

# MAC TIPS AND "HOW TO" ON A MAC

AFTER 20+ YEARS ON WINDOWS AND PC, IT TOOK ME 2 DAYS ON A  
MAC TO REALIZE THAT I WASTED 20 YEARS OF MY LIFE ON  
WINDOWS. BUMMER.

## DONATE

PROMOTE FREE  
AND RECEIVE A  
DICTIONARY AT



THE HIGHEST FORM OF IGNORANCE IS WHEN  
YOU REJECT SOMETHING YOU DON'T KNOW  
ANYTHING ABOUT.

Wayne Dyer (b 1940)

## How To Install Pyrit CUDA on Mac

<3

<3 th!2 1st.

USB Adapter  
Who Is The  
The Good, The  
Unbiased Review

Best Deals and



## Pyrit CUDA: *Release the Kraken!*

### How To Install Pyrit CUDA on a Mac



We hope that you have not landed here randomly, Pyrit CUDA is not  
for the faint of heart, But your patience will be rewarded. Highly  
rewarded.

CUDA stands for Compute Unified Device Architecture. It uses or  
unleash the power of your GPU(s) to compute a bit faster things like  
.... WPA key Recovery.

Pyrit CUDA is not a magic bullet, it's just a much, much bigger  
caliber.

Why Pyrit CUDA?

On my "Old MBP", Pyrit CUDA is 45% faster than Aircrack-ng without a  
sweat, If you have a "New Mac" with a much faster Graphic Card,

## BLOG ARCHIVE

▼ [2011](#) (18)

▼ [July](#) (3)

[KisMAC and OS X  
Lion 10.7, The  
Solution](#)

## How To Install Pyrit CUDA on Mac

The Kraken has  
been released!

- ▶ [June](#) (14)
- ▶ [January](#) (1)
- ▶ [2010](#) (11)
- ▶ [2009](#) (31)
- ▶ [2007](#) (1)

---

## FOLLOW BY EMAIL

---

you can expect 50-200% faster.


45% means than instead of running for 10hrs, you'll do the job in 5.5hrs, Some monsters claim 89,000 PMKs/S. With a little tweak, you can go 300% faster, see example under "tips"

I went to Pyrit after finagling wayyyyyy to much with Aircrack-CUDA. Using Backtrack5 on VMware, Aircrack-CUDA was the straw that broke the camel back: The time needed for the install and fixing the issues was longer than trying Aircrack the regular way. So, here is Pyrit, native on a Mac.

Pyrit also allows you to create database of pre-computed PMK, also known as Rainbow Tables, and here, it starts to go really fast ...

How To Install Pyrit CUDA on a Mac, OSX 10.6.8

For OSX 10.7 Lion, almost the same, but read the help first

- 1) Click  > About this Mac > More Info > Graphics/Display to check your Card model #
  - 1a) Verify that you have a [CUDA supported graphic card](#), if not , we're sorry to see you go...
- 2) Follow the steps exactly in the order they are mentioned.
- 3) You need to have Admin Rights or the Admin Password
- 4) You should be Terminal Savvy. If not read the tips at the end first
- 5) You need to type the commands *verbatim*. *A space too much and you're out ....*
- 6) You can click on the pictures to enlarge them. It could help....
- 7) There is a list of warnings at the end, read them!
- 8) **Verify that you have verified the verifications**

The full install takes about 30-45 min.

Ready?

Download, in a easily accessible folder the following , **do NOT install yet**

**Nvidia CUDA drivers for Mac:**

<http://www.nvidia.com/object/mac-driver-archive.html>

**CUDA SDK (requires OSX ver. 10.6.7 or higher)**

<http://developer.nvidia.com/cuda-toolkit-40>

## Instant MacOS CleanUp



[Clean your MacOS](#)

## MacOS performance Boost



### RETWEET THIS BLOG

0  
uptweets  
uptweet

### LOOKING FOR SOMETHING?

Select "CUDA Toolkit" under Mac OS  
[For Older Version \(10.6.6 and under\)](#)

#### NOTE:

If you have an "old" Nvidia card, try the "older version" first, you'll save a lot a space.

the "old version" is half the size of the new one, and you can always upgrade later

#### Libnet

<http://libdnet.googlecode.com/files/libdnet-1.12.tgz>






#### pylibpcap

<http://dfn.dl.sourceforge.net/sourceforge/pylibpcap/pylibpcap-0.6.2.tar.gz>

#### Scapy

<http://www.secdev.org/projects/scapy/files/scapy-latest.tar.gz>

You should now, have something looking like that:

Name
 cudatoolkit_4.0.17_macos.pkg
 cudadriver-4.0.19-macos.dmg
 scapy-latest.tar.gz
 pylibpcap-0.6.2.tar.gz
 libdnet-1.12.tgz

#### Install Nvidia Driver for Mac

Click on the DMG, etc ..

This install is going to take few minutes..

While installing....

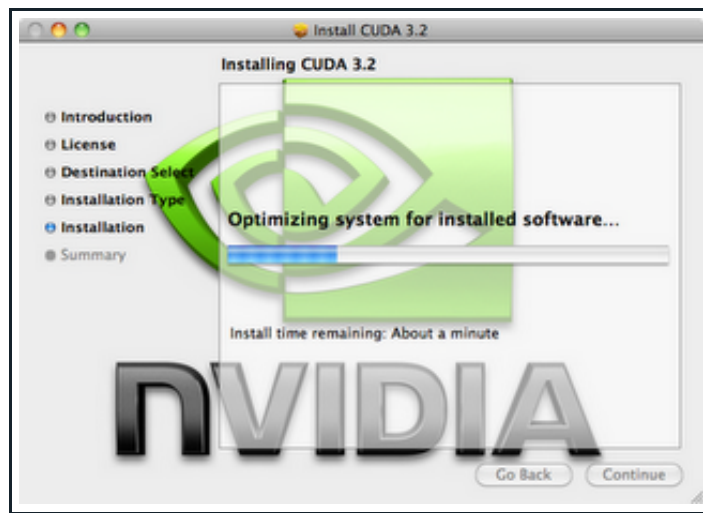
Take few seconds to click the Like button on our FB page

Thanks

<3 142 ppl <3 7h!5. B341 UR n00b5.

#### Install Nvidia CUDA TOOLKIT for Mac

Click on the DMG, etc



Open Terminal and start the installation of libnet

"Path to" refers to the path to the file. i.e

/Users/MyName/Downloads/

Example:



```
tar -xzf /Users/MyName/Downloads/libdnet-1.12.tgz
```

Install

```
tar -xzf "Path to" libdnet-1.12.tgz
cd libdnet-1.12
./configure
make
sudo make install
cd python
sudo python setup.py install
```

Install Nvidia Driver for Mac, C

## COOL FREE APPLICATIONS FOR MAC

All Listed below are Free unless stated otherwise (You may want to consider a little donation to the author(s))

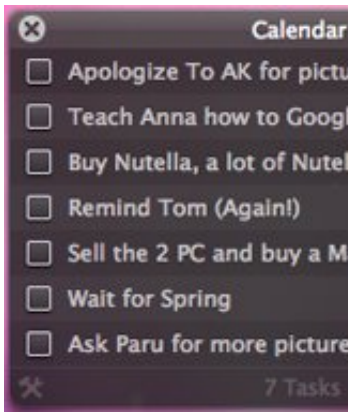
Now we install Pylibpcap

```
tar -xzf "Path to" pylibpcap-0.6.2.tar.gz
cd pylibpcap-0.6.2
sudo python setup.py install
```

Scapy Install

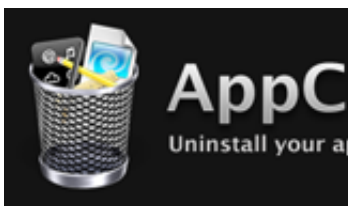
```
tar -xzf "Path to" scapy.tar.gz
cd scapy
sudo python setup.py install
```

## ANXIETY



Superlight interface to check all your tasks

## APP CLEANER.



Clean thoroughly & uninstall easily

## HPC in the Cloud [www.P](http://www.P)

Scalable HPC Services on Xeon Processors

## Clean Up Your Mac OS

Clean Mac is Happy and CleanMyMac Now.

## MSN Internet Explorer@

The Most Advanced & Fastest Web at It's Fullest.

## Password Recovery - \$9

Find your old password in Recovery. Save \$9.95.

## BUNCH O' FREE APP

Now the prerequisites are done, we can go in the heart of the subject.

From the Terminal Window, Download Pyrit

```
svn checkout http://pyrit.googlecode.com/svn/trunk/pyrit-read-only
```

Build and install Pyrit

```
cd pyrit-read-only
cd pyrit
sudo python setup.py install
```

Last step, Pyrit CUDA

```
cd pyrit-read-only
cd cpyrit_cuda
sudo LDFLAGS=-L/usr/local/cuda/lib python setup.py install
```

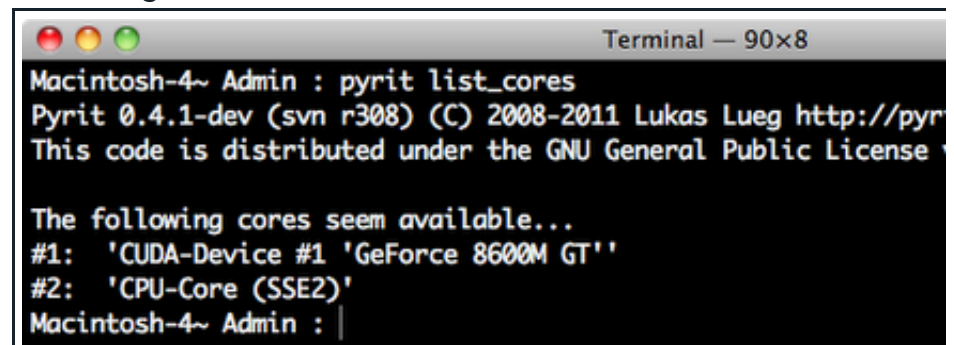
Now we need to check if all of that was worth it

Check if Pyrit CUDA is working

```
pyrit list_cores
```

You should see a list of your cores and a list of your GPU(s)

Something like



Once done, let's Benchmark it and see if we can *Release The*



## Kraken



pyrit benchmark

```
Terminal — 90x10
Macintosh-4~ Admin : pyrit benchmark
Pyrit 0.4.1-dev (svn r308) (C) 2008-2011 Lukas Lueg http://pyrit.org
This code is distributed under the GNU General Public License

Running benchmark (1883.2 PMKs/s)... \

Computed 1883.18 PMKs/s total.
#1: 'CUDA-Device #1 'GeForce 8600M GT': 1295.8 PMKs/s (RTT 2.5)
#2: 'CPU-Core (SSE2)': 694.2 PMKs/s (RTT 2.9)
Macintosh-4~ Admin :
```

## CAMOUFLAGE



Camouflage Hides your  
Icons in a Click

\*I have installed Pyrit on an old machine, the "good ones" are  
"reserved" for work. :-)

Leave a comment either here or on FB with your Config and the  
Benchmark Results.

Who has a monster?

## GTK, FAQ, RFAQ, TIPS, Help

### GTK! AKA "Good To Know"

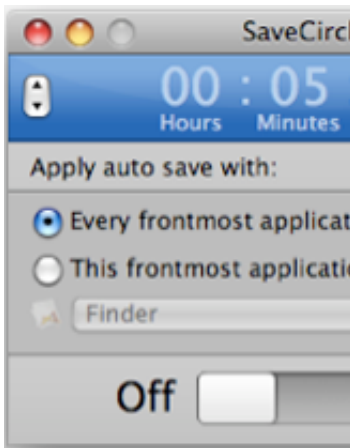
Bruteforcing is time consuming, so you need to go as fast as possible, and also [AS SMART AS POSSIBLE](#): Bruteforcing in blind mode, aka starting @ "00000000" and going all the way up, trying each alphanumerical combination is just a pure waste of your (limited) time, you silly mortal, and your electricity bill will go up a bit. Go smart, use a [statistically sorted Attack Dictionary](#): Most used password first: If the password to discover is "password" running an incremental attack will take you few months: The first set of numbers is 100,000,000 long, then for each set of letters, add 110,075,314,176. You have 24 sets, hence 2,641,807,540,224 passwords. Then repeat with upper cap... then mingle all of that ... a0a0a0a0

Your brain has now stopped perceiving the true value of those numbers. Mine too.

But that's not it!

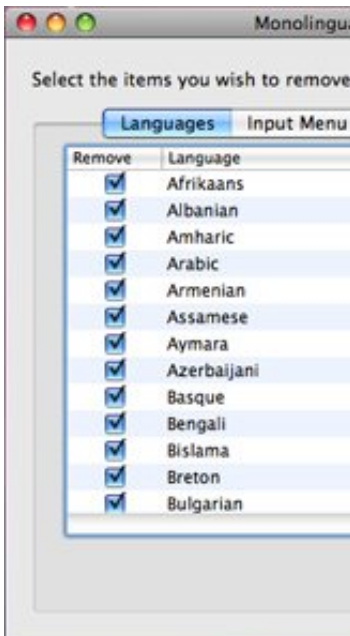
Mr. WPA is a tedious little man: Each password has 4096 round of hash, salted with the BSSID. It means in clear that your CPU/GPU

## SAVECIRCLE



Autosave Frontmost App.  
A Must Have

## MONOLINGUAL

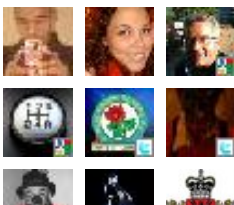


Remove unused  
languages and free up to  
5 Gb of space

## FOLLOWERS

Join this site  
with Google Friend  
Connect

Members (16)



will crunch about 1 Megabytes (byte, not bit) of data for each PMK (Pairwise Master Key) , Each Password is "about" 4 PMK 2500 PMK/second and you'll have 2.44 Gigabytes of data being pushed through per second...

The first set of numbers, 8ch long, will boil down to about a billion megabytes of data being crunched.

So, Act like a ninja, and think like Einstein: [Go smart!](#)

## Tips

If you are here, you should know it: Don't play too much with the sudo command: it's an unforgiving command. sudo does not give any warning, sudo is "Das Terminator"

### Long paths

Avoid typing long path with spaces or weird names:

- 1) Download in a easily accessible folder
- 2) Type your command, i.e "tar -xzf" then drag the file *from* Finder *to* Terminal; the path will automatically appear correctly.
- 3) Use the arrow up to call back a previous command
- 4) copy and paste the command instead of re-typing them:

"cd cpyrit" just looks like "cd pyrit" ( "C"pyrit )

(The first one that will comment about cd not working will get seriously flamed)

## CUDA use

When using Pyrit CUDA, quit all applications, including Anti-virus, Browser etc, I mean everything: Dropbox, Airport, etc. Anything that use a single %. Kill all processes but the vital ones: The performances will improve significantly.

For the best results, don't even use a screen saver: quit everything and let the screen go black. Remember? CUDA is using your GPUs.

Following those tips you'll see the performance increasing significantly:

**3906 PMKs per second.**

We are now 279% (Two-hundred-seventy-nine ) faster than Aircrack-ng 1.1 and 600% faster than KisMAC 0.3.3 . Yes, 600%. But, that's not it! Can you go faster than that?

### Temperature:

If you crunch for hours, don't forget that Pyrit CUDA will try to use 100% of your CPUs and 100% of your GPUs: The Temperature is going

---

to go up a bit. If you have a laptop, I'll suggest to elevate it on the four corners and leave at least 1/2 inch of free space under.

## FAQ, RFAQ, SFQRFA

- My card is not supported!

Bummer!

- How do you crack a WEP on Pyrit?

You don't! Pyrit is WPA only.

- kan't crack the pazwords!

[Probable Cause](#)

- it doz not workz!

[Probable Cause](#)

- It's not working on Windows

[Did you pass the test? \(successfully?\)](#)

- I want to crack my girlfriend password

[Talk to those guys](#)

- I overclocked my GPU and my computer shut down

Told ya! Try not to go over 200F / 93C. If you insist, you may be able to fry an egg on your Mac, please send us the picture. For a "runny-sunny side up" you can start at 66C.

- Please help

It's just below. on more line...

## Pyrit help

For more help type "pyrit -h [command]" i.e "pyrit -h attack\_passtrough"

Highly recommended reading:

<http://pyrit.wordpress.com/>

About Pyrit

<http://pyrit.wordpress.com/about/>

## Pyrit on OSX Lion 10.7

"Many people have [problems](#) compiling Pyrit on OSX Lion. The version of GCC distributed with the latest XCode no longer supports creating binary code for the PPC-architecture and Python's *setup.py* does not know about that; you can get an error message like the following:"

```
assembler (/usr/bin/../../libexec/gcc/darwin/ppc/as or
/usr/bin/../../local/libexec/gcc/darwin/ppc/as) for architecture ppc
not installed
```



You can solve this situation by forcing GCC to only compile code for the i386- and the x86\_64-architecture. To do this, put the following into your *.bash\_profile*:

```
export ARCHFLAGS="-arch i386 -arch x86_64"
```

Source: <http://pyrit.wordpress.com/>

Retrieved Aug 2, 2011

Pyrit 0.4.1-dev (svn r308) (C) 2008-2011 Lukas Lueg

<http://pyrit.googlecode.com>

This code is distributed under the GNU General Public License v3+

Usage: pyrit [options] command

Recognized options:

- b : Filters AccessPoint by BSSID
- e : Filters AccessPoint by ESSID
- h : **Print help for a certain command**
- i : Filename for input ('-' is stdin)
- o : Filename for output ('-' is stdout)
- r : Packet capture source in pcap-format
- u : URL of the storage-system to use
- all-handshakes : Use all handshakes instead of the best one

Recognized commands:

- analyze : Analyze a packet-capture file
- attack\_batch : Attack a handshake with PMKs/passwords from the db
- attack\_cowpatty : Attack a handshake with PMKs from a cowpatty-file
- attack\_db : Attack a handshake with PMKs from the db
- attack\_passthrough : **Attack a handshake with passwords from a file**
- batch : Batchprocess the database
- benchmark : **Determine performance of available cores**
- benchmark\_long : **Longer and more accurate version of benchmark (~10 minutes)**
- check\_db : Check the database for errors
- create\_essid : Create a new ESSID
- delete\_essid : Delete a ESSID from the database
- eval : Count the available passwords and matching results

export\_cowpatty : Export results to a new cowpatty file  
export\_hashdb : Export results to an airolib database  
export\_passwords : Export passwords to a file  
help : Print general help  
import\_passwords : Import passwords from a file-like source  
import\_unique\_passwords : Import unique passwords from a file-like source  
list\_cores : List available cores  
list\_essids : List all ESSIDs but don't count matching results  
passthrough : Compute PMKs and write results to a file  
relay : Relay a storage-url via RPC  
selftest : Test hardware to ensure it computes correct results  
serve : Serve local hardware to other Pyrit clients  
strip : Strip packet-capture files to the relevant packets  
stripLive : Capture relevant packets from a live capture-source  
verify : Verify 10% of the results by recomputation

## More help

<http://code.google.com/p/pyrit/>

## Basic Command lines

```
pyrit -h attack_passthrough
```

The -h option gives a more detailed help on an option, here help on "Attack\_Passthrough" -h should be used profusely.

```
pyrit benchmark
```

Does a Quick Benchmark

```
pyrit benchmark_long
```

Does a long Benchmark

```
pyrit -r test.pcap -b 00:de:ad:be:ef:00 -i words  
attack_passthrough
```

Regular attack on a specific ESSID via Dictionary

---

```
pyrit -r test.pcap -b 00:de:ad:c0:de:00 -o passwd.txt  
attack_batch
```

"Pairwise Master Keys that have been computed and stored in the database previously are taken from there; all other passwords are translated into their respective Pairwise Master Keys and added to the database for later re-use. ESSIDs are created automatically in the database if necessary."

Note: .PCAP, .CAP or Dumplogs are the same

## Overclocking

You can overclock, but:

As much as you will be tempted, let me remind you that if you have a laptop, things may get hot. Really hot! Even SMCFan Control may not be enough.

Frying your GPU will not be a good thing. Overclock at your own risk(s)

little hidden gem:

because you've read so far, you deserve a little bonus:

Bookmark us, follow that blog, and we'll provide you with a very good surprise.

Comments:

[Please read this before commenting](#)

.

1

tweet

. retweet



LABELS: [GPU ASSISTED CRACKING](#), [PYRIT CUDA](#)

---

## 1 COMMENTS:

Anonymous said...

You might want to tell people to use gfxCardStatus on MBPro's as the CUDA support dies as soon as the laptop switches back to the intel video card.

Here is the difference on my system (MacBookPro6,2 Intel Core i7)  
WITH Safari and other apps running in the background.

```
sh-3.2# sudo pyrit benchmark
```

Pyrit 0.4.1-dev (svn r308) (C) 2008-2011 Lukas Lueg

<http://pyrit.googlecode.com>

This code is distributed under the GNU General Public License v3+

Running benchmark (1420.3 PMKs/s)... |

Computed 1420.29 PMKs/s total.

#1: 'CPU-Core (SSE2)': 378.7 PMKs/s (RTT 3.0)

#2: 'CPU-Core (SSE2)': 381.5 PMKs/s (RTT 3.1)

#3: 'CPU-Core (SSE2)': 382.1 PMKs/s (RTT 3.1)

#4: 'CPU-Core (SSE2)': 377.6 PMKs/s (RTT 3.0)

```
sh-3.2# sudo pyrit list_cores
```

Pyrit 0.4.1-dev (svn r308) (C) 2008-2011 Lukas Lueg

<http://pyrit.googlecode.com>

This code is distributed under the GNU General Public License v3+

The following cores seem available...

#1: 'CUDA-Device #1 'GeForce GT 330M'

#2: 'CPU-Core (SSE2)'

#3: 'CPU-Core (SSE2)'

#4: 'CPU-Core (SSE2)'

```
sh-3.2# sudo pyrit benchmark
```

Pyrit 0.4.1-dev (svn r308) (C) 2008-2011 Lukas Lueg

<http://pyrit.googlecode.com>

This code is distributed under the GNU General Public License v3+

Running benchmark (2897.8 PMKs/s)... -

Computed 2897.80 PMKs/s total.

#1: 'CUDA-Device #1 'GeForce GT 330M': 2005.3 PMKs/s (RTT 2.9)

#2: 'CPU-Core (SSE2)': 383.5 PMKs/s (RTT 3.1)

#3: 'CPU-Core (SSE2)': 382.4 PMKs/s (RTT 3.1)

#4: 'CPU-Core (SSE2)': 381.4 PMKs/s (RTT 3.0)

AUGUST 12, 2011 2:14 PM

POST A COMMENT

Comment as: 

Select profile... ▾

Post Comment

Preview

## LINKS TO THIS POST

[Create a Link](#)

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)