

# Processing iPhones

Richard Gilleland  
Sacramento Police Department  
rgilleland@pd.cityofsacramento.org

This document describes the Jonathan Zdziarski method for processing iPhones.

Jonathan Zdziarski has designed a number of tools (along with great documentation) that can be used to both remove the passcode from an iPhone as well as to image an iPhone. Zdziarski offers his tools and documentation free to law enforcement through his website which is located at 'www.iphoneinsecurity.com/'.

The screenshot shows the top portion of the 'iPhone InSecurity' website. The header is dark blue with 'IPHONE INSECURITY' in white. To the right of the header are links for 'Access' and 'Contact'. Below the header, there is a welcome message: 'Welcome to the source for law enforcement tools and documentation in iPhone Forensic research. Home to the only complete suite of forensic tools for the iPhone, iPhone 3G, and iPhone 3G[s].'. A red link for 'Site Access and Tools Licensing' is present. Below this, a paragraph states that site access is freely available to law enforcement personnel with specific powers. A list of disallowed users follows, including commercial entities, contractors, consultants, private investigators, and part-time personnel. At the bottom of the screenshot, a red link states 'ALL WEBSITE ACCESS IS LOGGED AND AUDITED.'. An arrow points from the 'Access' link in the header to the text 'Once an account has been established...' in the main document.

Once an account has been established at [iphonesecurity.com](http://iphonesecurity.com), users have access to the tools and documentation that Zdziarski has created. Access to these tools is necessary to process iPhones using this method. I highly recommend reading 'iPhone Forensic Investigative Methods.pdf' by Zdziarski for a comprehensive description of processing iPhones.

This document is not meant to take the place of Zdziarski's comprehensive publication, it is simply meant to provide a short / detailed description for processing iPhones.

Prior to processing an iPhone, its firmware version must first be established. An iPhone's firmware can be determined in both a Windows environment and a Mac environment. The following steps can be used to determine the phones firmware version in a Windows environment. Go to page 6 of this document for instructions in a Mac environment.

Page 2	Determining iPhone Firmware Version - Windows
Page 6	Determining iPhone Firmware Version - Mac
Page 10	Removing the iPhone's Pass Code
Page 15	Imaging the iPhone
Page 23	Working with the Image File
Pages 27 / 28	Cheat Sheets

## Determining iPhone Firmware Version

### Firmware determined using Windows XP OS;

System requirements;

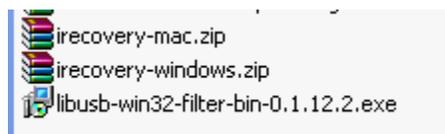
Windows XP  
iTunes (I used version 9.1.1.12 for this test)  
Internet access

(\* This method may not work for Vista and Windows 7 systems)

1. Note the model number located on the back of the iPhone.

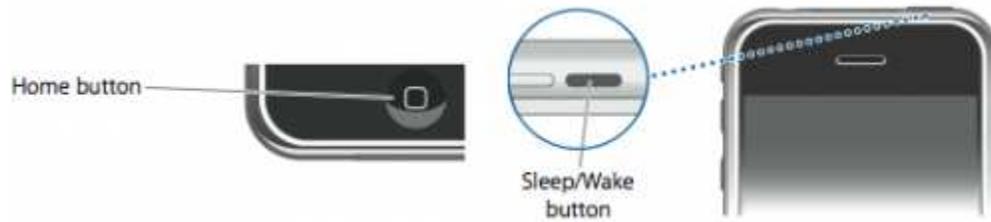


2. Download and unzip 'iHackintosh iRecovery Package for Windows & Mac.rar'  
Can be located here;  
<http://www.ihackintosh.com/2009/07/irecovery-iphone-recovery-mode-loop-restart/>  
The following programs should be included in the .rar file – (unzip 'irecovery-windows.zip');



3. Install 'libusb-win32-filter-bin-0.1.12.2.exe'
4. Re-start computer
5. Connect the iPhone to your Windows machine and place phone into **Recovery** mode (not DFU mode);

Press and hold the Home button and the Sleep/Wake button at the same time.



After exactly 10 seconds release the Sleep/Wake button. Continue holding the home button until your iTunes pops up a message telling you that it has detected an iPhone in recovery mode.

The phone should display the below listed screen.



6. Open a cmd prompt and navigate to directory containing iRecovery.exe file.

Type `iRecovery.exe -s`

Review the results and note the numbers listed after 'iBoot'.

```
C:\WINDOWS\SYSTEM32\CMD.EXE - iRecovery.exe -s
C:\Documents and Settings\Instructor01\Desktop\Libusb>iRecovery.exe -s
iRecovery - Recovery Utility
by wEsTbAeR-- and Tom3q

Got USB

=====
::
:: iBoot for m68ap, Copyright 2009, Apple Inc.
::
::   BUILD_TAG: iBoot-636.65
::
::   BUILD_STYLE: RELEASE
::
::   USB_SERIAL_NUMBER: CPID:8900 CPRU:20 CPFM:03 SCEP:05 BDID:00 ECID:000002
D2A404DDF6 IBFL:01 SRNM:[837470WQH8]
::
=====

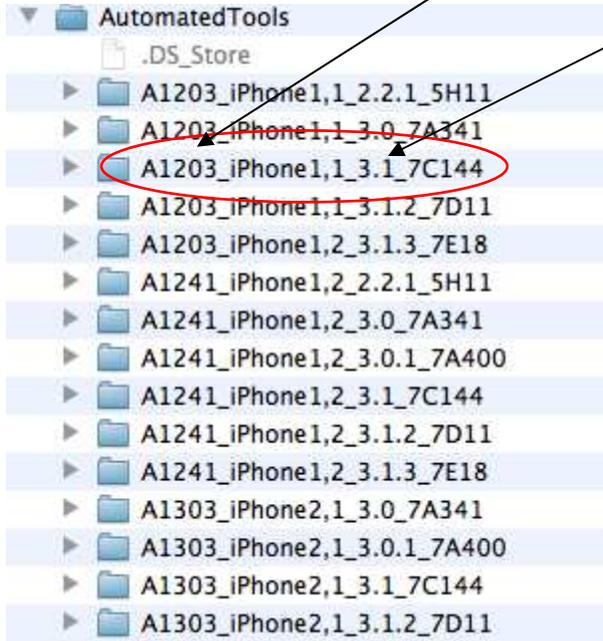
[FTL:MSG] Apple NAND Driver (AND) RO
[MAND] Device ID      0x2555d5ec
[MAND] BANKS_TOTAL   4
[MAND] BLOCKS_PER_BANK 8192
[MAND] PAGES_PER_BANK 1048576
[MAND] SECTORS_PER_PAGE 4
[MAND] BYTES_PER_SPARE 64
[FTL:MSG] FIL_Init      [OK]
[FTL:MSG] BUF_Init      [OK]
[FTL:MSG] FPart Init    [OK]
read old style signature 0x43303035 (line:371)
[FTL:MSG] UFL Register  [OK]
[FTL:MSG] UFL_Init      [OK]
[FTL:MSG] UFL_Open      [OK]
[FTL:MSG] FTL Register  [OK]
[FTL:MSG] FTL_Open      [OK]
Boot Failure Count: 1   Panic Fail Count: 0
Entering recovery mode, starting command prompt
(Recovery) iPhone$
```

Compare the iBoot number to the below listed chart (from page 32 of iPhone Forensic Investigative Methods by Jonathan Zdziarski) and obtain the corresponding version number;

The following list contains the presently known boot tags and corresponding version numbers.

iBoot-159	1.0.0 – 1.0.2
iBoot-204	1.1.0 – 1.1.1
iBoot-204.0.2	1.1.1
iBoot-204.2.9	1.1.2
iBoot-204.3.14	1.1.3 – 1.1.4
iBoot-204.3.16	1.1.5
iBoot-320.20	2.0.0 – 2.0.2
iBoot-385.22	2.1.0 – 2.1.1
iBoot-385.49	2.2.0 – 2.2.1
iBoot-596.24	3.0.0 – 3.0.1
iBoot-636.65	3.1.0 – 3.1.1
iBoot-636.66	3.1.2

Use the phone's model number (in this example 'A1203') in combination with the version number associated with the 'iBoot' number (in this case 3.1.0 – 3.1.1) and select the appropriate tool listed below.



These tools are available free to law enforcement here;  
<http://www.iphoneinsecurity.com/>.

## Firmware determined using Mac OS X;

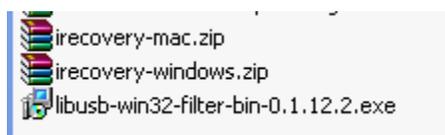
System requirements;

Apple OS X (Snow Leopard)  
iTunes (version 8.1.1)  
Internet access

1. Note the model number located on the back of the iPhone.

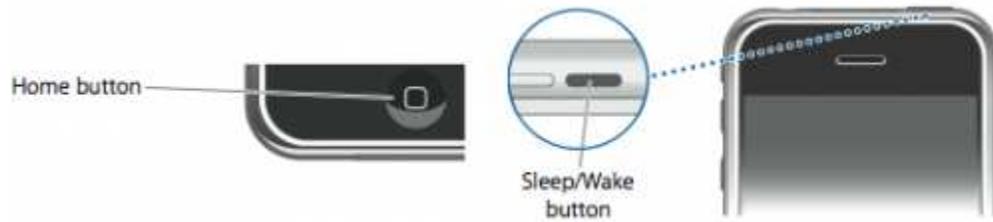


2. Download and unzip 'iHackintosh iRecovery Package for Windows & Mac.rar'  
Can be located here;  
<http://www.ihackintosh.com/2009/07/irecovery-iphone-recovery-mode-loop-restart/>  
The following programs should be included in the .rar file – (unzip 'irecovery-mac.zip');



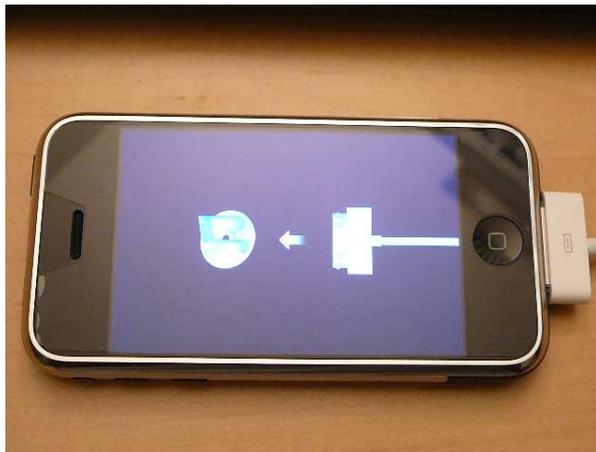
3. Connect the iPhone to your Windows machine and place phone into **Recovery** mode (not DFU mode);

Press and hold the Home button and the Sleep/Wake button at the same time.

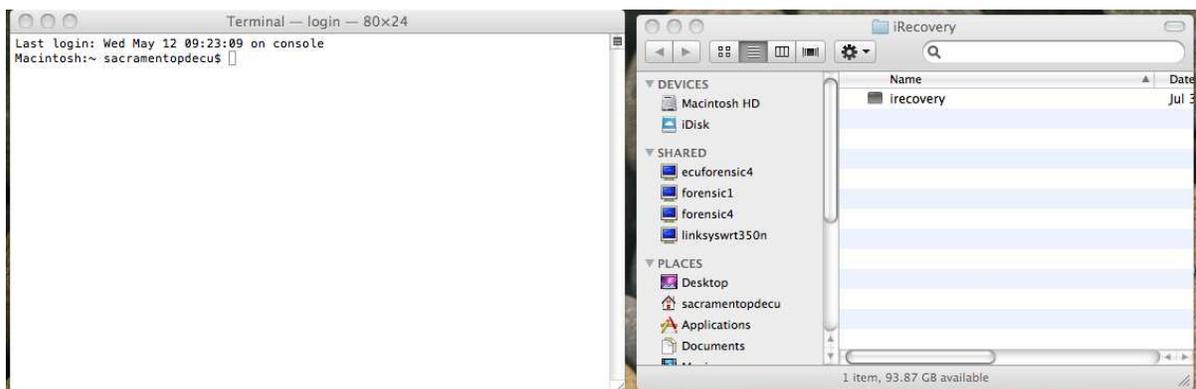


After exactly 10 seconds release the Sleep/Wake button. Continue holding the home button until your iTunes pops up a message telling you that it has detected an iPhone in recovery mode.

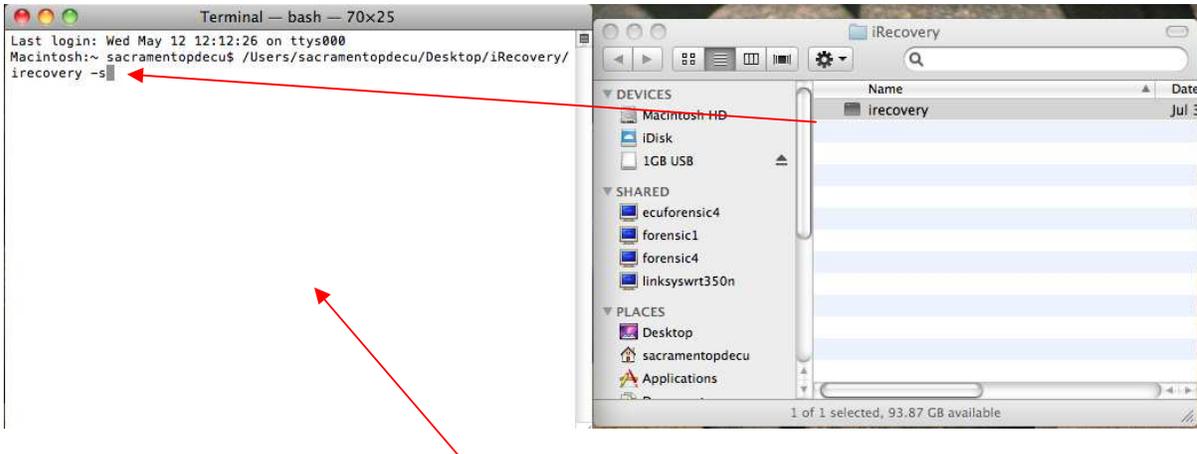
The phone should display the below listed screen.



4. Open directory containing the u-zipped files on desktop.
5. Open 'shell' on desktop.



6. Drag / drop 'irecovery' from open directory to open shell and add the '-s' switch (irecovery -s). Doing this will add the complete path and is easier than typing the path in the shell window (although this could also be done).



7. Click anywhere in the open shell making it the 'active' window and push the 'Enter' key. If all has worked correctly, information similar to the information listed below should populate the shell window.

```

Terminal — irecovery — 70x41
Last login: Wed May 12 12:12:26 on ttys000
Macintosh:~ sacramentopdecu$ /Users/sacramentopdecu/Desktop/iRecovery/
irecovery -s
irecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.

=====
::
:: iBoot for m68ap, Copyright 2009, Apple Inc.
::
::   BUILD_TAG: iBoot-636.65
::
::   BUILD_STYLE: RELEASE
::
::   USB_SERIAL_NUMBER: CPID:8900 CPRV:20 CPMF:03 SCEP:05 BDID:00 E
CID:000002D2A404DDF6 IBFL:01 SRNM:[837470WQWH8]
::
=====

[FTL:MSG] Apple NAND Driver (AND) R0
[NAND] Device ID      0x2555d5ec
[NAND] BANKS_TOTAL    4
[NAND] BLOCKS_PER_BANK 8192
[NAND] PAGES_PER_BANK 1048576
[NAND] SECTORS_PER_PAGE 4
[NAND] BYTES_PER_SPARE 64
[FTL:MSG] FIL_Init      [OK]
[FTL:MSG] BUF_Init      [OK]
[FTL:MSG] FPart Init    [OK]
read old style signature 0x43303035 (line:371)
[FTL:MSG] VFL Register  [OK]
[FTL:MSG] VFL Init      [OK]
[FTL:MSG] VFL_Open      [OK]
[FTL:MSG] FTL Register  [OK]
[FTL:MSG] FTL_Open      [OK]
Boot Failure Count: 0   Panic Fail Count: 0
Entering recovery mode, starting command prompt

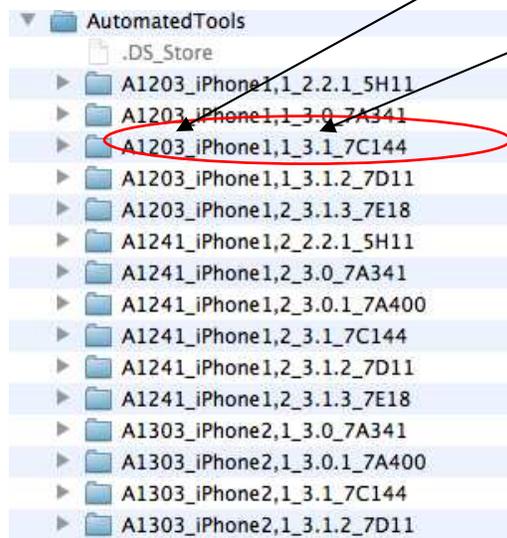
```

Compare the iBoot number to the below listed chart (from page 32 of iPhone Forensic Investigative Methods by Jonathan Zdziarski) and obtain the corresponding version number;

The following list contains the presently known boot tags and corresponding version numbers.

iBoot-159	1.0.0 – 1.0.2
iBoot-204	1.1.0 – 1.1.1
iBoot-204.0.2	1.1.1
iBoot-204.2.9	1.1.2
iBoot-204.3.14	1.1.3 – 1.1.4
iBoot-204.3.16	1.1.5
iBoot-320.20	2.0.0 – 2.0.2
iBoot-385.22	2.1.0 – 2.1.1
iBoot-385.49	2.2.0 – 2.2.1
iBoot-596.24	3.0.0 – 3.0.1
iBoot-636.65	3.1.0 – 3.1.1
iBoot-636.66	3.1.2

Use the phone's model number (in this example 'A1203') in combination with the version number associated with the 'iBoot' number (in this case 3.1.0 – 3.1.1) and select the appropriate tool listed below.



These tools are available free to law enforcement here;  
<http://www.iphoneinsecurity.com/>.

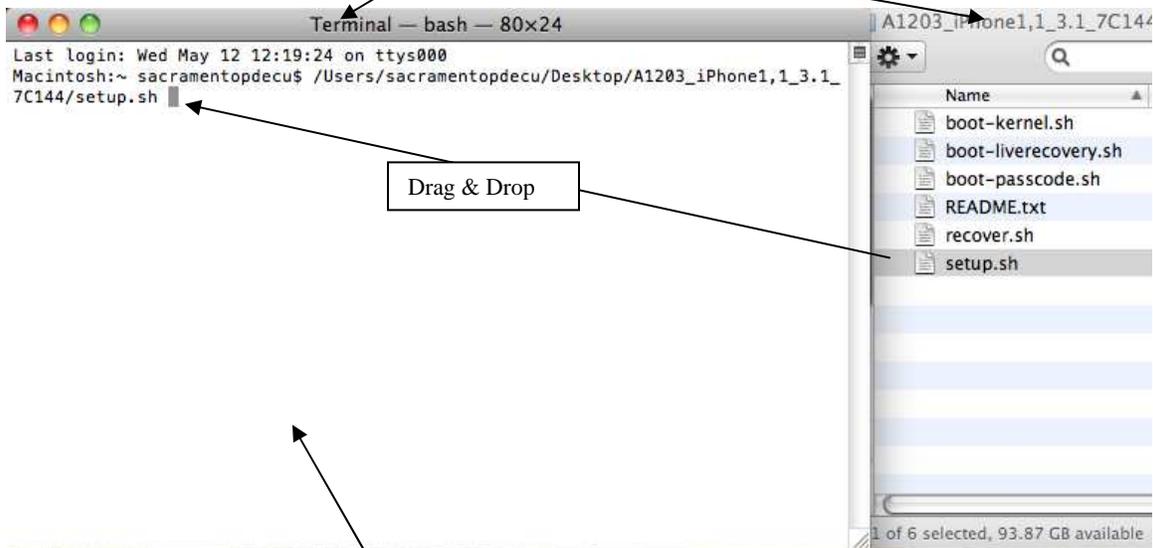
## Removing the iPhone Pass Code

Once the version number is established, copy and paste the appropriate ‘Automated Tool’ set listed above onto your desktop and rename the directory if desired. This will ensure that you have a good working copy of the data and that you do not accidentally contaminate the original tool set.

Inside each of the above listed directories are a series of tools similar to the ones listed below;

Name	Date Modified
boot-kernel.sh	Sep 16, 2009 11:46 AM
boot-liverecovery.sh	Sep 16, 2009 11:46 AM
boot-passcode.sh	Sep 16, 2009 11:46 AM
README.txt	Sep 16, 2009 11:46 AM
recover.sh	Sep 16, 2009 11:46 AM
setup.sh	Mar 1, 2010 11:14 AM

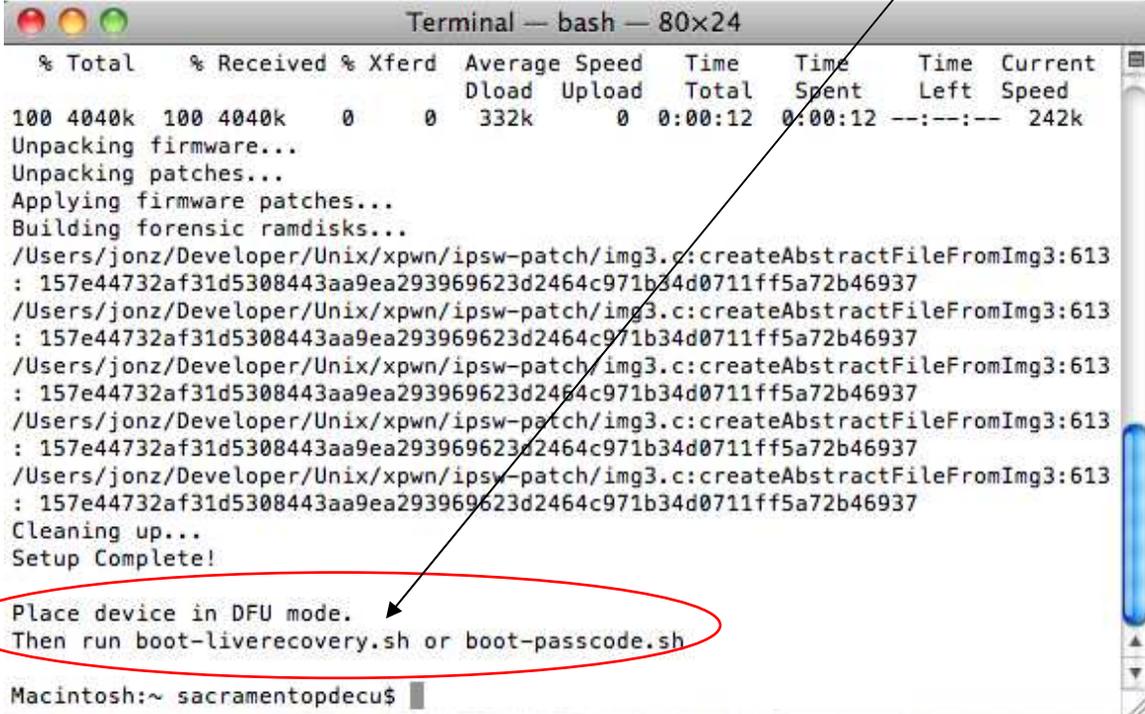
To use these tools, open a shell window as well as the directory containing the tools (I find it easiest if they are organized side by side). Drag and drop ‘setup.sh’ file into the shell.



Click anywhere in the shell window (to make it active) and press ‘Enter’ to launch the script.



When complete, the shell will display options for the user, follow the instructions. If the iPhone has a passcode, run 'boot-passcode.sh' to continue. If there is no passcode on the iPhone, run 'boot-liverecovery.sh' dump the phone's contents.



```
Terminal — bash — 80x24
% Total   % Received % Xferd  Average Speed   Time    Time     Time  Current
   100  4040k   100  4040k    0     0   332k    0  0:00:12  0:00:12  --:--:-- 242k
Unpacking firmware...
Unpacking patches...
Applying firmware patches...
Building forensic ramdisks...
/Users/jonz/Developer/Unix/xpwn/ipswn-patch/img3.c:createAbstractFileFromImg3:613
: 157e44732af31d5308443aa9ea293969623d2464c971b34d0711ff5a72b46937
Cleaning up...
Setup Complete!

Place device in DFU mode.
Then run boot-liverecovery.sh or boot-passcode.sh

Macintosh:~ sacramentopdecu$
```

To remove a passcode, follow the above listed instructions by first placing the phone in DFU mode.

To put the phone into DFU mode, **connect the phone to your computer** and then (from iPhone Forensics Cheat Sheet by Jonathan Zdziarski);

### DFU Mode

1. Power off device
2. Wait 5 seconds, then:
  - a. Hold Home and Power for 5 seconds
  - b. Release Power only, continue holding Home
  - c. Wait 10 seconds, verify "USB DFU Device"

When in DFU mode, the iPhone screen will appear blank; there will be no signs on the phone that it is in DFU mode. To confirm that a phone is in DFU mode;

Launch the Mac System Profiler and choose USB in the left pane. In the phone was successfully put into DFU mode, it will show in the right pane. This window does not 'auto update', it must be re-started each time a change is made.



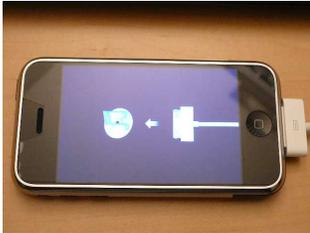
Once the phone is connected to your computer and the iPhone has been successfully been put into DFU mode, drag 'boot-passcode.sh' into the open shell, click in the shell to make it the active window and press enter to launch.

Name	Date Modified
boot-kernel.sh	Sep 16, 2009 11:46 AM
boot-liverecovery.sh	Sep 16, 2009 11:46 AM
boot-passcode.sh	Sep 16, 2009 11:46 AM
README.txt	Sep 16, 2009 11:46 AM
recover.sh	Sep 16, 2009 11:46 AM
setup.sh	Mar 1, 2010 11:14 AM

The script will list whether phone is to be in DFU mode of Recovery mode – follow the instructions shown on the screen.

**Recovery Mode:**

Home + Power until screen shows ‘Recovery Mode’ (displayed below)



**DFU Mode:**

Home + Power for 5 seconds

Release Power button (only) and wait for 10 seconds (screen will be blank)

Verify USB DFU mode in System Profile Application

When text in shell tells you,

`#####DISCONNECT AND RECONNECT FROM USB#####`

Disconnect phone from cable and then reconnect the phone (quickly). This command may be displayed several times during the process.

If this process worked correctly, the phone will reboot on its own and the passcode will have been removed.

Once the iPhone’s passcode has been removed, the phone can be processed using a number of analysis tools including;

Cellebrite UFED

Susteen DataPilot

Paraben Device Seizure

CellIDEK

iPhones may also be imaged if necessary (using the following instructions) allowing examiners to search for data that has been deleted from the iPhone as well as obtain data not normally documented by standard tools.

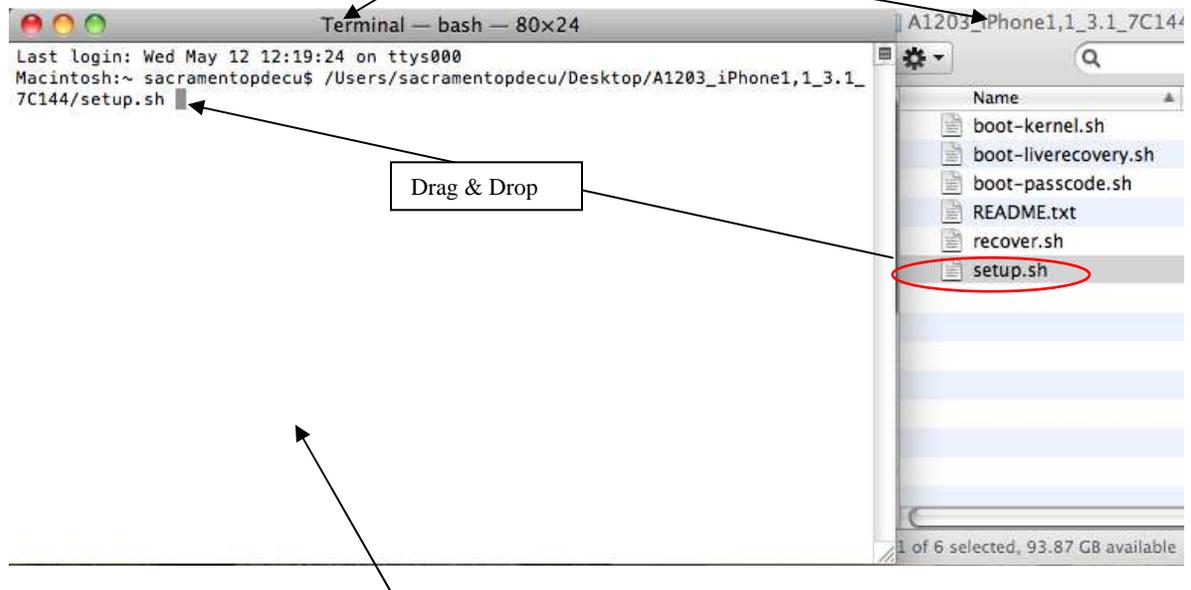
## Imaging the iPhone

Once the version number is established, copy and paste the appropriate ‘Automated Tool’ set listed above onto your desktop and rename the directory if desired. This will ensure that you have a good working copy of the data and that you do not accidentally contaminate the original tool set.

Inside each of the above listed directories are a series of tools similar to the ones listed below;

Name	Date Modified
boot-kernel.sh	Sep 16, 2009 11:46 AM
boot-liverecovery.sh	Sep 16, 2009 11:46 AM
boot-passcode.sh	Sep 16, 2009 11:46 AM
README.txt	Sep 16, 2009 11:46 AM
recover.sh	Sep 16, 2009 11:46 AM
setup.sh	Mar 1, 2010 11:14 AM

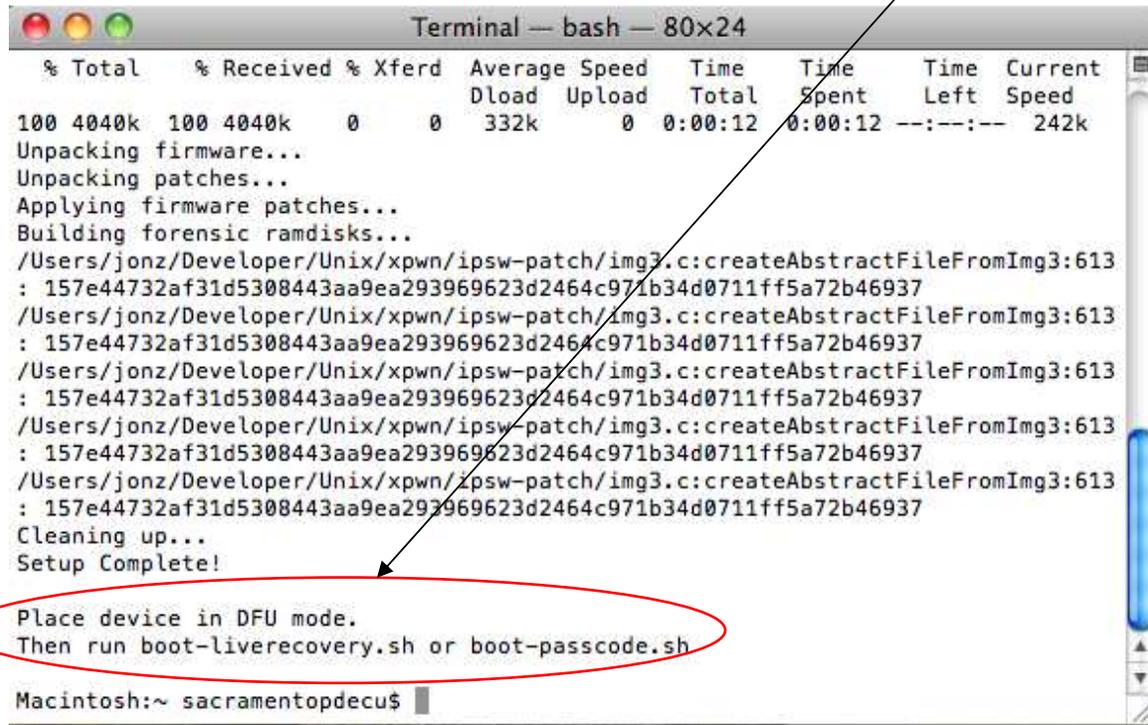
To use these tools, open a shell window as well as the directory containing the tools (I find it easiest if they are organized side by side). Drag and drop ‘setup.sh’ file into the shell.



Click anywhere in the shell window (to make it active) and press ‘Enter’ to launch the script.



When complete, the shell will display user options, follow the instructions. The 'boot-liverecovery.sh' utility will work to image the iPhone even if the iPhone is 'passcode' protected and the passcode has not been removed.



```
Terminal — bash — 80x24
% Total  % Received % Xferd  Average Speed   Time    Time     Time  Current
   100   4040k   100   4040k    0     0   332k      0  0:00:12  0:00:12  --:--:--  242k
Unpacking firmware...
Unpacking patches...
Applying firmware patches...
Building forensic ramdisks...
/Users/jonz/Developer/Unix/xpwn/ips-w-patch/img3.c:createAbstractFileFromImg3:613
: 157e44732af31d5308443aa9ea293969623d2464c971b34d0711ff5a72b46937
Cleaning up...
Setup Complete!

Place device in DFU mode.
Then run boot-liverecovery.sh or boot-passcode.sh

Macintosh:~ sacramentopdecu$
```

To create an image of an iPhone, follow the above listed instructions by first placing the phone in DFU mode.

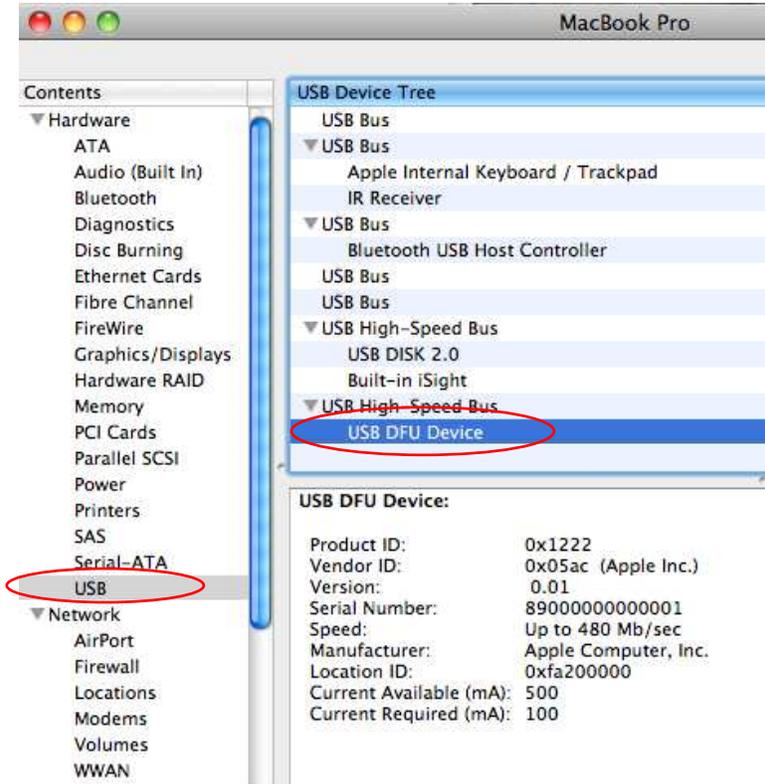
To put the phone into DFU mode, **connect the phone to your computer** and then (from iPhone Forensics Cheat Sheet by Jonathan Zdziarski);

### DFU Mode

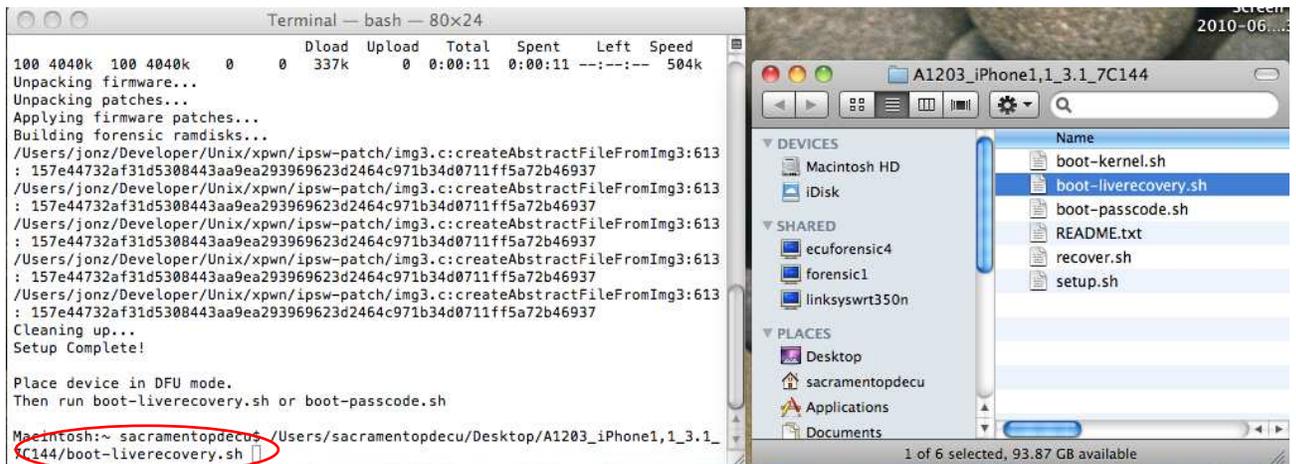
1. Power off device
2. Wait 5 seconds, then:
  - a. Hold Home and Power for 5 seconds
  - b. Release Power only, continue holding Home
  - c. Wait 10 seconds, verify "USB DFU Device"

When in DFU mode, the iPhone screen will appear blank; there will be no signs on the phone that it is in DFU mode. To confirm that a phone is in DFU mode;

Launch the Mac System Profiler and choose USB in the left pane. In the phone was successfully put into DFU mode, it will show in the right pane. This window does not 'auto update', it much be re-started each time a change is made.



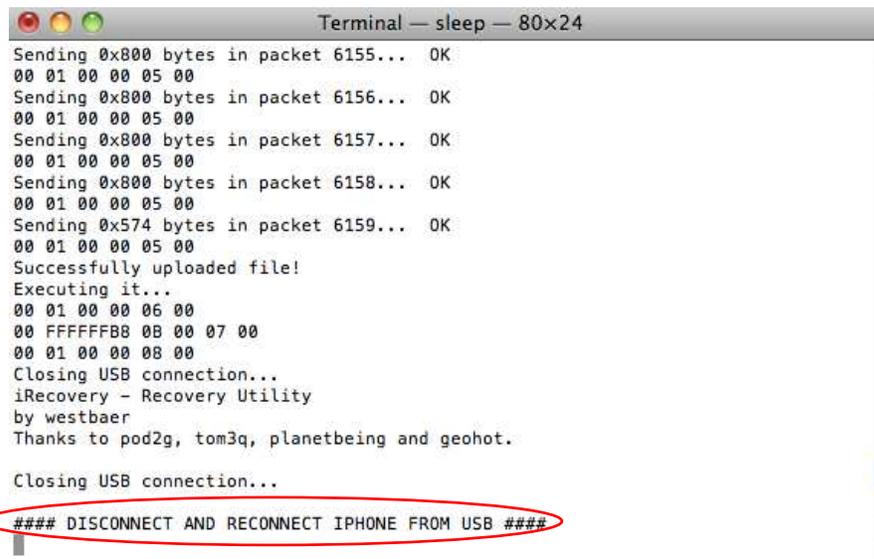
Once the phone is connected to your computer and the iPhone has been successfully been placed in DFU mode, drag 'livercovery.sh' into the open shell, click in the shell to make it the active window and press enter to launch.



While running through its process, the user will be prompted to

```
'#####DISCONNECT AND RECONNECT FROM USB#####'
```

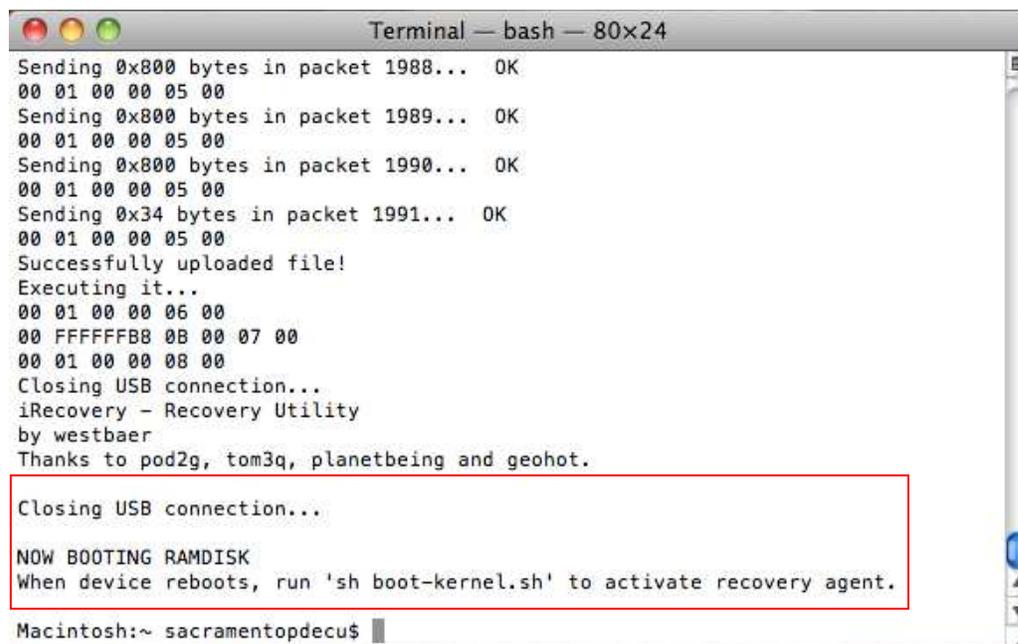
When this message is displayed, disconnect the iPhone from cable and then reconnect the iPhone (quickly - within a few seconds). This command may be displayed several times during the process.



```
Terminal — sleep — 80x24
Sending 0x800 bytes in packet 6155... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 6156... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 6157... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 6158... OK
00 01 00 00 05 00
Sending 0x574 bytes in packet 6159... OK
00 01 00 00 05 00
Successfully uploaded file!
Executing it...
00 01 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 01 00 00 08 00
Closing USB connection...
iRecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.

Closing USB connection...
### DISCONNECT AND RECONNECT IPHONE FROM USB ###
```

When completed, the screen will show the following message;



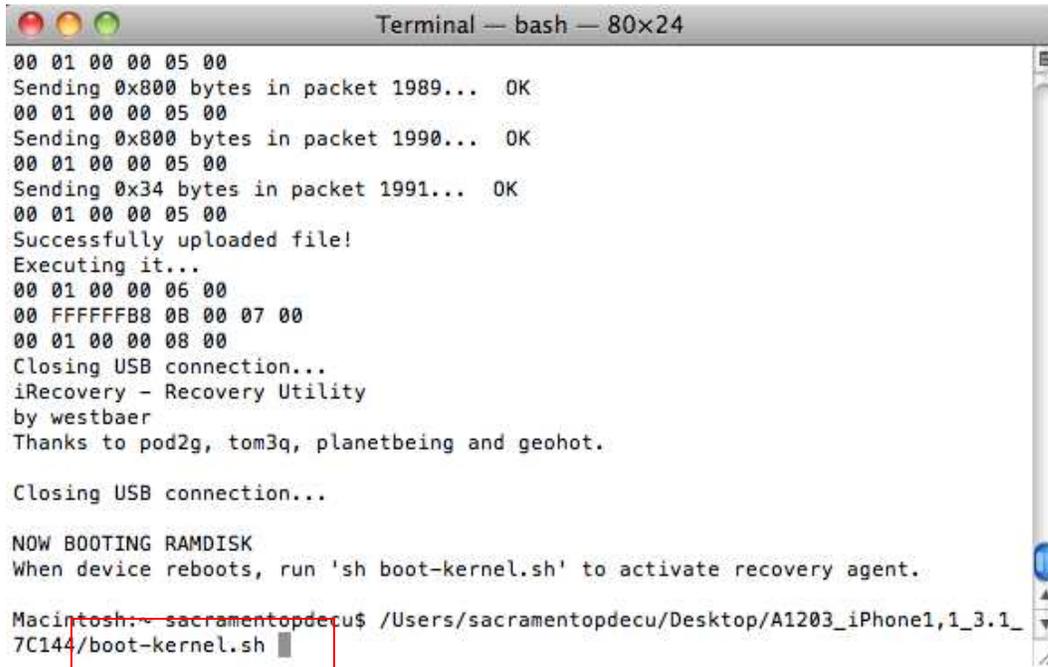
```
Terminal — bash — 80x24
Sending 0x800 bytes in packet 1988... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 1989... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 1990... OK
00 01 00 00 05 00
Sending 0x34 bytes in packet 1991... OK
00 01 00 00 05 00
Successfully uploaded file!
Executing it...
00 01 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 01 00 00 08 00
Closing USB connection...
iRecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.

Closing USB connection...

NOW BOOTING RAMDISK
When device reboots, run 'sh boot-kernel.sh' to activate recovery agent.

Macintosh:~ sacramentopdecu$
```

Drag and drop 'boot-kernel.sh' to the open shell, click anywhere in the shell to make it the active window and press enter to start the process.



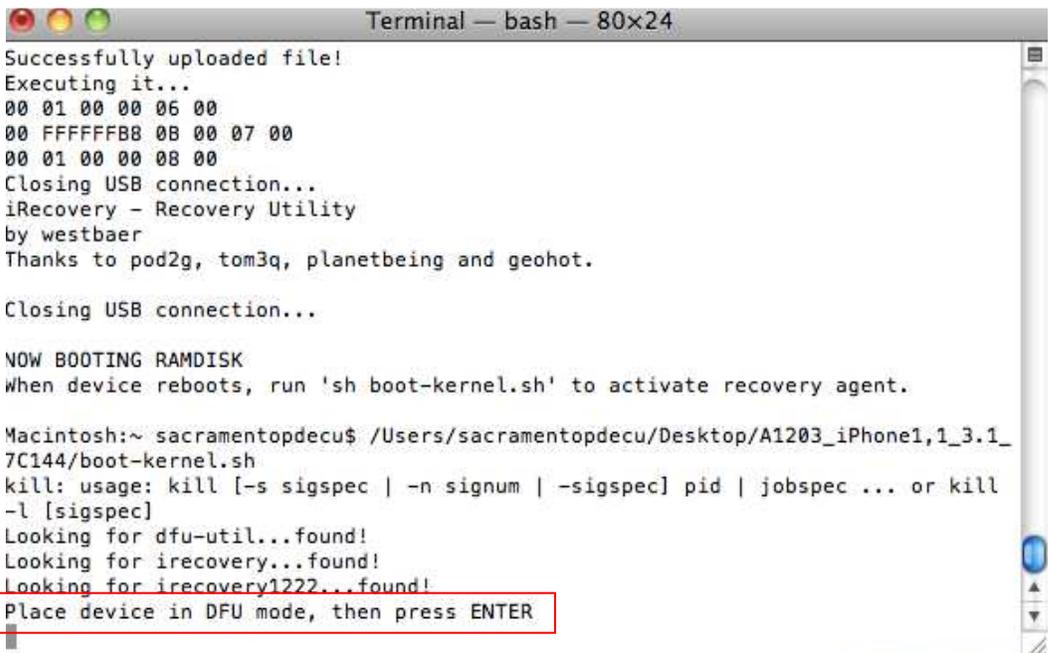
```
Terminal — bash — 80x24
00 01 00 00 05 00
Sending 0x800 bytes in packet 1989... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 1990... OK
00 01 00 00 05 00
Sending 0x34 bytes in packet 1991... OK
00 01 00 00 05 00
Successfully uploaded file!
Executing it...
00 01 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 01 00 00 08 00
Closing USB connection...
iRecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.

Closing USB connection...

NOW BOOTING RAMDISK
When device reboots, run 'sh boot-kernel.sh' to activate recovery agent.

Macintosh:~ sacramentopdecu$ /Users/sacramentopdecu/Desktop/A1203_iPhone1,1_3.1_7C144/boot-kernel.sh
```

If everything has worked properly, the following screen should be displayed (as it is working);



```
Terminal — bash — 80x24
Successfully uploaded file!
Executing it...
00 01 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 01 00 00 08 00
Closing USB connection...
iRecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.

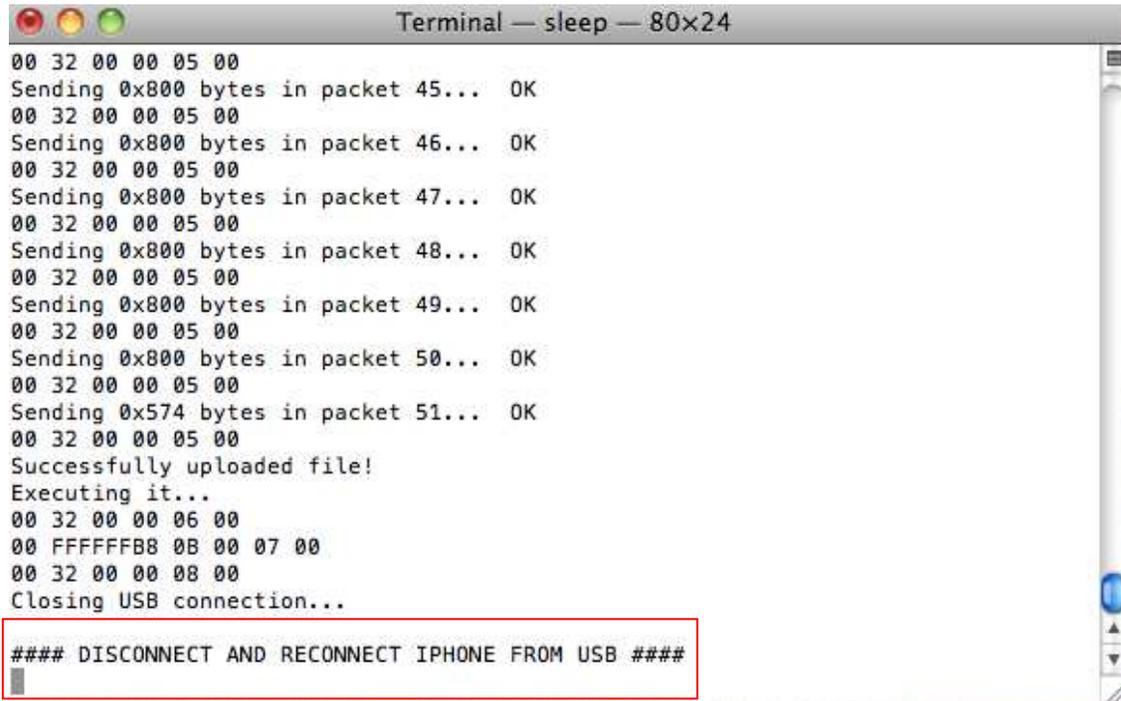
Closing USB connection...

NOW BOOTING RAMDISK
When device reboots, run 'sh boot-kernel.sh' to activate recovery agent.

Macintosh:~ sacramentopdecu$ /Users/sacramentopdecu/Desktop/A1203_iPhone1,1_3.1_7C144/boot-kernel.sh
kill: usage: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill -l [sigspec]
Looking for dfu-util...found!
Looking for irecovery...found!
Looking for irecovery1222...found!
Place device in DFU mode, then press ENTER
```

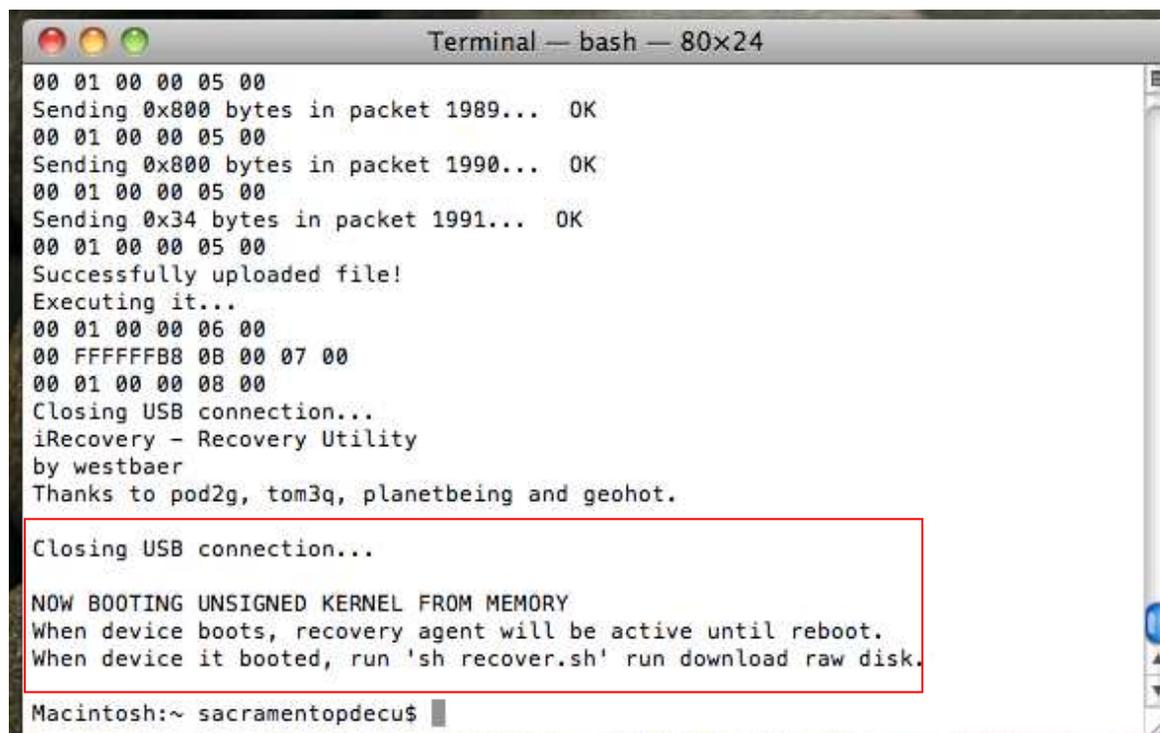
Place the iPhone back into DFU mode and press 'Enter'

Follow the instructions that are displayed including disconnecting and reconnecting the iPhone when prompted;



```
Terminal — sleep — 80x24
00 32 00 00 05 00
Sending 0x800 bytes in packet 45... OK
00 32 00 00 05 00
Sending 0x800 bytes in packet 46... OK
00 32 00 00 05 00
Sending 0x800 bytes in packet 47... OK
00 32 00 00 05 00
Sending 0x800 bytes in packet 48... OK
00 32 00 00 05 00
Sending 0x800 bytes in packet 49... OK
00 32 00 00 05 00
Sending 0x800 bytes in packet 50... OK
00 32 00 00 05 00
Sending 0x574 bytes in packet 51... OK
00 32 00 00 05 00
Successfully uploaded file!
Executing it...
00 32 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 32 00 00 08 00
Closing USB connection...
#### DISCONNECT AND RECONNECT IPHONE FROM USB ####
```

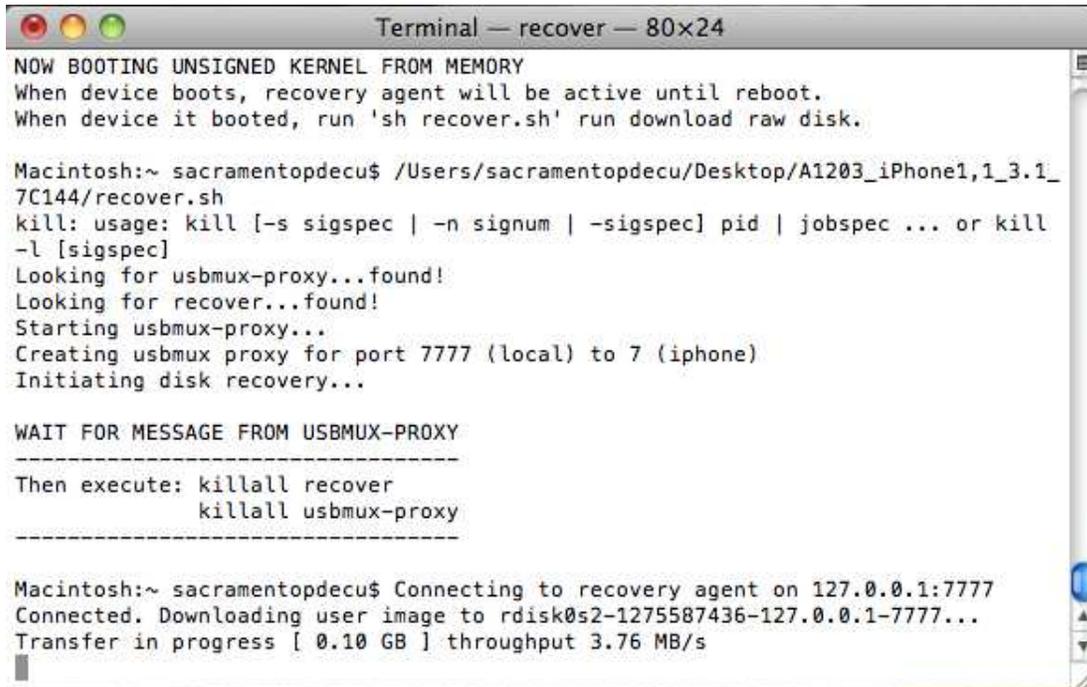
When complete, the following screen will be displayed;



```
Terminal — bash — 80x24
00 01 00 00 05 00
Sending 0x800 bytes in packet 1989... OK
00 01 00 00 05 00
Sending 0x800 bytes in packet 1990... OK
00 01 00 00 05 00
Sending 0x34 bytes in packet 1991... OK
00 01 00 00 05 00
Successfully uploaded file!
Executing it...
00 01 00 00 06 00
00 FFFFFFFB8 0B 00 07 00
00 01 00 00 08 00
Closing USB connection...
iRecovery - Recovery Utility
by westbaer
Thanks to pod2g, tom3q, planetbeing and geohot.
Closing USB connection...
NOW BOOTING UNSIGNED KERNEL FROM MEMORY
When device boots, recovery agent will be active until reboot.
When device is booted, run 'sh recover.sh' run download raw disk.
Macintosh:~ sacramentopdecus$
```

Follow the instructions by dragging 'recover.sh' into the shell and pressing 'Enter'

The following screen should be displayed if everything is working properly. The following screen will show the image process. When complete, the iPhone image should reside on the computer and can be examined using a variety of tools.

A screenshot of a macOS Terminal window titled "Terminal — recover — 80x24". The terminal displays the output of a script named "recover.sh". The script starts with instructions for booting an unsigned kernel from memory and activating a recovery agent. It then runs the script from a Macintosh user's desktop. The script checks for and starts "usbmux-proxy", creating a proxy for port 7777 on the local machine to connect to the iPhone. It then initiates disk recovery and waits for a message from the USBMux-Proxy. The user is instructed to run "killall recover" and "killall usbmux-proxy". Finally, the terminal shows the connection to the recovery agent on 127.0.0.1:7777, the downloading of the user image to a specific disk, and the start of a transfer at 3.76 MB/s.

```
Terminal — recover — 80x24
NOW BOOTING UNSIGNED KERNEL FROM MEMORY
When device boots, recovery agent will be active until reboot.
When device it booted, run 'sh recover.sh' run download raw disk.

Macintosh:~ sacramentopdecu$ /Users/sacramentopdecu/Desktop/A1203_iPhone1,1_3.1_
7C144/recover.sh
kill: usage: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill
-l [sigspec]
Looking for usbmux-proxy...found!
Looking for recover...found!
Starting usbmux-proxy...
Creating usbmux proxy for port 7777 (local) to 7 (iphone)
Initiating disk recovery...

WAIT FOR MESSAGE FROM USBMUX-PROXY
-----
Then execute: killall recover
              killall usbmux-proxy
-----

Macintosh:~ sacramentopdecu$ Connecting to recovery agent on 127.0.0.1:7777
Connected. Downloading user image to rdisk0s2-1275587436-127.0.0.1-7777...
Transfer in progress [ 0.10 GB ] throughput 3.76 MB/s
```

## Working with the image file

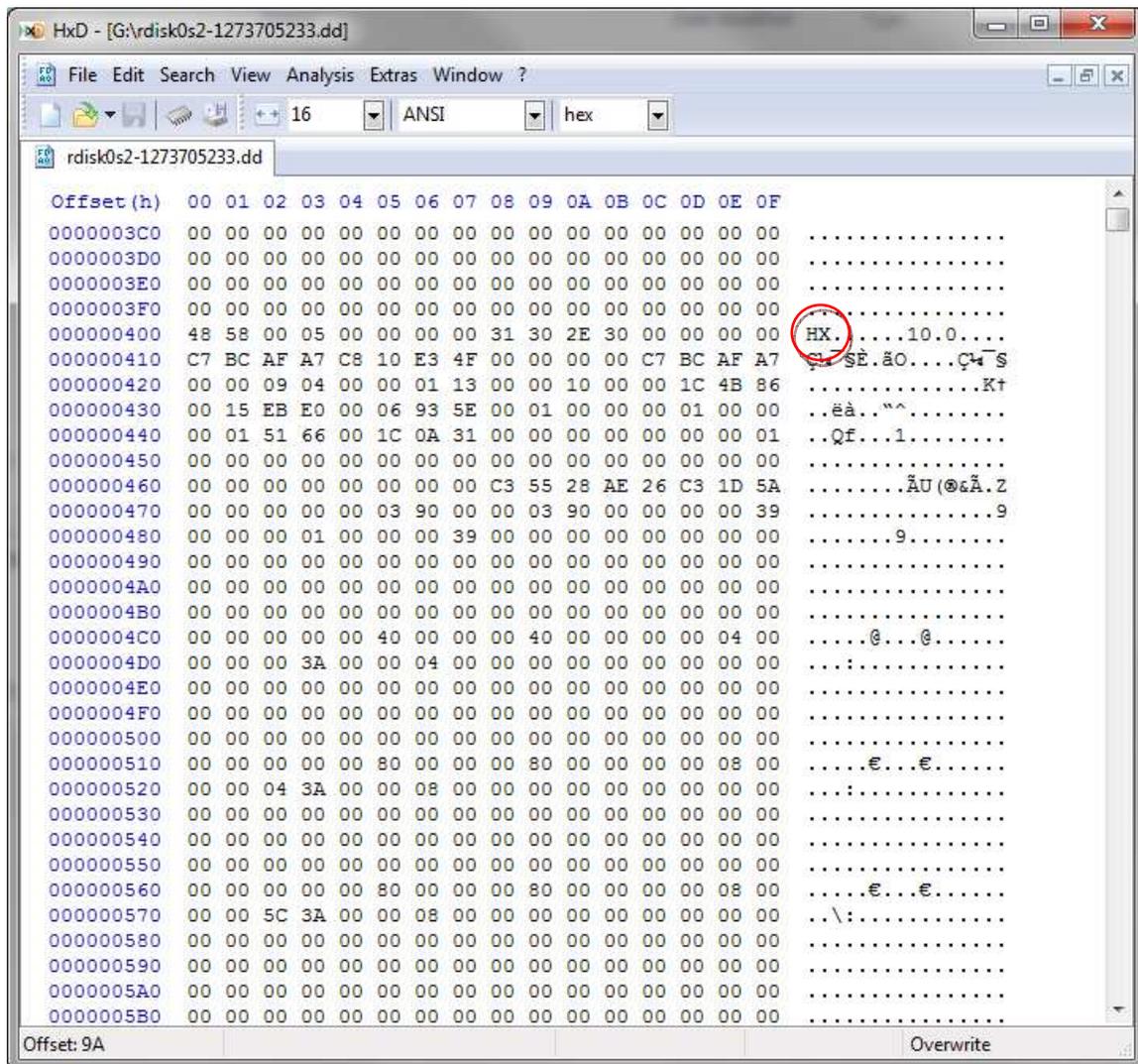
The following is an excerpt from Jonathan Zdziarski's manual (Page 96);

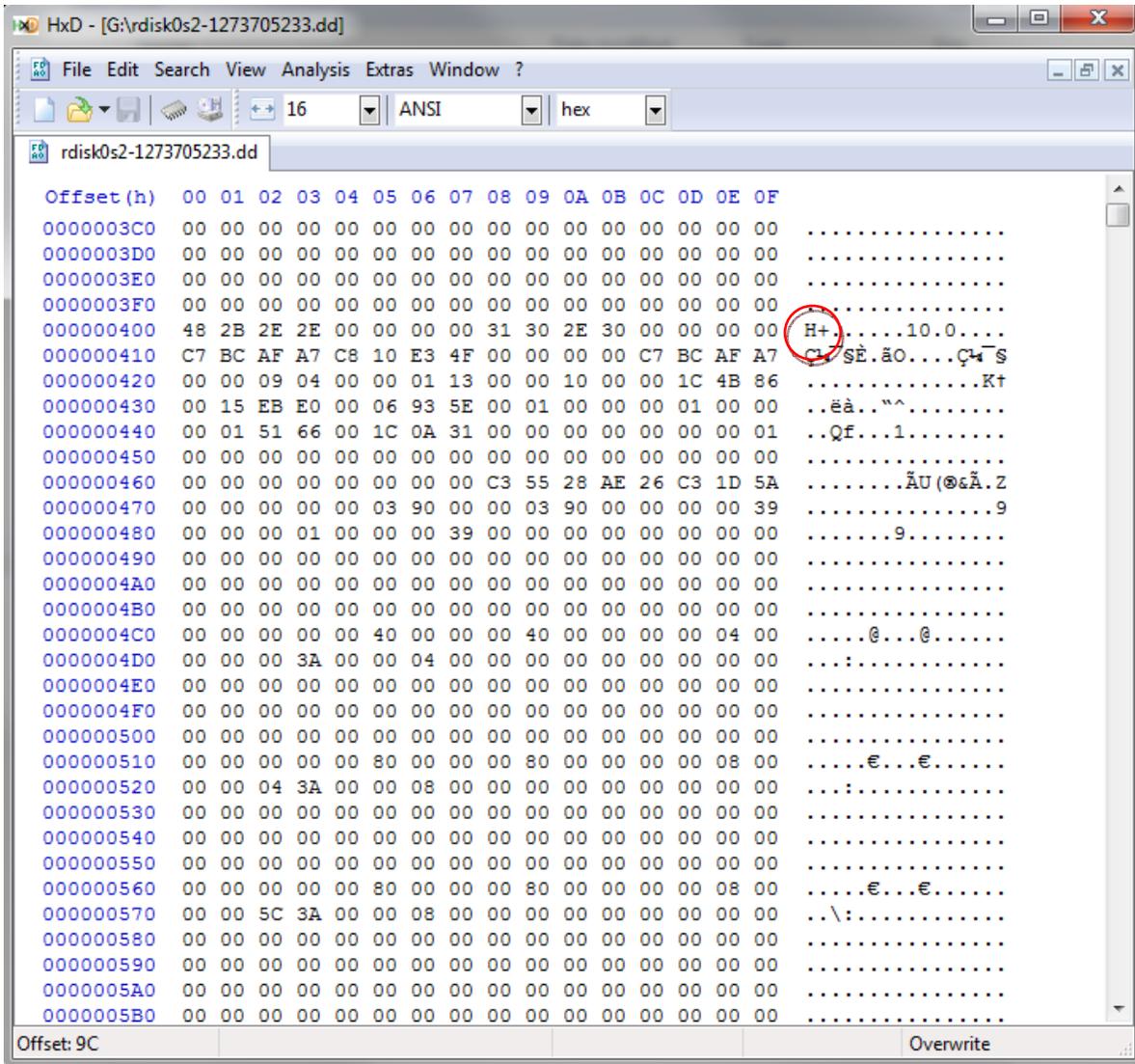
## Making Commercial Tools Compatible

Once a raw disk image has been recovered from the iPhone, it can be read by many commercial forensics tools such as Encase or FTK, but with one caveat. The disk image itself is reported as an HFS/X image (fifth generation HFS), which some tools do not yet recognize. It may be necessary to modify the file system header if your tool of choice doesn't recognize the volume. The identifier for this format is located at or around offset 0x400 inside the image file. Changing the identifier from **HX** to **H+** (denoting an HFS/+ file system) causes most existing tools to accept the file for processing. To make this change, document it and then use a hex editor, such as Hex Fiend or HexEdit 32.

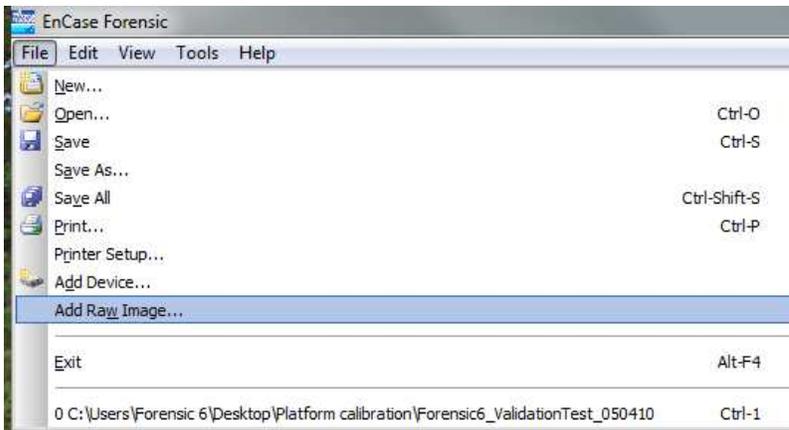
I use HxD hex editor (freeware)

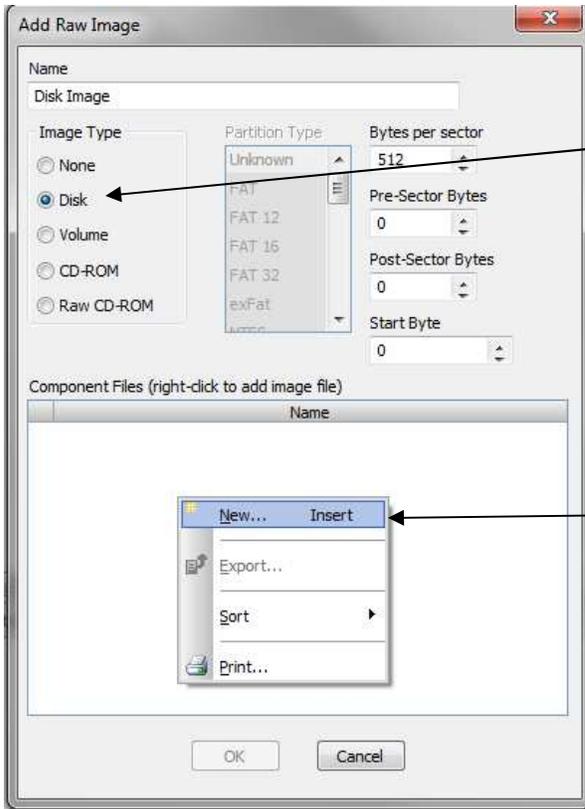
Below are screen captures showing the original header followed by the changed header;





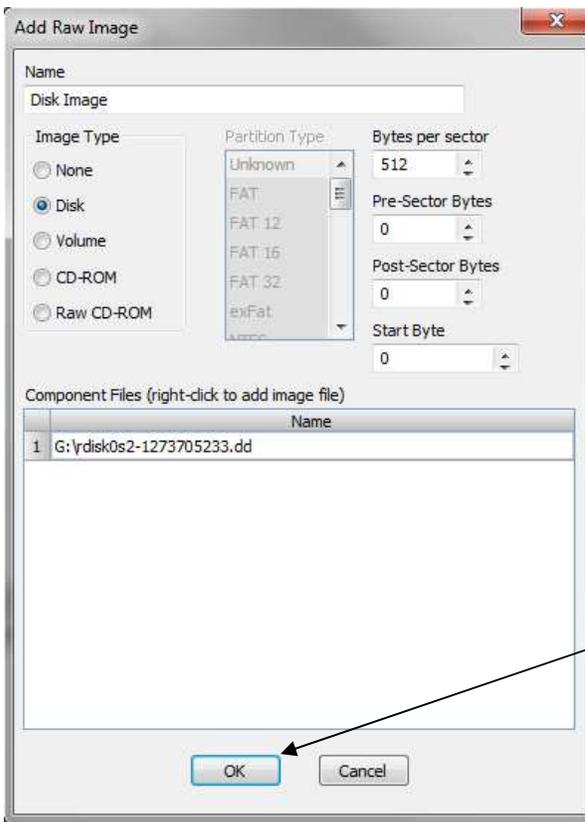
Once the identifier has been changed from HX to H+, the image can be brought into Encase as a 'Raw' image for examination as follows;





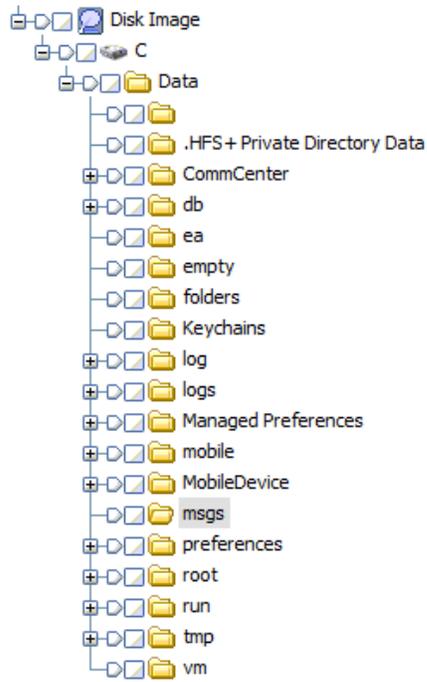
Choose 'Disk' Image Type

Rt. click and select 'New' then navigate to your image file.



Click 'OK'

Encase will then detect the proper file system and show the associated directory structure.



## Passcode Bybass (cheat sheet);

1. Determine firmware version of the phone (see above listed steps)
2. Locate the automated tools folder that corresponds to the firmware on the phone that you have.
3. Copy / paste the 'automated tools' folder to your computers desktop and rename the folder is desired (prevents altering the original).
4. Connect to the internet.
5. Open shell window
6. Run **setup.sh**
  - a. (username and password will be needed before files can be downloaded)
7. Run **boot-passcode.sh**
  - a. (script will list whether phone is to be in DFU mode of Recovery mode – follow the instructions)

### Recovery Mode:

Home + Power until screen shows  
'Recovery Mode'



### DFU Mode:

Home + Power for 5 seconds  
Release Power button (only) and  
wait for 10 seconds (screen will be  
blank)  
Verify USB DFU mode in System  
Profile Application

8. When text in shell tells you,
  - a. #####DISCONNECT AND RECONNECT FROM USB#####  
Disconnect phone from cable and then reconnect the phone (quickly). This command may be displayed several times during the process.

If this process worked correctly, the phone will reboot on its own and the passcode will have been removed.

## **iPhone Live Recovery (cheat sheet)**

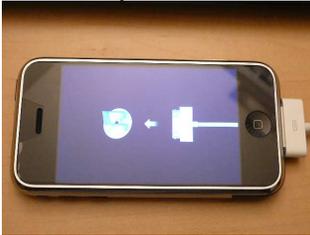
Turn phone on and connect to your computer.

Run `boot-liverecovery.sh`

(script will list whether phone is to be in DFU mode of Recovery mode – follow the instructions)

Recovery Mode:

Home + Power until screen shows  
'Recovery Mode'



DFU Mode:

Home + Power for 5 seconds  
Release Power button (only) and wait for  
10 seconds (screen will be blank)  
Verify USB DFU mode in System  
Profile Application

When text in shell tells you,

`#####DISCONNECT AND RECONNECT FROM USB#####`

Disconnect phone from cable and then reconnect the phone (quickly).

Allow phone to reboot on its own (may take a while)

Run `boot-kernel.sh`

(script will list whether phone is to be in DFU mode of Recovery mode – follow the instructions)

When text in shell tells you,

`#####DISCONNECT AND RECONNECT FROM USB#####`

Disconnect phone from cable and then reconnect the phone (quickly). This command may be displayed several times during the process.

Allow phone to reboot on its own (may take a while – screen may turn different colors)

Run `recover.sh`

Shell window will show transfer in progress if working properly.

When completed, shell will show; 'Could not read from usbmux'.

User 'Control – C' the stop process.

To finish, type;

`killall recover`

`killall usbmux-proxy`

## **Definitions;**

DFU – Device Firmware Upgrade

## iPhone Forensic Method FAQ

A few have written in with questions about the latest version of the “Zdziarski” method of iPhone forensic recovery, which is used in the automated tools available free to law enforcement agencies worldwide. This is a quick rundown of the most frequently asked questions.

**Q. Does this method “jailbreak” the device?**

No. In fact, the latest method has an extremely lightweight footprint and the device will boot back into its normal operating mode once the imaging process is complete. The latest methods do not rewrite the operating system, do not patch the NOR, do not patch the kernel, do not grant the examiner access to the device, and do not require a system restore. All of the available automated forensic tools on this site have been updated to use these new methods. The new technique does not even use the 24KPWN exploit, widely touted by the hacking community.

**Q. How can you image the device without jailbreaking?**

The system components needed to image a device are loaded into the iPhone’s RAM rather than written to disk. This allows the kernel and other components to be booted from memory. The imaging software is contained on a RAM disk, which is also booted from memory. Think of it as booting a Helix CD-ROM or a USB key chain. A small recovery agent is instituted in the protected operating area of the device. Once the imaging process is complete, the phone will reboot back into the same kernel it had when you seized it.

**Q. Do you have to bypass the passcode to image the device?**

No. The passcode and any other front-door security is all user-interface based, and the imaging software runs on a much lower level, transparent to the user interface. You’ll be able to get a raw disk image from a device that is passcode protected, has backup encryption enabled, or even has been disabled by too many passcode attempts. With that said, these tools do offer the option to bypass these functions in the event that your case requires access to the device’s user interface. For example, an active kidnapping case might call for intercepting phone calls or downloading email from the suspect’s active accounts and put saving human life as a precedent over preserving the evidence. You may also want to defeat the passcode and backup encryption in order to make commercial triage tools, such as Celebrite, compatible.

**Q. Does your tool write to any user data on the device?**

No. The user data partition is treated as sacred and no writes are made to user data whatsoever. All of the source code for these tools is also available for peer-review by the law enforcement agencies using them, so you can verify this in the code itself. Don’t trust closed source commercial tools, see it for yourself.

**Q. How long does it take to image a device?**

About 15-30 minutes is all it takes, regardless of whether you're imaging a 4GB iPhone or a 32GB iPhone 3G[s]. The method makes use of high speed USB protocols, allowing device imaging to be conducted in record time, as opposed to other commercial tools which use the slower USB serial protocol, and can take 4-6 hours, or more. Some cases just can't wait that long, and most departments are now suffering through a backlog of iPhones. Ten iPhones would take a commercial tool 40-60 hours of time! The automated tools found on this site can do all ten in 2-5 hours, or concurrently in 15-30 minutes.

**Q. What devices and firmware versions are supported?**

As of 9-16-2009, all three devices (iPhone, iPhone 3G, and iPhone 3G[s]) running all firmware versions from 1.0 – 3.1.2 are supported.

**Q. Is the hardware encryption on the 3G[s] a problem?**

No. This method invokes the device's hardware encryption chip to automatically decrypt the disk image prior to transferring it to the desktop. While the data is stored encrypted on the iPhone, you get the decrypted image on your desktop machine.

**Q. What format is the disk image in?**

The disk image is a standard HFS volume, and can either be mounted directly in Mac OS X as a .dmg file, or can be loaded into Encase, FTK, X-Ways, or a number of other tools capable of reading HFS images.

**Q. Why is this stuff free? Shouldn't you be making millions off us?**

I make a good living already. Someone needs to be supporting the good guys who are protecting our country, and since Apple won't do it, I'm doing what I can to make sure LE and the military have the tools they need to keep us safe. If you really want to support my efforts, you're invited to host an Advanced iPhone Forensics workshop on your campus. Contact me if you have at least 10 seats and would like to put a workshop together in the US or Canada.

**Q. Well I read that this other dude says your methods are jailbreaking**

Not everyone who purports to be an expert in the world of digital forensics knows entirely what they're talking about; especially when it comes to the iPhone. Anyone who believes these methods constitute jailbreaking is quite frankly ignorant of the technical details. No jailbreaking is performed here, and anyone who does understand the technical details behind it can attest to it. Another good example of why an open source solution is so important – so you can see exactly what's happening and judge for yourself.