Challenge 5: Log Mysteries (intermediate)

(provided by Raffael Marty from the Bay Area Chapter, Anton Chuvakin from the Hawaiian Chapter, Sebastien Tricaud from the French Chapter) takes you into the world of virtual systems and confusing log data. In this challenge, figure out what happened to a virtual server using all the logs from a possibly compromised server.

The questions are a more open ended than past challenges. To score highly, we recommend to answer the following way:

- Accuracy is highly encouraged to get the highest note
- You must explain tools you used and how
- If you use visualization tools such as afterglow, picviz, graphviz, gnuplot etc. explain why this was better (than other tools, than other visualization): such as good timeline representation etc.
- Outline HOW you found things

Submission Template

Submit your solution at http://www.honeynet.org/challenge2010/ by 17:00 EST, Thursday, September 30th 2010. Results will be released on Thursday, October 21st 2010.

Name (required): David Bernal Michelena	Email (required): dbernal@seguridad.unam.mx
Country (optional): MEXICO	Profession (optional): _ Student _X Security Professional _ Other

1. Was the system compromised and when? How do you know that for sure?	Possible Points: 5
Tools Used: shell programming, awk, logwatch Awarded Points:	

Answer
Yes, it was compromised several times, from several IP addresses.
Attacking IP address -- time of compromise

IP address 61.168.227.12 made a brute force attack and breaked into the system 1 time(s)
IP address 122.226.202.12 made a brute force attack and breaked into the system 2 time(s)
IP address 219.150.161.20 made a brute force attack and breaked into the system 4 time(s)
IP address 222.66.204.246 made a brute force attack and breaked into the system 1 time(s)
IP address 222.169.224.197 made a brute force attack and breaked into the system 1 time(s)
IP address 121.11.66.70 made a brute force attack and breaked into the system 2 time(s)

First, failed password logs, do not just come from an attackers. It is also possible that a authorized had forgotten his/her password or that a user had forgotten the IP address of the secure shell server. To distinguish between these and avoid generating too many false positives, the following criteria has been taken into account:

If there are accepted logins before valid user failed password logs for an IP address, it is less likely to be an attacker, so it will need at least N failed password logins to be considered as hostile.
If there are failed password logs for a valid user before accepted password logs, X failed password login attemps for a valid user, and Y failed password for an invalid user are needed for an IP address to be considered as an attacker.
If there are at least Z invalid failed password or B valid failed password login attemps, an IP address is considered to be an attacker, it does not matter if there are accepted password logs before these logs.

These values will be defined by every system administrator, based on their specific needs. Hence, this numbers should be modifiable to met their requirements. For this challenge I used the following values:

N=30
X=5
Y=3
Z=20
B=30

Methodology
I developed a hash of hashes in perl to store data of every IP address loged in auth.log file. Then I used the criteria above to classify into successful attacker, failed attacker and not attacker IP addresses, storing the following data for every attacker IP IP address, number of successful logins attemps, number of valid user and invalid user login attemps, time of successful logins, time

With all the data stored, it was possible to answer all the questions of the challenge.

2. If the was compromised, what was the method used?	Possible Points: 5
Tools Used: Awarded Points:	
Answer Bruteforce attack on secure shell service on root user.	

3. Can you locate how many attackers failed? If some succeeded, how many were they? How many stopped attacking after the first success? Yes, the attackers who failed are the following: (the first attacked more times) 8.12.45.242 124.207.117.9 211.154.254.248 217.15.55.133 65.208.122.48 58.17.30.49 116.6.19.70 210.68.70.170 24.192.113.91 124.51.108.68 173.9.147.165 209.59.222.166 125.235.4.130 201.64.234.2 114.80.166.219 203.81.226.86 59.46.39.148 122.102.64.54 219.139.243.236 200.72.254.54 220.170.79.247 61.151.246.140 190.4.21.190 218.56.61.114 89.46.213.128 122.165.9.200 24.94.90.96 The count number attacks for every of these IP addresses is show in the failed.gif gnuplot graph provided in the Graphs directory	Possible Points: 5
Tools Used: Awarded Points:	

Answer

Attackers who succeded:

IP address 61.168.227.12 made a brute force attack and breaked into the system 1 time(s)
IP address 122.226.202.12 made a brute force attack and breaked into the system 2 time(s)
IP address 219.150.161.20 made a brute force attack and breaked into the system 4 time(s)
IP address 222.66.204.246 made a brute force attack and breaked into the system 1 time(s)
IP address 222.169.224.197 made a brute force attack and breaked into the system 1 time(s)
IP address 121.11.66.70 made a brute force attack and breaked into the system 2 time(s)

Command used:
perl sshAnalysis.pl auth.log | grep "and breaked into the system"

More detailed information for every IP address is provided below
accepted means the number of times an IP successfully logged into the system.
Accepted<number>: date and time of every successful login.
EndAttack: date and time of the last failed password login
startAttack: date and time of the first failed password login
firstFailed: the first log for the given IP addresses was failed, it was one of the criteria to clasify attacker IP addresses.
User <number>: The user name for every successful login.
:
IP address 61.168.227.12 made a brute force attack and breaked into the system 1 time(s)
accepted --> 1
acceptedDate1 --> Apr 24 15:28:37
endAttack --> Apr 24 15:40:00
failedInvalid --> 20
failedValid --> 193
firstFailed --> 1
startAttack --> Apr 24 15:26:00
user1 --> root

IP address 122.226.202.12 made a brute force attack and breaked into the system 2 time(s)
accepted --> 2
acceptedDate1 --> Apr 23 03:11:03
acceptedDate2 --> Apr 23 03:20:41
endAttack --> Apr 23 03:42:03
failedInvalid --> 185
failedValid --> 328
firstFailed --> 1
startAttack --> Apr 23 03:06:17
user1 --> root
user2 --> root

IP address 219.150.161.20 made a brute force attack and breaked into the system 4 time(s)
accepted --> 4
acceptedDate1 --> Apr 19 05:41:44
acceptedDate2 --> Apr 19 05:42:27
acceptedDate3 --> Apr 19 05:55:20
acceptedDate4 --> Apr 19 05:56:05
endAttack --> Apr 19 08:58:54
failedInvalid --> 7574
failedValid --> 1685
firstFailed --> 1
startAttack --> Apr 19 05:38:01
user1 --> root
user2 --> root
user3 --> root
user4 --> root

IP address 222.66.204.246 made a brute force attack and breaked into the system 1 time(s)
accepted --> 1
acceptedDate1 --> Apr 19 10:45:36
endAttack --> Apr 19 11:24:39
failedInvalid --> 1063
failedValid --> 510

firstFailed --> 1
startAttack --> Apr 19 10:41:41
user1 --> root

IP address 222.169.224.197 made a brute force attack and breaked into the system 1 time(s)
accepted --> 1
acceptedDate1 --> Apr 22 11:02:15
endAttack --> Apr 22 11:21:34
failedInvalid --> 457
failedValid --> 189
firstFailed --> 1
startAttack --> Apr 22 11:01:29
user1 --> root

IP address 121.11.66.70 made a brute force attack and breaked into the system 2 time(s)
accepted --> 2
acceptedDate1 --> Apr 20 06:13:03
acceptedDate2 --> Apr 24 11:36:19
endAttack --> Apr 24 11:41:59
failedInvalid --> 6
failedValid --> 1429
firstFailed --> 1
startAttack --> Apr 20 05:48:07
user1 --> root
user2 --> root

4. What happened after the brute force attack?	Possible Points: 5

Tools Used:
Awarded Points:

Answer

Many programs were replaced, and exim mail server was installed. All this information

Answer
Exim was reconfigured after the system was compromised, maybe this changes were done by the attacker. Many programs were replaced. This information can be find in the apt log

/var/lib/python-support/python2.5/yum/__init__.py:1129: Warning: 'with' will become a reserved keyword in Python 2.6
/var/lib/python-support/python2.5/yum/depsolve.py:73: Warning: 'with' will become a reserved keyword in Python 2.6
/var/lib/python-support/python2.5/yum/repos.py:236: Warning: 'with' will become a reserved keyword in Python 2.6
/var/lib/python-support/python2.5/yum/repos.py:260: Warning: 'with' will become a reserved keyword in Python 2.6
/var/lib/python-support/python2.5/yum/repos.py:263: Warning: 'with' will become a reserved keyword in Python 2.6
/usr/share/yum-cli/cli.py:614: Warning: 'with' will become a reserved keyword in Python 2.6
/usr/share/yum-cli/cli.py:615: Warning: 'with' will become a reserved keyword in Python 2.6
/usr/share/yum-cli/cli.py:616: Warning: 'with' will become a reserved keyword in Python 2.6

Preparing to replace libkrb53 1.6.dfsg.3~beta1-2ubuntu1.3 (using .../libkrb53_1.6.dfsg.3~beta1-2ubuntu1.4_amd64.deb) ...

```
Unpacking replacement libkrb53 ...
Preparing to replace exim4-config 4.69-2 (using .../exim4-config_4.69-2ubuntu0.1_all.deb) ...
Unpacking replacement exim4-config ...
Preparing to replace exim4-base 4.69-2 (using .../exim4-base_4.69-2ubuntu0.1_amd64.deb) ...
Unpacking replacement exim4-base ...
Preparing to replace exim4-daemon-light 4.69-2 (using .../exim4-daemon-light_4.69-2ubuntu0.1_amd64.deb) ...
 * Stopping MTA       #[125G
#[119G[ OK ]
Unpacking replacement exim4-daemon-light ...
Preparing to replace exim4 4.69-2 (using .../exim4_4.69-2ubuntu0.1_all.deb) ...
Unpacking replacement exim4 ...
Preparing to replace fuse-utils 2.7.2-1ubuntu2 (using .../fuse-utils_2.7.2-1ubuntu2.1_amd64.deb) ...
Unpacking replacement fuse-utils ...
Preparing to replace libfuse2 2.7.2-1ubuntu2 (using .../libfuse2_2.7.2-1ubuntu2.1_amd64.deb) ...
Unpacking replacement libfuse2 ...
Preparing to replace libpq5 8.3.9-0ubuntu8.04 (using .../libpq5_8.3.10-0ubuntu8.04_amd64.deb) ...
Unpacking replacement libpq5 ...
Preparing to replace sudo 1.6.9p10-1ubuntu3.5 (using .../sudo_1.6.9p10-1ubuntu3.7_amd64.deb) ...
Unpacking replacement sudo ...
Setting up libkrb53 (1.6.dfsg.3~beta1-2ubuntu1.4) ...

Setting up exim4-config (4.69-2ubuntu0.1) ...

Setting up exim4-base (4.69-2ubuntu0.1) ...
Installing new version of config file /etc/init.d/exim4 ...

Setting up exim4-daemon-light (4.69-2ubuntu0.1) ...
 * Starting MTA       #[125G
#[119G[ OK ]

Setting up exim4 (4.69-2ubuntu0.1) ...

Setting up libfuse2 (2.7.2-1ubuntu2.1) ...

Setting up fuse-utils (2.7.2-1ubuntu2.1) ...
creating fuse group...
update-initramfs: deferring update (trigger activated)

Setting up libpq5 (8.3.10-0ubuntu8.04) ...

Setting up sudo (1.6.9p10-1ubuntu3.7) ...
```

5. Locate the authentication logs, was a bruteforce attack performed? if yes how many?	Possible Points: 5
Tools Used: Awarded Points:	
Answer authentication logs is auth.log Yes, there were several bruteforce attacks performed. There were 11 succesful attacks from 6 different IP addresses. (listed on the answer for question 3	

There were 27 unsuccessful attacks.

6. What is the timeline of significant events? How certain are you of the timing?	Possible Points: 5

Tools Used:
Awarded Points:

Answer

The timeline of significant events can be viewed in the afterglow graph. It shows the timeline for every succesful login of each of the 6 successful IP addresses. Afterglow was used for this because it shows in a very clear way how many successful attacks there were for every attacker, their time and date and the user name attacked.

121.11.66.70
Apr 20 06:13:03
Apr 24 11:36:19

222.66.204.246
Apr 19 10:45:36
Apr 19 10:45:36

219.150.161.20
Apr 19 05:41:44
Apr 19 05:42:27
Apr 19 05:55:20
Apr 19 05:56:05

222.169.224.197
Apr 22 11:02:15

61.168.227.12
Apr 24 15:28:37

122.226.202.12
Apr 23 03:11:03
Apr 23 03:20:41

7. Anything else that looks suspicious in the logs? Any misconfigurations? Other issues?	Possible Points: 5

Tools Used:
Awarded Points:

8. Was an automatic tool used to perform the attack? if yes which one?	Possible Points: 5

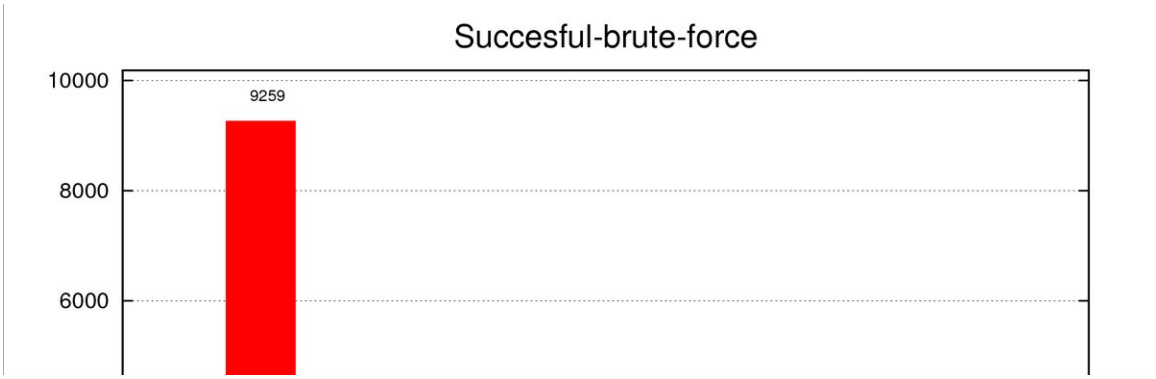
Tools Used:

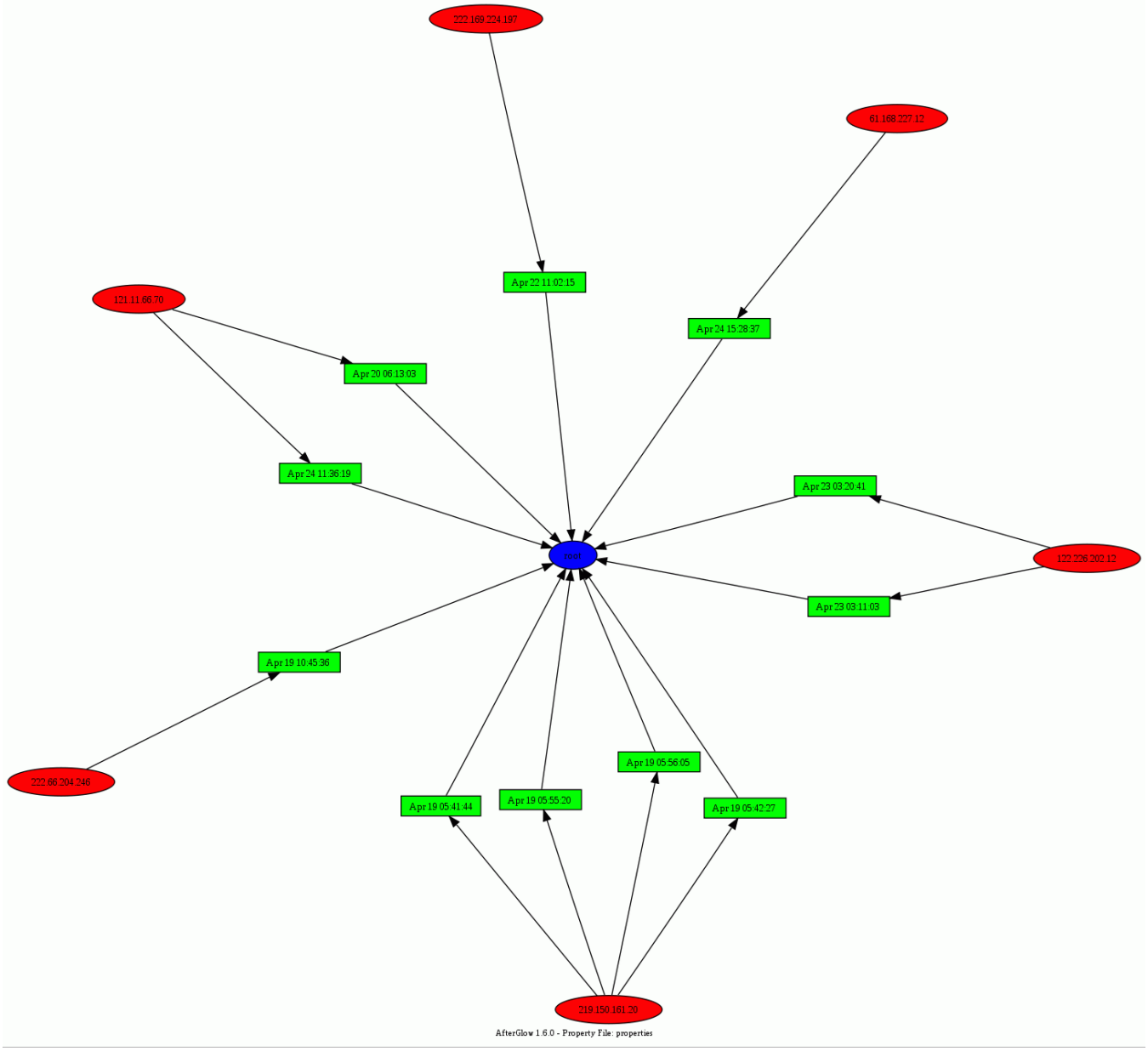
Awarded Points:
Answer Yes, because there are too many logs in a very short time, it is obvious that an automatic tool was used. Maybe a tool like hydra could have been used, but I can not be sure.

9. What can you say about the attacker's goals and methods?	Possible Points: 5
Tools Used: Awarded Points:	
Answer They used brute force attack to log into a remote system using root password, to get administrator privileges and be able to install programs in the remote server.	

Bonus. What would you have done to avoid this attack?	Possible Points: 5
Tools Used: Awarded Points:	
Answer I would have enforced a strong password policy, for example, using cracklib in PAM. I would have installed a host IPS such as fail2ban or any other tool that would analyze logs for auth.log file and then create dynamic firewall rules, to block attacker IP addresses. Disable root remote login. Possibly change secure shell port. I would have restricted traffic only from the IP addresses where I would expect connections to come from. Host based IDS, such as logwatch could also have helped avoiding this incident. Network based IDS, such as snort could also have helped avoiding this incident. Also, policies are important, as wells as telling the unexperienced users they that hardening is important and guiding them trough this proccess.	

Tools used





Requirements:
perl, gnuplot, imagemagick, afterglow

./perl sshAnalysis.pl auth.log

All the information for the attacker IP address will be shown on the standard output. By default, all the information for succesful attacker Ips with be shown in detail
For example:
IP address 61.168.227.12 made a brute force attack and breaked into the system 1 time(s)
  accepted --> 1
  acceptedDate1 --> Apr 24 15:28:37
  endAttack --> Apr 24 15:40:00
  failedInvalid --> 20
  failedValid --> 193
  firstFailed --> 1
  startAttack --> Apr 24 15:26:00
  user1 --> root

If the user wants to see the information for unsuccesful brute force attacks, the line
#print " $valor --> $hash{$cuenta}{$valor} \n" must be uncommented.

input files: auth.log

output files:
It creates graphs using gnuplot for failed and succesful IP addresses, show the attack count numbers.

It also creates a cvs file for the succesful attacker IP addresses, showing the user they used and the date and time for every attack. This file is called ag and can be plotted using
a command like this:

cat ag | perl $afterPath/afterglow.pl -c $afterPath/sample.properties | neato -Tgif -o attackerTimeStamps.gif

---

```
# sshAnalysis.pl
#!/usr/local/bin/perl
%hash=();
print $ARGV;

open (LOG, $ARGV[0]);
open (AE, ">ae");
open (AF, ">af");
open (AFTER, ">ag");

foreach $line (<LOG>) {
        chomp($line);

        if ( $line =~ m/^.*Failed password.*/ ){
                @arr = split(' +',$line);

                # Is the first log for this IP address is an attack log?
```

```
                if ( ! exists $hash{$arr[$#arr-3]}) {
        $hash{$arr[$#arr-3]}{firstFailed} = 1;
    }

    # Gets the date for the failed password log
    $line =~ m/^(.*)app/;

    #If there havent been any valid or invalid user failed password, this is the first attack
    if ( ! exists $hash{$arr[$#arr-3]}{failedInvalid} && ! exists $hash{$arr[$#arr-3]}{failedValid} ) {
          #print "La fecha/hora del primer ataque para $arr[$#arr-3] es $1 \n";
        $hash{$arr[$#arr-3]}{startAttack} = $1;
    }
    #Stores the date for the last attack
    else{
          $hash{$arr[$#arr-3]}{endAttack} = $1;
    }

                if ( $line =~ m/^.*invalid user.*/){
                        # an invalid user was detected
                        #print "Invalid user from $arr[$#arr-3]: $arr[$#arr-5]"."\n";
                        $hash{$arr[$#arr-3]}{failedInvalid}++;
                }else{
                        $hash{$arr[$#arr-3]}{failedValid}++;
                }
          }elsif( $line =~ m/^.*Accepted.*/ ){
            # Gets the date for the failed password log
   $line =~ m/^(.*)app/;
            @arr = split(' +',$line);
                        #print "User $arr[$#arr-5] accepted from $arr[$#arr-3]"."\n";
                        $hash{$arr[$#arr-3]}{accepted}++;
                        $indice=$hash{$arr[$#arr-3]}{accepted};
                        $hash{$arr[$#arr-3]}{acceptedDate."$indice"} = "$1";
                        $hash{$arr[$#arr-3]}{user."$indice"} = $arr[$#arr-5];

                        if ( ! exists $hash{$arr[$#arr-3]}) {
        $hash{$arr[$#arr-3]}{errorPrimero} = 0;
    }
          }
}

foreach $cuenta (keys % hash)                   ## En este primer ciclo obtenemos las llaves del hash principal
{                                               ## que en este caso son las cuentas
          #print "IP: $cuenta  ";                ## Se imprime el valor en cada iteracion para verificar lo
anterior
          $keyscuenta = $hash{$cuenta};         ## Se asigna a $keyscuentas el valor
          #print " keycuenta tiene la direccion de este hash: $keyscuenta\n";

          $attacker=&isAttacker($cuenta);

                if($attacker == 2){
                        print "IP address $cuenta made a brute force attack and breaked into the system
".$hash{$cuenta}{accepted}." time(s)\n";

                        $s = $hash{$cuenta}{failedValid}+$hash{$cuenta}{failedInvalid};
                        print AE $s."    ".$cuenta."\n";
```

```
                        for ($i=1; $i<=$hash{$cuenta}{accepted};$i++){
                                print AFTER
$cuenta.",".$hash{$cuenta}{acceptedDate."$i"}.",".$hash{$cuenta}{user."$i"}."\n";

                        }

                        foreach $valor (sort keys %{ $hash{$cuenta}})
        {
                print "   $valor --> $hash{$cuenta}{$valor} \n";
         }
         print "\n";
                }elsif( $attacker == 1){
                        #print "IP address $cuenta made a brute force attack, but did not breaked into the system\n";

                        $s = $hash{$cuenta}{failedValid}+$hash{$cuenta}{failedInvalid};
                 print AF  $s."   ".$cuenta."\n";
                        foreach $valor (sort keys %{ $hash{$cuenta}})
        {
                 #print "   $valor --> $hash{$cuenta}{$valor} \n";
         }

         print "\n";

                }else{
                        #print "IP address $cuenta does not meet the criteria to be considered an attacker \n\n";
                }
}

close(LOG);
close(AE);
close(AF);

`sort -nrk1 ae > succesfulAttacks`;
`sort -nrk1 af > failedAttacks`;
`./graph.sh failedAttacks Failed-brute-force`;
`./graph.sh succesfulAttacks Succesful-brute-force`;
# change
# generate afterglow graph with ag cvs file, for example: `cat ag | perl $afterPath/afterglow.pl -c $afterPath/sample.properties
| neato -Tgif -o attackerTimeStamps.gif`;


sub isAttacker
{
        if ( $hash{$_[0]}{firstFailed} == 1 && ( $hash{$_[0]}{failedInvalid} >= 3 || $hash{$_[0]}{failedValid} >= 5 ) ){
                        if(  $hash{$_[0]}{accepted} >= 1){
                                return 2;
                        }else{
                                return 1;
                        }
        }elsif ( $hash{$_[0]}{failedInvalid} >= 20 || $hash{$_[0]}{failedValid} >= 30 ){
         if(  $hash{$_[0]}{accepted} == 1){
                                return 2;
                }else{
                                return 1;
```

```
                }
        }else{
                return 0;
        }
}
```