ~/Desktop/SOChecker/Attacks_outputs directory'

                #   This command request user to specify the password list filename and store in the variable "PWHydra"
                echo 'Provide the password filename:' && read PwHydra
```

- This function runs Hydra, but the script first provides a sub menu for user to choose how they would like to run hydra.
  - a) Download a username and password file (from github user jeanphorn and Brannondorsey) and run with it.
  - b) Provide a specific username and password or provide users own list.
  - c) Go back to main menu

- When user chooses (a), the script will prompt user to input the target IP address and also download a password and username list from the github users and store it in the working folder and run it.

- The results file is saved in the specific folder with the IP address as its name inside the “Attacks_outputs” folder.

- It also creates a log entry inside the logsfile.



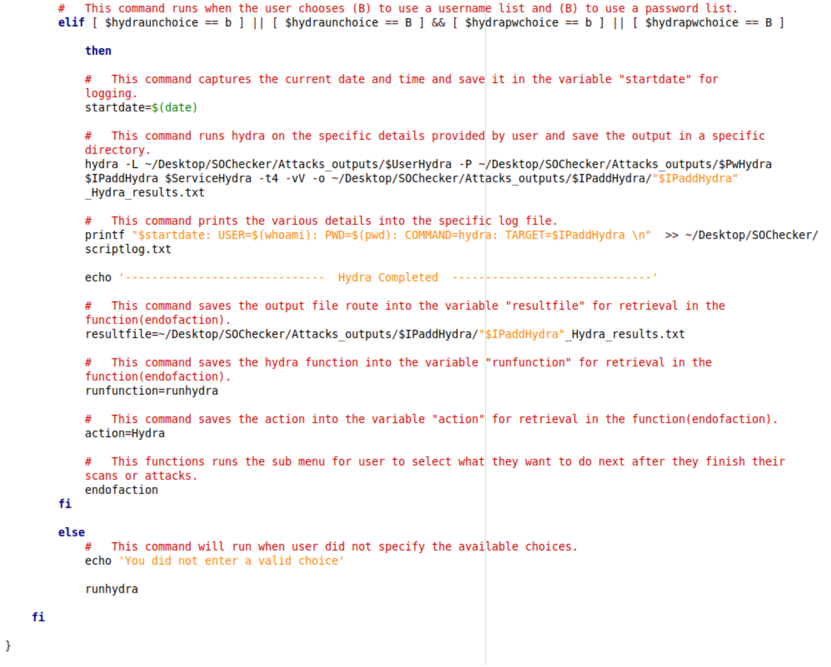
```
    else

    echo '-------------------  You did not enter any valid choices  -------------------'

    runhydra


    fi

#   This command runs when the user chooses (A) to use a specific username and (A) to use a specific password.
if [ $hydraunchoice == a ] || [ $hydraunchoice == A ] && [ $hydrapwchoice == a ] || [ $hydrapwchoice == A ]

    then

    #   This command captures the current date and time and save it in the variable "startdate" for logging.
    startdate=$(date)

    #   This command runs hydra on the specific details provided by user and save the output in a specific
    directory.
    hydra -l $UserHydra -p $PwHydra $IPaddHydra $ServiceHydra -t4 -vV -o ~/Desktop/SOChecker/Attacks_outputs/
    $IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    result=sudo cat ~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt | grep -i login

    #   This command prints the various details into the specific log file.
    printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddHydra \n"  >> ~/Desktop/SOChecker/
    scriptlog.txt

    #   This command saves the output file route into the variable "resultfile" for retrieval in the
    function(endofaction).
    resultfile=~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    echo '-----------------------------  Hydra Completed  -----------------------------'

    #   This command saves the hydra function into the variable "runfunction" for retrieval in the
    function(endofaction).
    runfunction=runhydra

    #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
    action=Hydra

    #   This functions runs the sub menu for user to select what they want to do next after they finish their
    scans or attacks.
    endofaction

#   This command runs when the user chooses (A) to use a specific username and (B) to use a password list.
elif [ $hydraunchoice == a ] || [ $hydraunchoice == A ] && [ $hydrapwchoice == b ] || [ $hydrapwchoice == B ]

    then

    #   This command captures the current date and time and save it in the variable "startdate" for
    logging.
    startdate=$(date)

    #   This command runs hydra on the specific details provided by user and save the output in a specific
    directory.
    hydra -l $UserHydra -P ~/Desktop/SOChecker/Attacks_outputs/$PwHydra $IPaddHydra $ServiceHydra -t4 -vV -o ~/
    Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    #   This command prints the various details into the specific log file.
    printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddHydra \n"  >> ~/Desktop/SOChecker/
    scriptlog.txt

    echo '-----------------------------  Hydra Completed  -----------------------------'

    #   This command saves the output file route into the variable "resultfile" for retrieval in the
    function(endofaction).
    resultfile=~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    #   This command saves the hydra function into the variable "runfunction" for retrieval in the
    function(endofaction).
    runfunction=runhydra

    #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
    action=Hydra

    #   This functions runs the sub menu for user to select what they want to do next after they finish their
    scans or attacks.
    endofaction

#   This command runs when the user chooses (B) to use a username list and (A) to use a specific password.
elif [ $hydraunchoice == b ] || [ $hydraunchoice == B ] && [ $hydrapwchoice == a ] || [ $hydrapwchoice == A ]

    then

    #   This command captures the current date and time and save it in the variable "startdate" for
    logging.
    startdate=$(date)

    #   This command runs hydra on the specific details provided by user and save the output in a specific
    directory.
    hydra -L ~/Desktop/SOChecker/Attacks_outputs/$UserHydra -p $PwHydra $IPaddHydra $ServiceHydra -t4 -vV -o ~/
    Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    #   This command prints the various details into the specific log file.
    printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddHydra \n"  >> ~/Desktop/SOChecker/
    scriptlog.txt

    echo '-----------------------------  Hydra Completed  -----------------------------'

    #   This command saves the output file route into the variable "resultfile" for retrieval in the
    function(endofaction).
    resultfile=~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

    #   This command saves the hydra function into the variable "runfunction" for retrieval in the
    function(endofaction).
    runfunction=runhydra

    #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
    action=Hydra

    #   This functions runs the sub menu for user to select what they want to do next after they finish their
    scans or attacks.
    endofaction
```

- When user choose (b), the script will prompt user to input the target IP address, the port and how they want to input the username and password.

- The script allows the flexibility for user to either input a specific password and username or to use a list. (Just make sure that the username and password list is at the stated directory for the script to retrieve.)

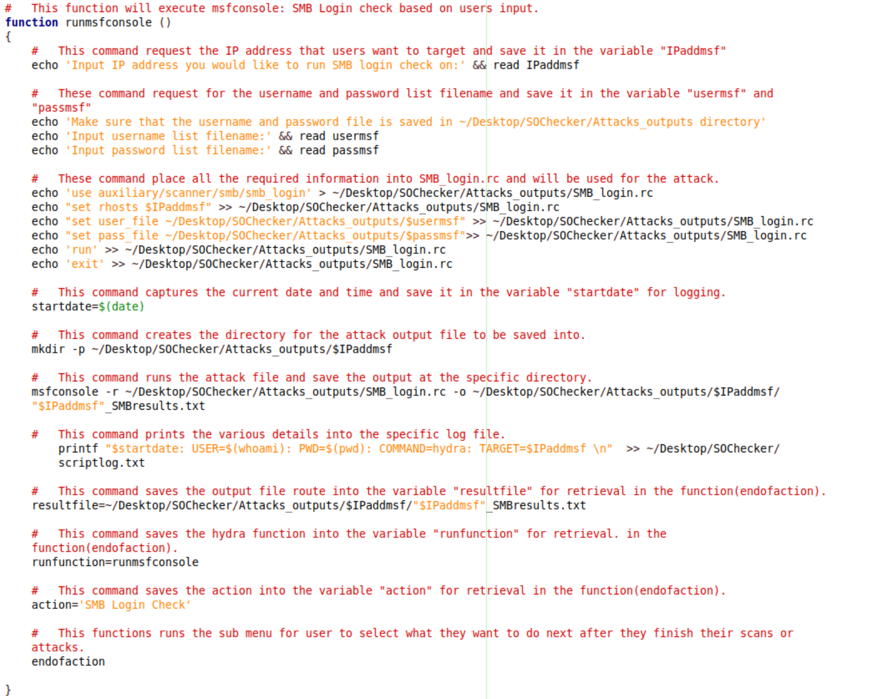
- The results file is saved in the specific folder with the IP address as its name inside the “Attacks_outputs” folder.

- It also creates a log entry inside the logsfile.

S/N	CODES	FUNCTIONS
		

```
        #   This command runs when the user chooses (B) to use a username list and (B) to use a password list.
        elif [ $hydraunchoice == b ] || [ $hydraunchoice == B ] && [ $hydrapwchoice == b ] || [ $hydrapwchoice == B ]

            then

            #   This command captures the current date and time and save it in the variable "startdate" for
            logging.
            startdate=$(date)

            #   This command runs hydra on the specific details provided by user and save the output in a specific
            directory.
            hydra -L ~/Desktop/SOChecker/Attacks_outputs/$UserHydra -P ~/Desktop/SOChecker/Attacks_outputs/$PwHydra
            $IPaddHydra $ServiceHydra -t4 -vV -o ~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"
            _Hydra_results.txt

            #   This command prints the various details into the specific log file.
            printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddHydra \n"  >> ~/Desktop/SOChecker/
            scriptlog.txt

            echo '------------------------------  Hydra Completed  ------------------------------'

            #   This command saves the output file route into the variable "resultfile" for retrieval in the
            function(endofaction).
            resultfile=~/Desktop/SOChecker/Attacks_outputs/$IPaddHydra/"$IPaddHydra"_Hydra_results.txt

            #   This command saves the hydra function into the variable "runfunction" for retrieval in the
            function(endofaction).
            runfunction=runhydra

            #   This command saves the action into the variable "action" for retrieval in the function(endofaction).
            action=Hydra

            #   This functions runs the sub menu for user to select what they want to do next after they finish their
            scans or attacks.
            endofaction
        fi

        else
            #   This command will run when user did not specify the available choices.
            echo 'You did not enter a valid choice'

            runhydra

    fi

}
```

S/N	CODES	FUNCTIONS

8	
	# This function will execute msfconsole: SMB Login check based on users input.
	function runmsfconsole ()
	{
	# This command request the IP address that users want to target and save it in the variable "IPaddmsf"
	echo 'Input IP address you would like to run SMB login check on:' && read IPaddmsf
	
	# These command request for the username and password list filename and save it in the variable "usermsf" and "passmsf"
	echo 'Make sure that the username and password file is saved in ~/Desktop/SOChecker/Attacks_outputs directory'
	echo 'Input username list filename:' && read usermsf
	echo 'Input password list filename:' && read passmsf
	
	# These command place all the required information into SMB_login.rc and will be used for the attack.
	echo 'use auxiliary/scanner/smb/smb_login' > ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	echo "set rhosts $IPaddmsf" >> ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	echo "set user_file ~/Desktop/SOChecker/Attacks_outputs/$usermsf" >> ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	echo "set pass_file ~/Desktop/SOChecker/Attacks_outputs/$passmsf">> ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	echo 'run' >> ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	echo 'exit' >> ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc
	
	# This command captures the current date and time and save it in the variable "startdate" for logging.
	startdate=$(date)
	
	# This command creates the directory for the attack output file to be saved into.
	mkdir -p ~/Desktop/SOChecker/Attacks_outputs/$IPaddmsf
	
	# This command runs the attack file and save the output at the specific directory.
	msfconsole -r ~/Desktop/SOChecker/Attacks_outputs/SMB_login.rc -o ~/Desktop/SOChecker/Attacks_outputs/$IPaddmsf/"$IPaddmsf"_SMBresults.txt
	
	# This command prints the various details into the specific log file.
	printf "$startdate: USER=$(whoami): PWD=$(pwd): COMMAND=hydra: TARGET=$IPaddmsf \n" >> ~/Desktop/SOChecker/scriptlog.txt
	
	# This command saves the output file route into the variable "resultfile" for retrieval in the function(endofaction).
	resultfile=~/Desktop/SOChecker/Attacks_outputs/$IPaddmsf/"$IPaddmsf"_SMBresults.txt
	
	# This command saves the hydra function into the variable "runfunction" for retrieval. in the function(endofaction).
	runfunction=runmsfconsole
	
	# This command saves the action into the variable "action" for retrieval in the function(endofaction).
	action='SMB Login Check'
	
	# This functions runs the sub menu for user to select what they want to do next after they finish their scans or attacks.
	endofaction
	
	}
		• This function runs msfconsole (SMB login) module. It first request user to input the target IP address and followed by the username and password list to be used for the attack.
		
		• The results file is saved in the specific folder with the IP address as its name inside the “Attacks_outputs” folder.
		
		• It also creates a log entry inside the logsfile.