SSH Intrusion Detector Design

Gary Khodayari and Yuri Elt
February 8th, 2022

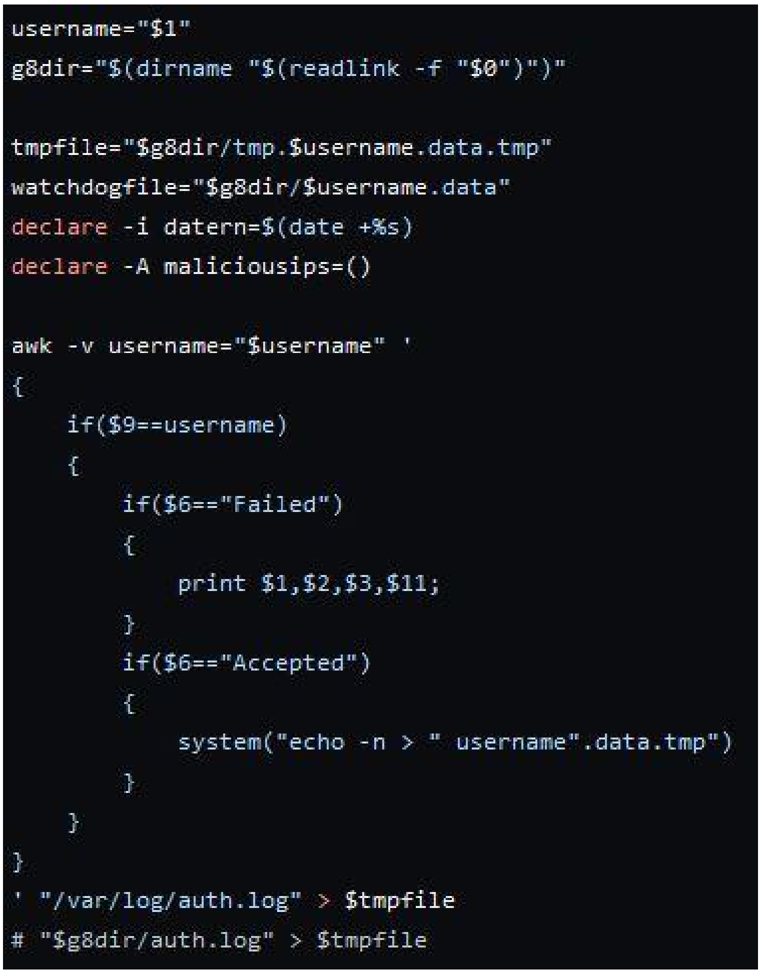
Application Process Steps

Monitoring System Logins

parser.sh

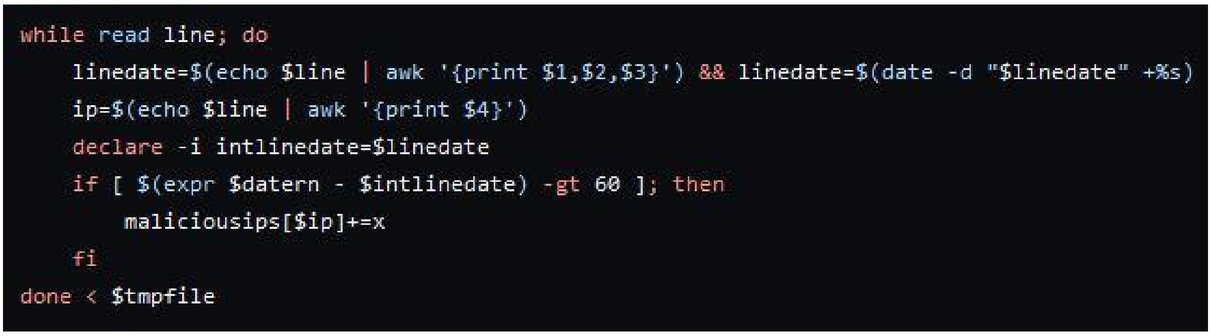
The purpose of parser.sh is to get information with a timestamp from users who enter the wrong password.

This is the design of parser.sh.



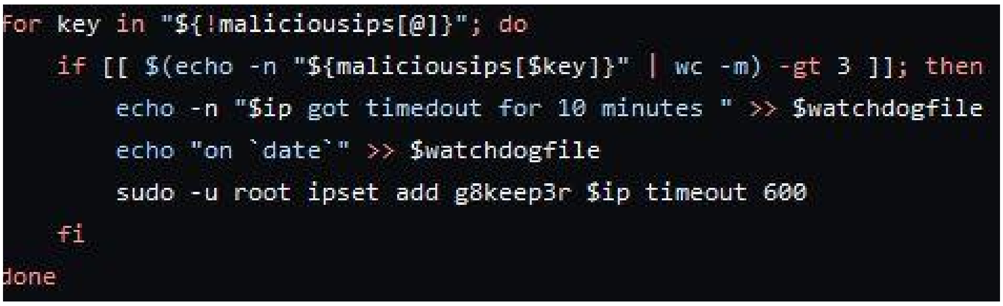
```
username="$1"
g8dir="$(dirname "$(readlink -f "$0")")"

tmpfile="$g8dir/tmp.$username.data.tmp"
watchdogfile="$g8dir/$username.data"
declare -i datern=$(date +%s)
declare -A maliciousips=()

awk -v username="$username" '
{
    if($9==username)
    {
        if($6=="Failed")
        {
            print $1,$2,$3,$11;
        }
        if($6=="Accepted")
        {
            system("echo -n > " username".data.tmp")
        }
    }
}
' "/var/log/auth.log" > $tmpfile
# "$g8dir/auth.log" > $tmpfile
```

This part is declared variables to create watchdog and maliciousisp to retrieve the specific username from var/log/auth.log and turn them into the value.
Set username = “$1” as the assigned variable.
G8dir = set the home directory where this script is in.



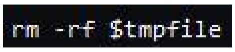
```
while read line; do
    linedate=$(echo $line | awk '{print $1,$2,$3}') && linedate=$(date -d "$linedate" +%s)
    ip=$(echo $line | awk '{print $4}')
    declare -i intlinedate=$linedate
    if [ $(expr $datern - $intlinedate) -gt 60 ]; then
        maliciousips[$ip]+=x
    fi
done < $tmpfile
```

This part is the print timestamp when the program identified possible malicious IP and creates output.



```
for key in "${!maliciousips[@]}"; do
    if [[ $(echo -n "${maliciousips[$key]}" | wc -m) -gt 3 ]]; then
        echo -n "$ip got timedout for 10 minutes " >> $watchdogfile
        echo "on `date`" >> $watchdogfile
        sudo -u root ipset add g8keep3r $ip timeout 600
    fi
done
```

This part is where the program retrieves IP on the base of the username then enforces the rule to block the username based on IP.



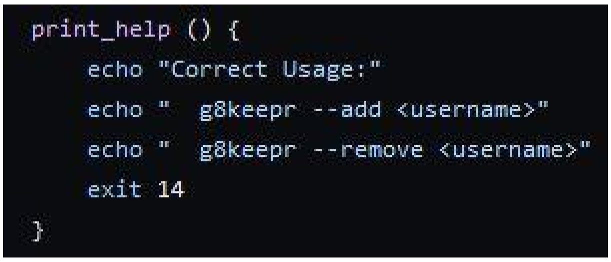
```
rm -rf $tmpfile
```

This part is to remove the temporary file created by this program.

g8keep3r.sh

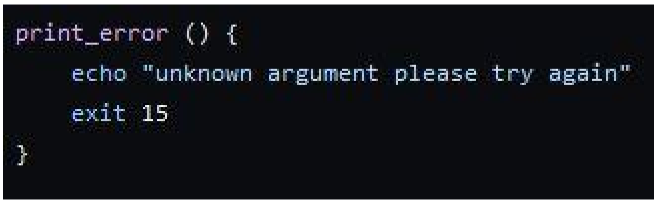
The purpose of g8keep3r.sh is access to var/log/auth.log and retrieve username who enter the wrong password and add it to the watchlist as possible malicious IP.

This is the design of g8keep3r.sh.



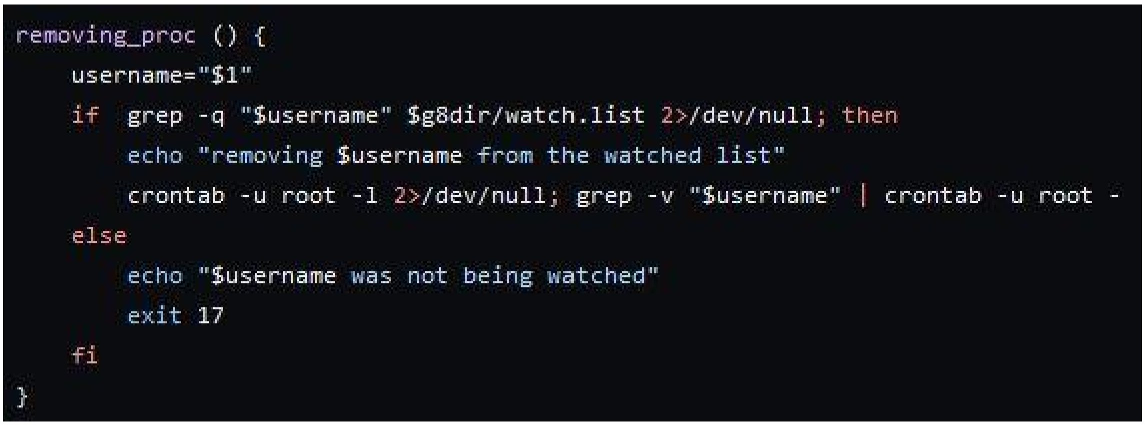
```
print_help () {
    echo "Correct Usage:"
    echo "  g8keepr --add <username>"
    echo "  g8keepr --remove <username>"
    exit 14
}
```

Print lines for humans to read the instruction about adding and removing user-based IP.



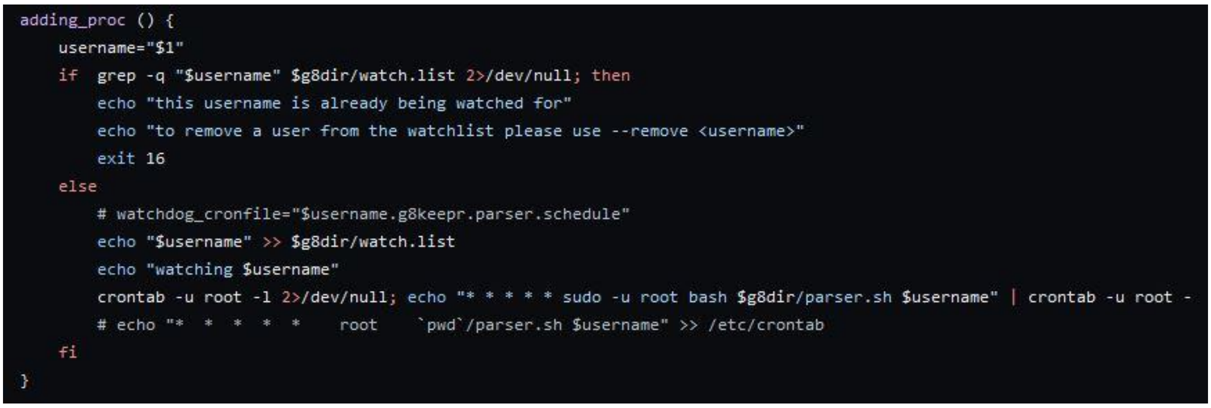
```
print_error () {
    echo "unknown argument please try again"
    exit 15
}
```

If the program does not recognize the argument for user-based IP.



```
removing_proc () {
    username="$1"
    if  grep -q "$username" $g8dir/watch.list 2>/dev/null; then
        echo "removing $username from the watched list"
        crontab -u root -l 2>/dev/null; grep -v "$username" | crontab -u root -
    else
        echo "$username was not being watched"
        exit 17
    fi
}
```

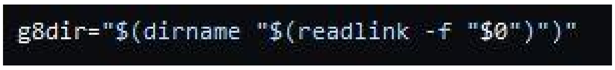
Remove user-based IP from the watchlist file.



```
adding_proc () {
    username="$1"
    if  grep -q "$username" $g8dir/watch.list 2>/dev/null; then
        echo "this username is already being watched for"
        echo "to remove a user from the watchlist please use --remove <username>"
        exit 16
    else
        # watchdog_cronfile="$username.g8keepr.parser.schedule"
        echo "$username" >> $g8dir/watch.list
        echo "watching $username"
        crontab -u root -l 2>/dev/null; echo "* * * * * sudo -u root bash $g8dir/parser.sh $username" | crontab -u root -
        # echo "*  *  *  *  *    root    `pwd`/parser.sh $username" >> /etc/crontab
    fi
}
```

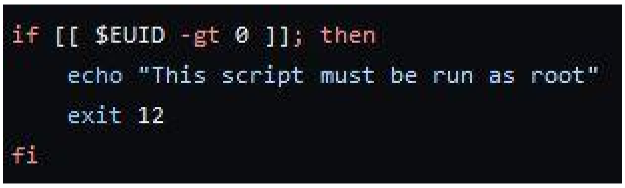
Add user-based IP to the watchlist file. Prevent users to add duplicated-user-based IP if it is already on the list.

In the Else section - if user-based IP is not on the list, then this program will access the parser.sh to find the timestamp and value of flagged user-based IP.



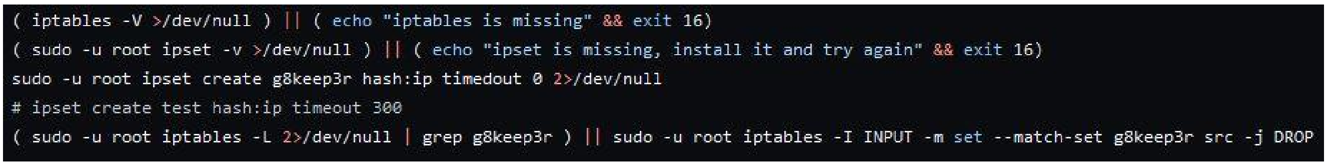
```
g8dir="$(dirname "$(readlink -f "$0")")"
```

Setting a home directory of the script is in.



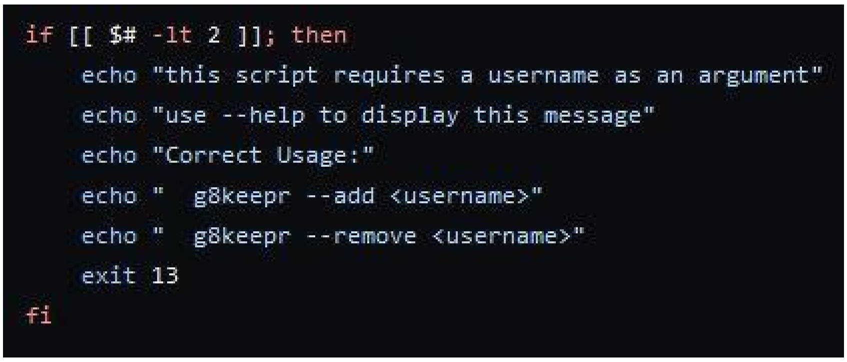
```
if [[ $EUID -gt 0 ]]; then
    echo "This script must be run as root"
    exit 12
fi
```

Check to make sure the root user is authorized to use this program.



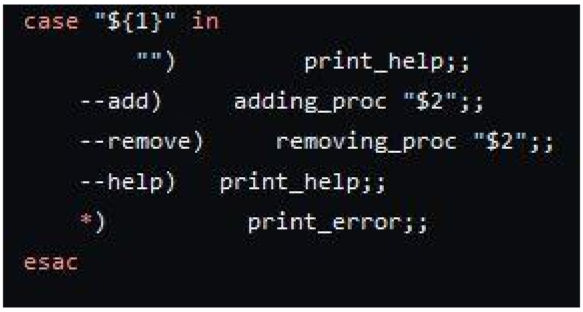
```
( iptables -V >/dev/null ) || ( echo "iptables is missing" && exit 16)
( sudo -u root ipset -v >/dev/null ) || ( echo "ipset is missing, install it and try again" && exit 16)
sudo -u root ipset create g8keep3r hash:ip timedout 0 2>/dev/null
# ipset create test hash:ip timeout 300
( sudo -u root iptables -L 2>/dev/null | grep g8keep3r ) || sudo -u root iptables -I INPUT -m set --match-set g8keep3r src -j DROP
```

Check to see if the user has iptable running in the background. If not, then provide output “iptable is missing”. Also, create a timeout to set up block IP address based on match with a username.



```
if [[ $# -lt 2 ]]; then
    echo "this script requires a username as an argument"
    echo "use --help to display this message"
    echo "Correct Usage:"
    echo "  g8keepr --add <username>"
    echo "  g8keepr --remove <username>"
    exit 13
fi
```

Provide instruction for a human to read if they entered invalid input.



```
case "${1}" in
        "")         print_help;;
    --add)     adding_proc "$2";;
    --remove)     removing_proc "$2";;
    --help)   print_help;;
    *)          print_error;;
esac
```

Set the direction for the program to make calls based on input from the root user.