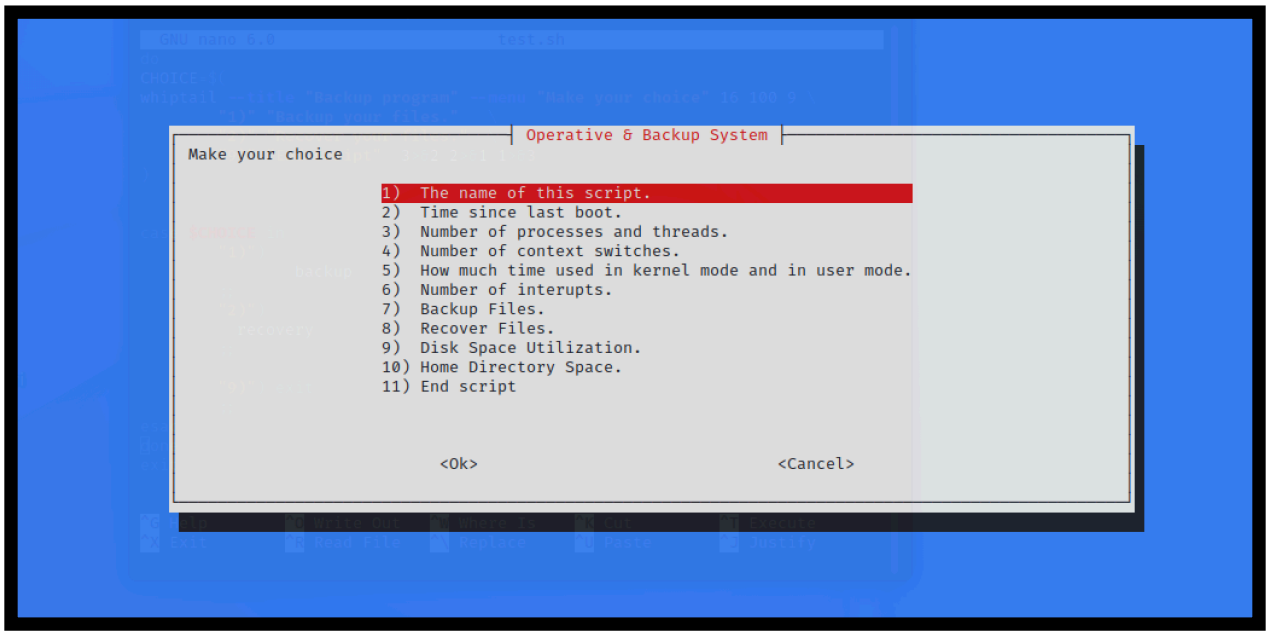
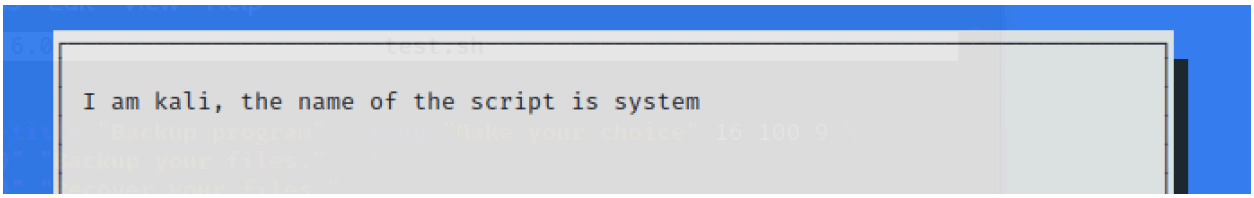
Bash Script
Bailey Williams

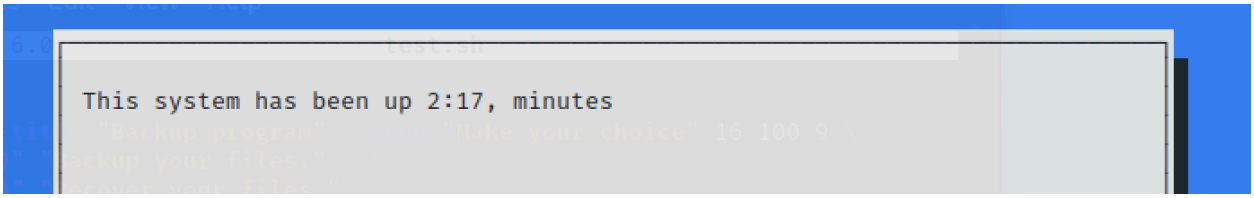
Operative & System Information Script:

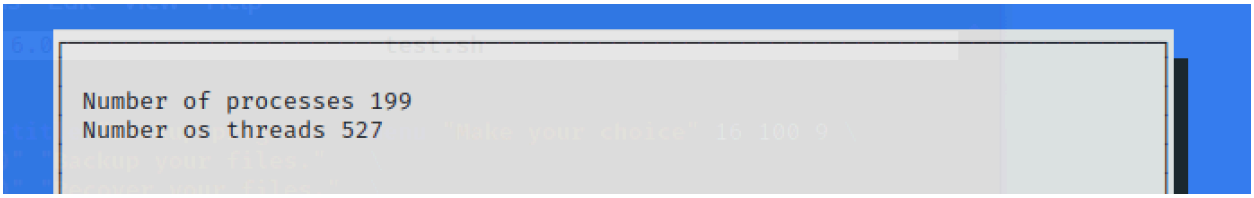
The created script contains many functions which can become useful to an administrator due to its simplicity in getting information in regards to a system's operations, storage, and usage. This script tells the administrator information that includes the following: backing up & recovering files, see ongoing context switching done by the CPU between processes, the # of threads, processes, & interrupts, the users uptime including user & kernel mode, and finally the disk and home space. With these tools, an administrator would be able to see the activity of different users as well as give an idea as to what they are doing which is important for defensive security purposes. This information can help them identify what users and doing their jobs correctly and which ones are abusing the system or getting into information they shouldn't be. It not only gives the information about a users activity but also how much storage space has been utilized so that it does not become an issue. Another benefit of this script is its ability to backup and recover files in a directory which will help with the maintenance of data. The script can easily be used manually with the *bash system.sh* command which will automatically initiate a backup of file and then display a menu using whiptail. This menu is easy to navigate and informative by data being displayed depending on which selection you make. This menu can be seen in the first screenshot below. I also included a simple firewall using iptables but it will only be implemented when the script is ran by the root user, if ran by another user it will tell you in the Terminal Emulator that it can only be used by the root user. The Cron Job tool also allows this script to be ran every minute, hours, or whatever you set it to which will ensure that files are being backed up. The system.sh file will also be uploaded with the report.

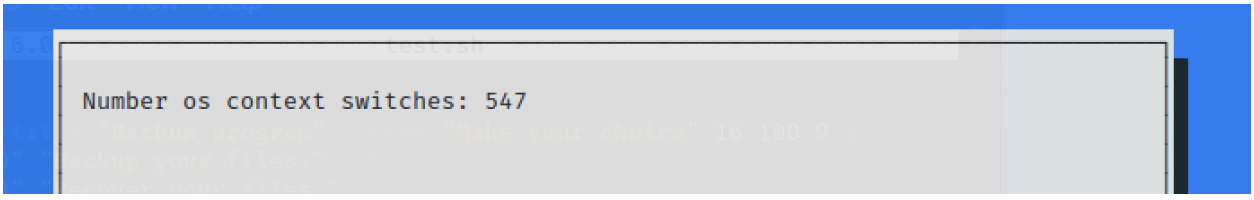
Manual Testing:











```
Percentage in usermode: 40%
and in kernelmode: 60%
```

```
Number of interupts: 359
```

```
Disk space utilization: Filesystem      Size  Used Avail Use% Mounted on
udev            933M     0  933M   0% /dev
tmpfs           195M  1.2M  194M   1% /run
/dev/sda1        78G   17G   58G  23% /
tmpfs           974M     0  974M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           195M   68K  195M   1% /run/user/1000
```

```
Home directory space: 3987360/home
```

```
(base) ┌──(kali㉿x86_64-conda-linux-gnu)-[~/SystemSecurity/backup]
└─$ ls
B20220226205322  B20220226211127  B20220227193157  B20220228011617  B20220228014105  B20220228015102  B20220228022901  backup.txt
B20220226210956  B20220226211525  B20220228002853  B20220228013813  B20220228015000  B20220228022801  B20220228023001
```

```
(base) ┌──(kali㉿x86_64-conda-linux-gnu)-[~/SystemSecurity/backup]
└─$ ls
B20220226205322  B20220226211127  B20220227193157  B20220228011617  B20220228014105  B20220228015102  B20220228022901  backup.txt
B20220226210956  B20220226211525  B20220228002853  B20220228013813  B20220228015000  B20220228022801  B20220228023001

(base) ┌──(kali㉿x86_64-conda-linux-gnu)-[~/SystemSecurity/backup]
└─$ cd B20220226205322

(base) ┌──(kali㉿x86_64-conda-linux-gnu)-[~/SystemSecurity/backup/B20220226205322]
└─$ cat working.txt
This is the working directory.
```

Cron Job - Every Minute:

```
# m h  dom mon dow   command
* * * * * /home/kali/SystemSecurity/system.sh
```

```
backup_2022-02-28_01_51_03am.txt
backup_2022-02-28_02_28_03am.txt
backup_2022-02-28_02_29_03am.txt
backup_2022-02-28_02_30_03am.txt
```

Cron Job - Every Hour:

```
# m h  dom mon dow   command
0 * * * * /home/kali/SystemSecurity/system.sh
```

```
backup_2022-02-28_03_00_00am.txt
backup_2022-02-28_04_00_00am.txt
backup_2022-02-28_05_00_00am.txt
backup_2022-02-28_06_00_00am.txt
backup_2022-02-28_07_00_00am.txt
backup_2022-02-28_08_00_00am.txt
backup_2022-02-28_09_00_00am.txt
```

System.sh

```
#! /bin/bash

#==================================
# System Information & Operations
# Bailey Williams
#==================================
# This program is demonstrating a script which a system administrator may find useful when tracking
# different users activity, storage, and data. A system administrator can use this script to
# backup & recover files, see context switching, the # of threads, the # of processes, the
# # of interrupts, the users uptime including in kernal mode, and finally the disk and home space.

clear

backup_dir="backup"         #dir that will have the backups
working_dir="working"       #dir that is to be backed up
recovery_dir="recovery"     #dir where you want your recovery to be copied to
temp_dir="temp"             #needed temp dir

#function to backup files in working dir
function backup {
    {
    new_backup=B$(date +"%Y%m%d%H%M%S")
    if [ "$(ls -A $backup_dir)" ]; then
        mkdir $new_backup
        rsync -a  $working_dir"/" $temp_dir
        for entry in "$backup_dir"/*
        do
          rm -r $new_backup
          rsync -a  --compare-dest=../$entry/ $temp_dir/ $new_backup
          rm -r $temp_dir
          rsync -a $new_backup/ $temp_dir
        done
        mv $new_backup $backup_dir/$new_backup
        rm -r $temp_dir
    else
        rsync -av  $working_dir"/" backup/$new_backup
    fi
    } | whiptail --gauge "Backing up data ..." 10 60 0
}

```

```
#function that recovers files using the backup dir
function recovery {
    {
      if [ "$(ls -A $backup_dir)" ]; then
    mkdir $temp_dir
    for entry in "$backup_dir"/*
    do
      rsync -av  --compare-dest=../$temp_dir/ $entry/ $recovery_dir
      rm -r $temp_dir
      rsync -av $recovery_dir/ $temp_dir/
      done
      rm -r $temp_dir
    else
      echo "No backup found"
    fi
    } | whiptail --gauge "Recovering data ..." 10 60 0
}

#function gets the # of context switching going on
function contextSwitch {
    {
    ctxt1=$(grep ctxt /proc/stat | awk '{print $2}')
        echo 50
    sleep 1
        ctxt2=$(grep ctxt /proc/stat | awk '{print $2}')
        ctxt=$(($ctxt2 - $ctxt1))
        result="Number os context switches: $ctxt"
    echo $result > result
    } | whiptail --gauge "Getting data ..." 10 60 0
}
```

```
#function gets the uptime in user and kernel made
function userKernelMode {
    {
    raw=( $(grep "cpu " /proc/stat) )
        userfirst=$((${raw[1]} + ${raw[2]}))
        kernelfirst=${raw[3]}
    echo 50
        sleep 1
    raw=( $(grep "cpu " /proc/stat) )
        user=$(( $((${raw[1]} + ${raw[2]})) - $userfirst ))
    echo 90
        kernel=$(( ${raw[3]} - $kernelfirst ))
        sum=$(($kernel + $user))
        result="Percentage in usermode: \
        $(( $user*100)/$sum ))% \
        \nand in kernelmode: $((($kernel*100)/$sum ))%"
    echo $result > result
    echo 100
    } | whiptail --gauge "Getting data ..." 10 60 0
}

#function gets the # of interupts
function interrupts {
    {
    ints=$(vmstat 1 2 | tail -1 | awk '{print $11}')
        result="Number of interupts:  $ints"
    echo 100
    echo $result > result
    } | whiptail --gauge "Getting data ..." 10 60 50
}

#function gets disk space utilization
function system_ds {
        ds=$(df -h)
}| whiptail --gauge "Getting data ..." 10 60 0

#function gets home space of user
function home_space {
    #only superuser can get this information
    if [[ $(id -u ) == 0 ]]; then
        homespace=$(du -s /home* | sort -nr)
    fi
}| whiptail --gauge "Getting data ..." 10 60 0

```

```
117
118 # simple firewall using iptables
119 # will only be run if done by root user
120 iptables -F
121 iptables -X
122 iptables -t nat -F
123 iptables -t nat -X
124 iptables -t mangle -F
125 iptables -t mangle -X
126 modprobe ip_conntrack
127 modprobe ip_conntrack_ftp
128
129 # default filter
130 iptables -P INPUT DROP
131 iptables -P OUTPUT ACCEPT
132
133 #  access to loop back
134 iptables -A INPUT -i lo -j ACCEPT
135 iptables -A OUTPUT -o lo -j ACCEPT
136
137 # allow  UDP, DNS and Passive FTP
138 iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
139
140 # drop and log it
141 iptables -A INPUT -j LOG
142 iptables -A INPUT -j DROP
143
144 backup
145 cat > "backup_$(date '+%F_%I_%M_%S%P').txt" <<- _EOF_
146         System Information
147     $(date)
148     uptime=$(uptime | awk '{print $3;}')
149     processes=$(ps ax | wc -l)
150     threads=$(ps amx | wc -l)
151     diskspace=$(df -h)
152     homespace=$(du -s /home* | sort -nr)
153     Directory backup complete.
154
155 _EOF_
156
```

```
while [ 1 ]
do
CHOICE=$(
whiptail --title "Operative & Backup System" --menu "Make your choice" 20 100 12 \
    "1)" "The name of this script."    \
    "2)" "Time since last boot."  \
    "3)" "Number of processes and threads." \
    "4)" "Number of context switches." \
    "5)" "How much time used in kernel mode and in user mode." \
    "6)" "Number of interrupts." \
    "7)" "Backup Files." \
    "8)" "Recover Files." \
    "9)" "Disk Space Utilization." \
    "10)" "Home Directory Space." \
    "11)" "End script"  3>&2 2>&1 1>&3
)


result=$(whoami)
case $CHOICE in
    "1)")
        result="I am $result, the name of the script is system"
    ;;
    "2)")
            OP=$(uptime | awk '{print $3;}')
        result="This system has been up $OP minutes"
    ;;

    "3)")
            p=$(ps ax | wc -l)
                t=$(ps amx | wc -l)
        result="Number of processes $p\nNumber os threads $t"
        ;;

    "4)")
            contextSwitch
        read -r result < result
        ;;

    "5)")
                userKernelMode
        read -r result < result
        ;;

    "6)")
        interrupts
        read -r result < result
        ;;
    "7)")
        backup
    ;;
    "8)")
        recovery
    ;;
    "9)")
        ds=$(df -h)
        result="Disk space utilization: $ds"
        ;;
    "10)")
        homespace=$(du -s /home* | sort -nr)
        result="Home directory space: $homespace"
        ;;
    "11)") exit
        ;;
esac
whiptail --msgbox "$result" 30 78
done
exit
```