

HEX DUMPING PRIMER

Part II

Michael Harrington, CFCE, EnCE

In the first part of this two part series on obtaining a hex dump of the data of a cell phone, I gave a brief definition of what hex dumping is and why I feel it's important to the forensic examiner. I also covered equipment and software concentrating on the UFS3 and Tornado software by SarasSoft.

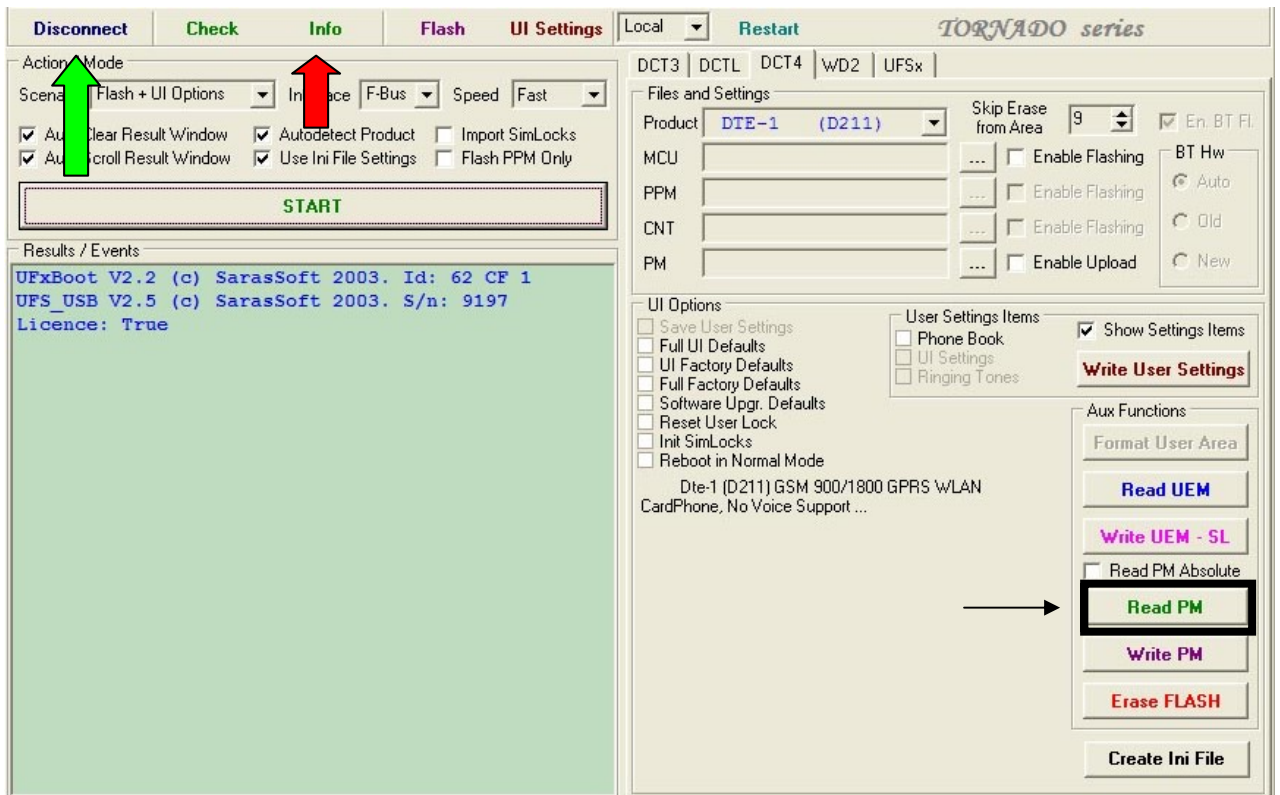
In Part II, the goal is to introduce you to the files you obtain from using the software and equipment, and how to begin to make sense of what the data you see means. Next, we will discuss research methodologies and finally put it all together decoding calendar entry obtained from a hex dump of a Nokia 6015.

As always, the caveat emptor of "Examiner Beware" holds true of using a flashing box and this software (as it must with all new techniques and software). It is incumbent upon the examiner to test, validate and understand the techniques and software he or she is reading and experimenting with. Failure to do so, can, at the very least, provide some embarrassing moments on the stand in court. At the worst, it can be destructive to your evidence.

Of .PMs and Absolutes

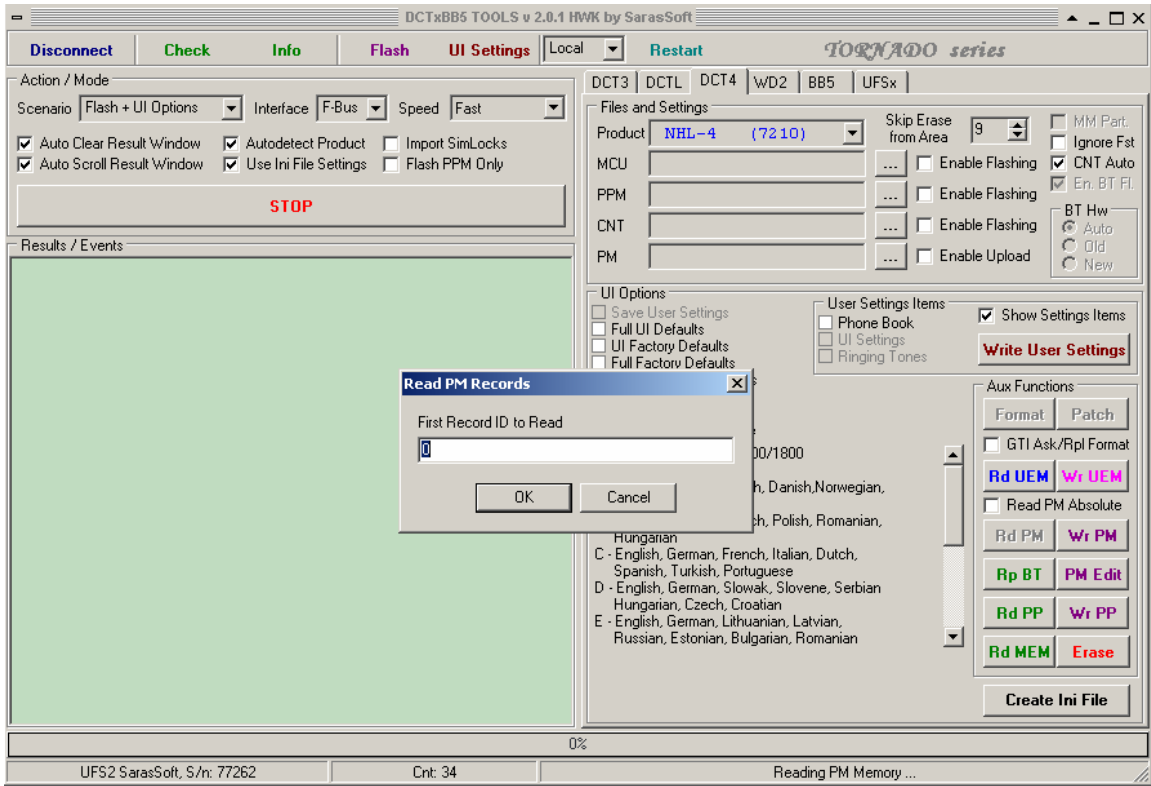
We left off our discussion of Hex Dumping in Part I of the Series having connected a phone and knowing that we want to get a snapshot of the phone's memory. But exactly what types of files are we attempting to obtain. What do they look like and what can you expect to see data-wise within them?

You may recall from Part I, the "Read PM" button from the Tornado Software interface. Below is a screen capture emphasizing this button in the software (for DCT 4).

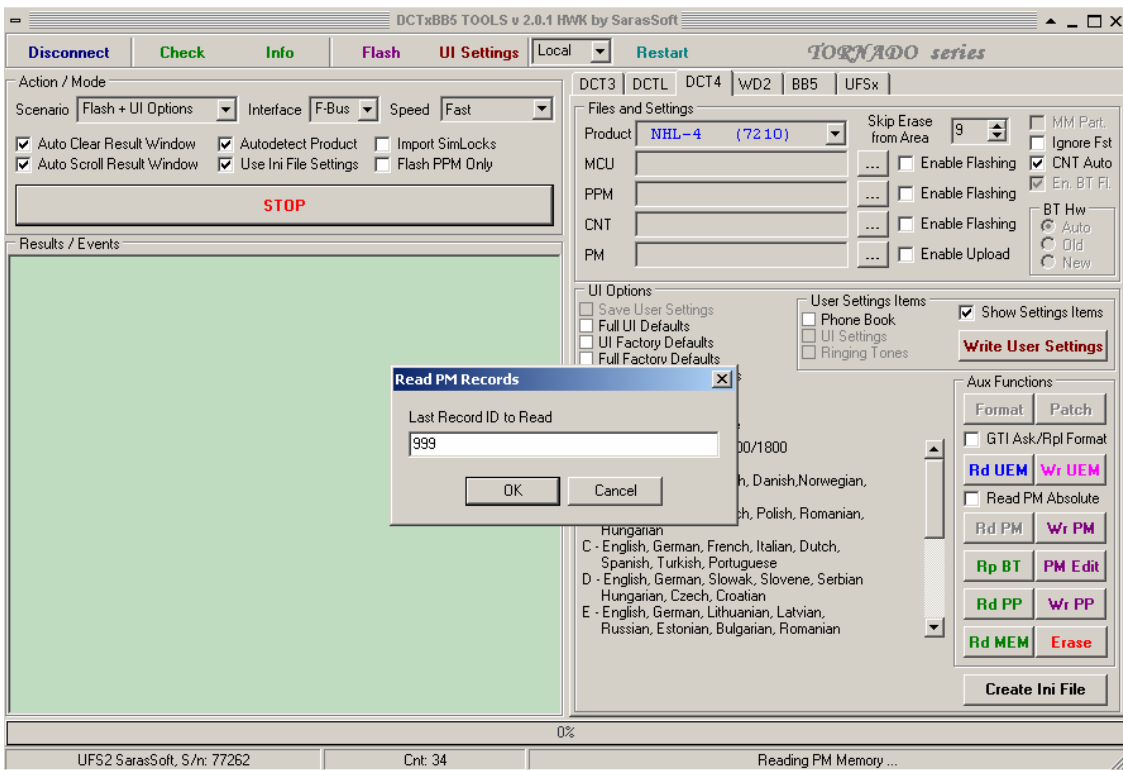


The first step in obtaining the .PM read is to connect the UFS box to the computer and fire up the Tornado software. After you have attached the box and the software up and running, press the “Connect” button (as indicated above by the green arrow). The software should read the box serial number and verify your license (if this fails to work contact your reseller.) Once you have connected the phone and verified that it is correctly “communicating” with the box and software by pressing the “Info” button on the quick buttons menu (emphasized by the red arrow above). Here’s a quick tip regarding the results window after pressing the “Info” button—you can select all the text within this window, right click and copy the result and paste it into a text file. This comes in handy for documenting the firmware version etc. in a report.

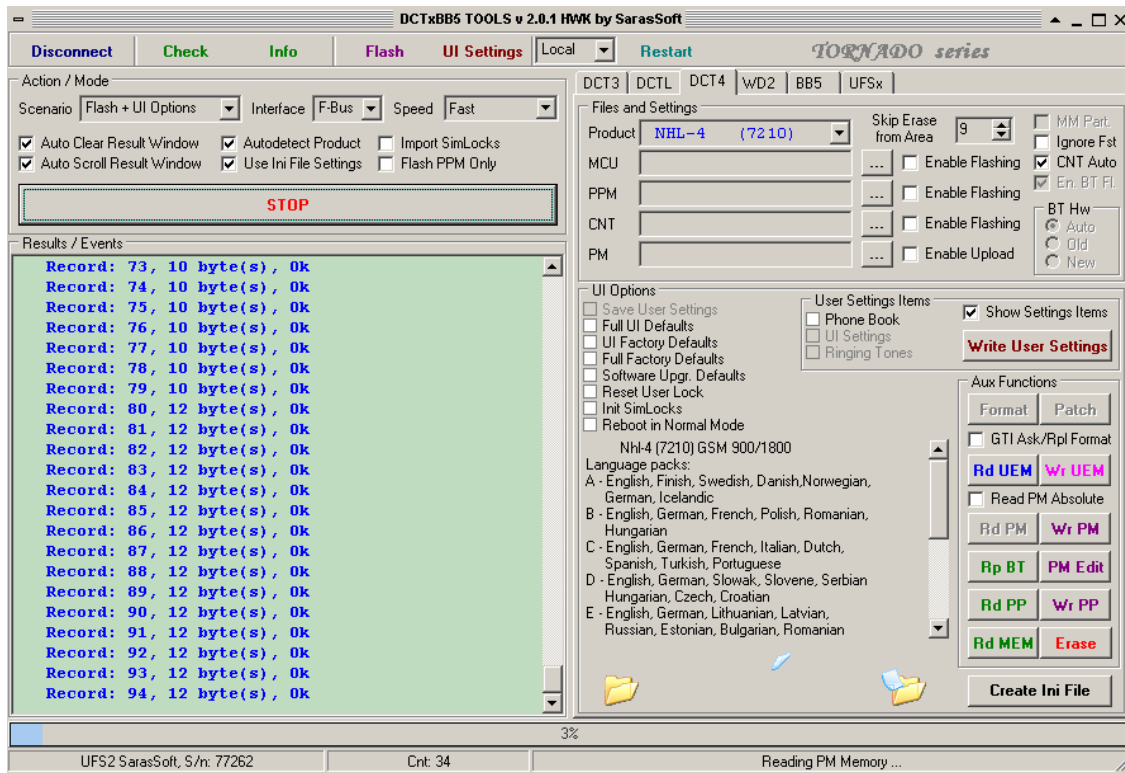
Once you have verified that the phone is correctly communicating with the software and box, you may press the “Read PM” button. After a slight pause you should be greeted with a request for the starting record—accept the default. Next you will be greeted by a request for the ending record. Enter 999 here to get ALL possible records, and press ok. The software should now start reading the records. These steps are shown in the following graphics.



The above picture shows the selection of the first record for the software to read.



The preceding picture shows the selection of the last record. Finally we have the file dump in progress.



Once the software has completed reading the records from the phone, you will be prompted to save a file to a location of your hard drive. This file will typically be pre-named with the International Mobile Equipment Identifier (IMEI) if a GSM phone or the ESN if a CDMA phone as shown in the below examples.

GSM : Rh-18_354319003928039.pm
 CDMA : Rh-55_Esn_04407218650.pm

The “RH-18” and “RH-55” refer to the firmware on the phone.

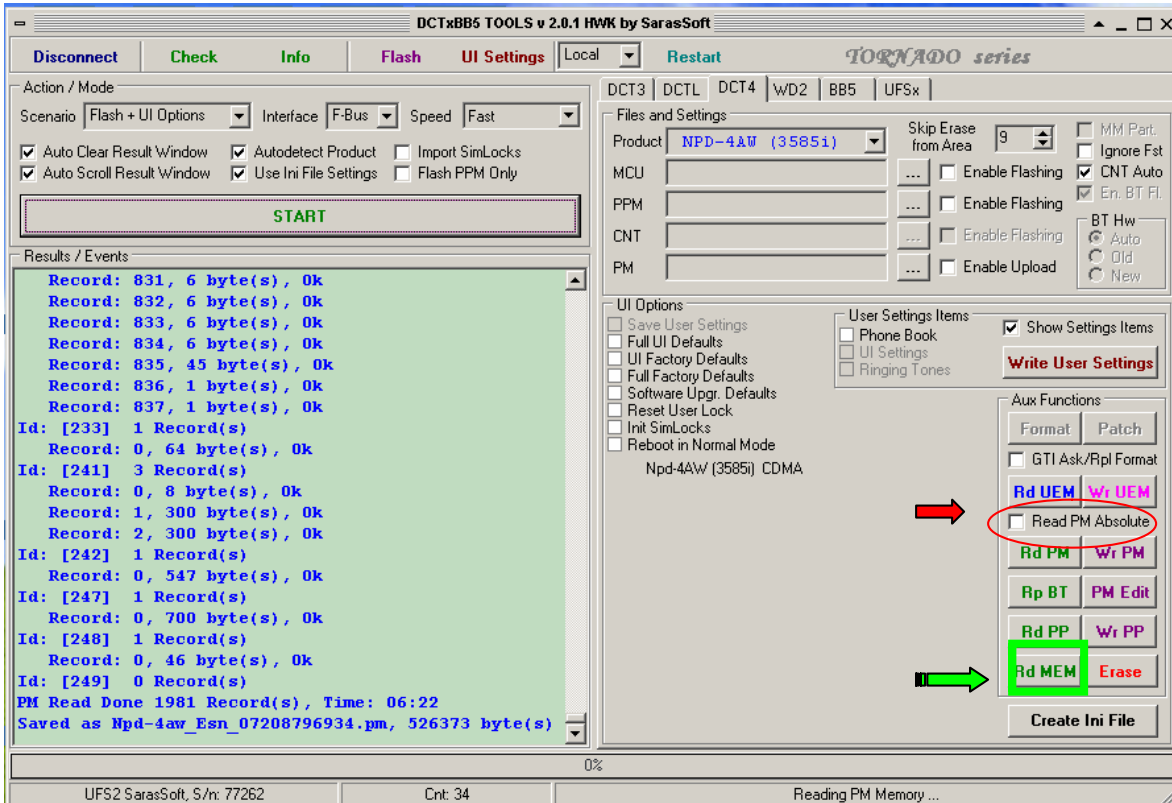
The examiner is, of course, encouraged to create a case folder structure he or she understands and is free to name the files in anyway he or she wants.

The .PM file is a read of the data records on the phone laid out in a logical fashion. The .PM can be read in any text editor-though I would recommend a more functional code editor such as PSPAD (<http://www.pspad.com/>). Below is a sample from a file.

```
0=0000000207D00B1B173B00000000383FFFF00040000000007D60B1B173B00008000004
2FFFFFFFF07BD002000000004000000000041006D00690074
1=0000000307D00C08173B00000000383FFFF00040000000007D50C08173B00008000004
2FFFFFFFF079E00200000000300000000004D006F006D
```

The PM Absolute is a bit trickier to configure and read. The PM Absolute is a binary file and is a read of the data between absolute memory addresses. The process to read this file is the same as to read a straight PM with the exception that the starting and ending absolute memory addresses must be specified.

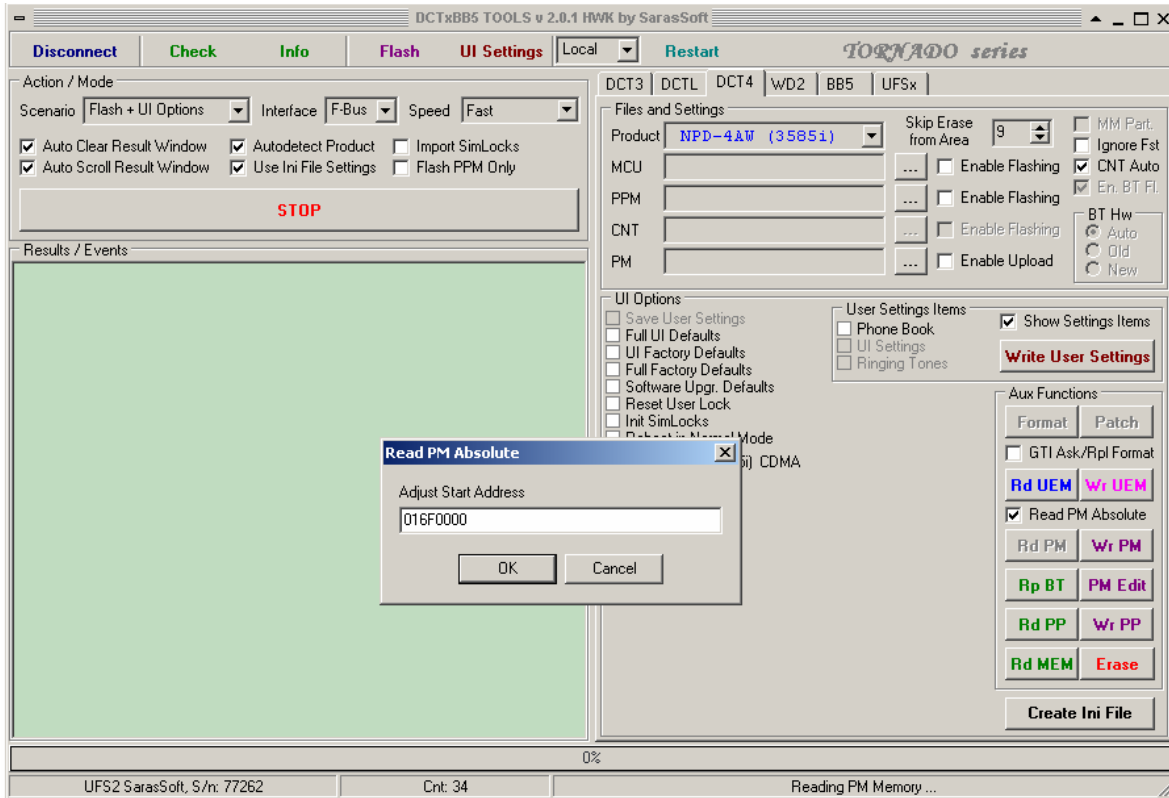
There are two ways to obtain a PM Absolute file. The first way is to press the “Rd MEM” button (shown with the green box and arrow below). If the Tornado software has the correct algorithm for the model of phone it will read the proper address range and you are save a lot of guess work.



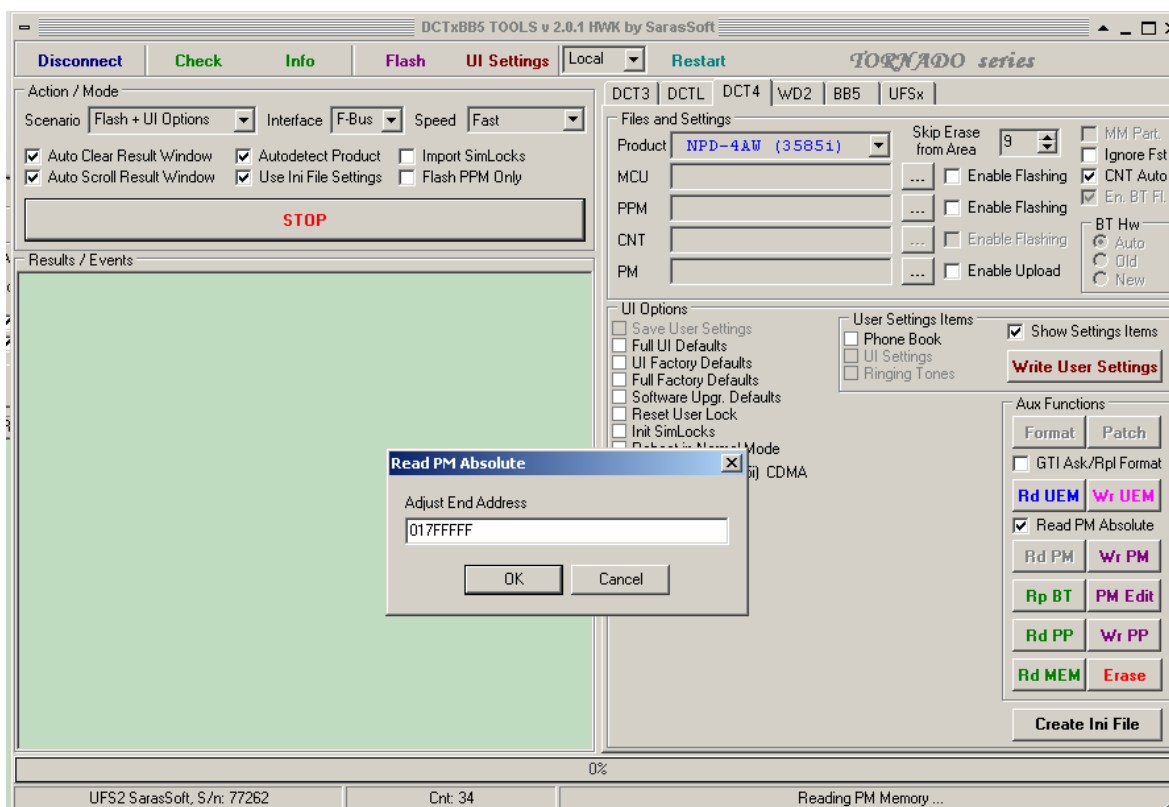
The real tricky part comes when you have to specify the memory range to read. This type of read is done by ticking the check box above the “Rd PM” button (shown with the read arrow above). Unfortunately, this involves guesswork and trial and error. Start first with the default that the software gives you. If the read is very quick and you fail to see information in the absolute that you have found in the straight PM you may need to extend the memory read. In fact, it may not hurt to take multiple reads of the absolute with extended memory ranges.

Luckily, there is a growing list of memory ranges for various Nokia phones located on the Phone Forensics forum. It would behoove the examiner to check to see if someone has done some of the legwork for you in mapping out the memory addresses. And if not, please be sure to share what you have found!

The below graphic shows the absolute memory ranges for a Nokia 3585i (CDMA).



The above graphic is for the starting address and the following for the ending address.



Obtaining the PM absolute should take longer than the straight PM and is another indication that you have selected the proper memory range.

As with the straight PM file once you have successfully obtained the absolute, you will be prompted to save the file. It will be pre-named as with the straight PM. I find it handy to name my absolute files with the abbreviation “abs” before the “.pm” extension as in the below example of a 3585i.

Npd-4aw_Esn_07207792448-abs.pm

One last comment on the PM files. They are separated into Keys and sub keys. A key is identified by a number in a bracket, such as this “[52]” and a sub key follows. Usually the sub key is a number followed by an equal sign and some hex data like thus “0=4A5D”

Research Techniques

Because this is a nascent sub discipline of cell phone forensics, it is constantly evolving and requires research and development. In this next section I am going to outline some suggestions on how to conduct this research to help you decode what you find using a flasher box. Though I am concentrating on Nokia phones, these suggestions and techniques readily apply themselves to any hex dumps you obtain (for instance hex dumps you may get from BitPim while traversing the file system).

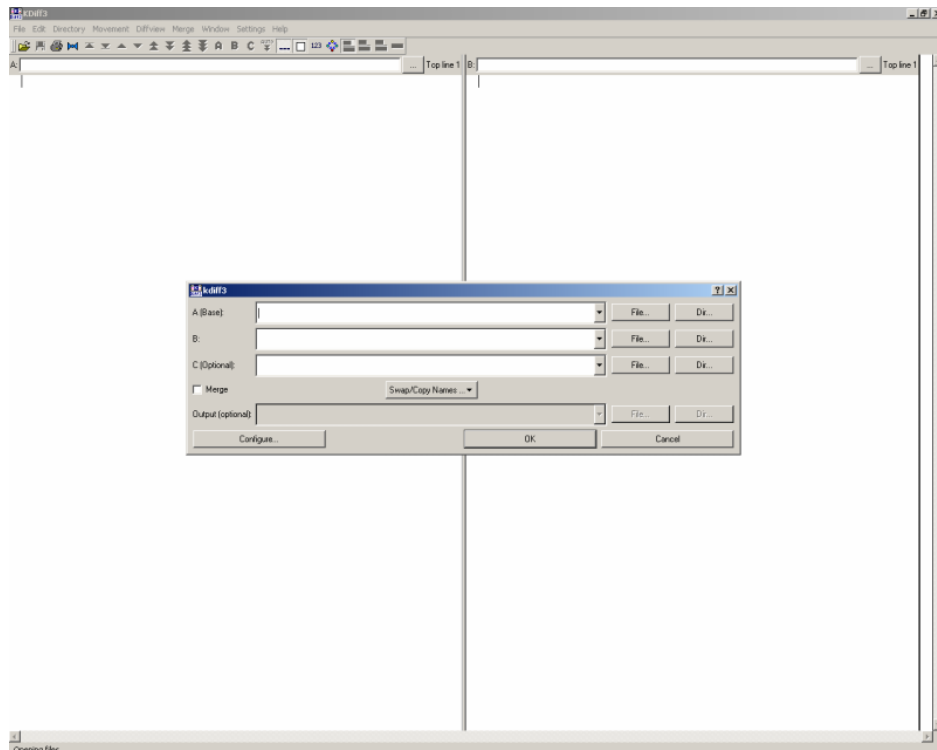
First, I would recommend getting a bed of test phones from an outlet like the popular auction site eBay or in a similar fashion. The great thing about using the flash boxes is that your test phones can be (and actually, you want them to be) in various states of repair or disrepair-since one of the reasons we are using these boxes is to go beyond the capabilities of the traditional tools. This means you can get them for CHEAP. I once bought eight Nokia 3585i phones (missing keys, no screen etc.) for five dollars including shipping. And I was able to use every one of them with the UFS3. The deals are there-look for them. Of course you might soon find (like me) that all your available storage space is taken up by cell phones-such is the price of a mobile obsession.

Once you have obtained a nice group of test phones, you will need to take a base read of them using the UFS3. After your test reads you can activate the phones on a wireless network if you choose and place calls, text messages etc. I would recommend sticking to one area of research at a time however.

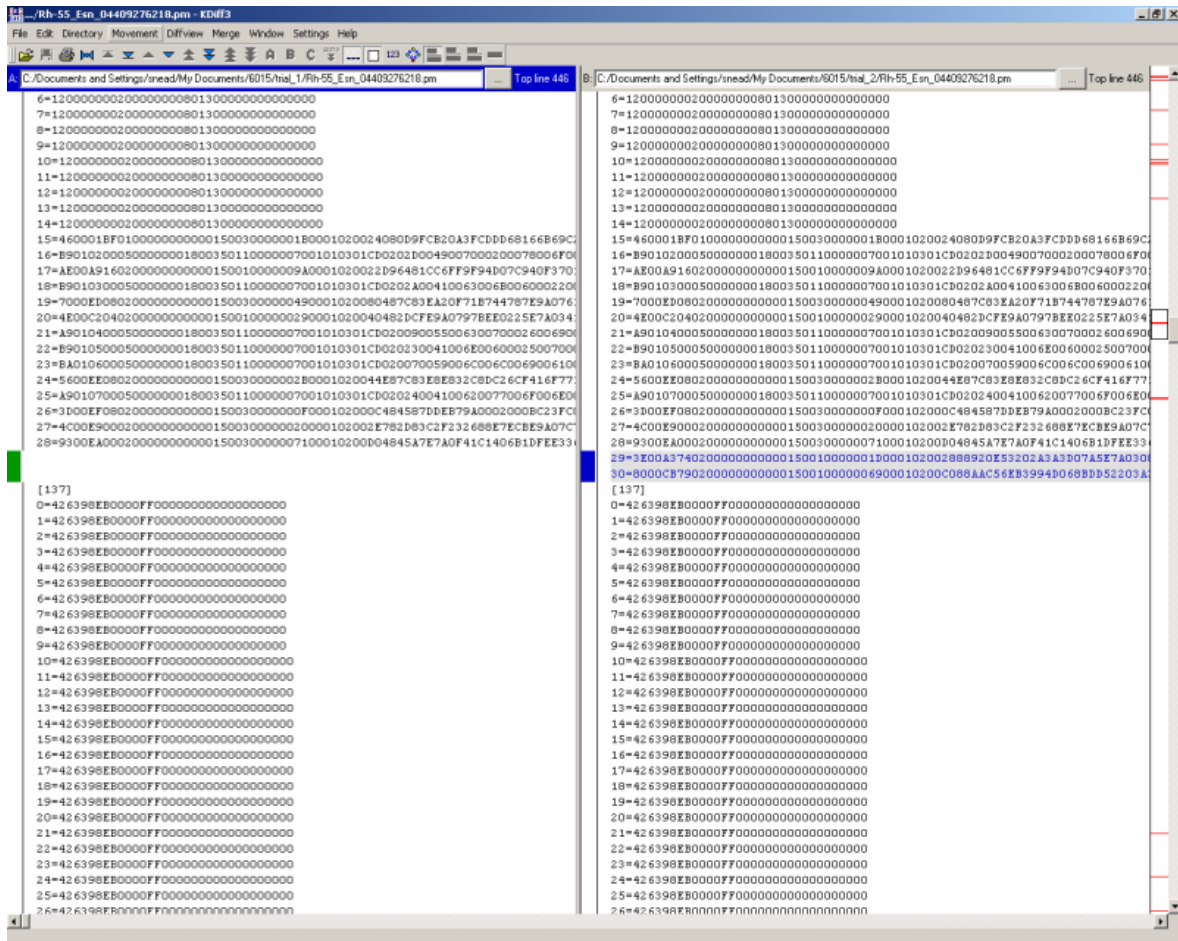
When you have your test data loaded onto the phone, you then take another read of the phone. The two resulting files can then be compared for changes.

How does one compare the two files to see where the changes have occurred in the reads?

For comparing two non-binary files, I have found the program Kdiff3 to be very useful. The program is free and available on Source Forge (<http://sourceforge.net/projects/kdiff3>). It can be used to compare up to three files at one time. Below are some screen shots of Kdiff3 in action.



This first shot shows the start-up of Kdiff3 with the file load dialog.



This shot shows Kdiff3 comparing two PM files. Notice the changes highlighted and also the red lines in the far right gutter. These are quick links to changes with compared files. The thicker the red lines, the more changes that have occurred.

Using Kdiff3 and test reads conjunctively you can identify for the type of data you are looking to decode - i.e. SMS, Call Records, Calendar etc.

Patterns and Leveraging Tools

So you have done some test reads and have applied Kdiff3 to them. You have identified which keys and sub keys you are interested in. So where do you go from here?

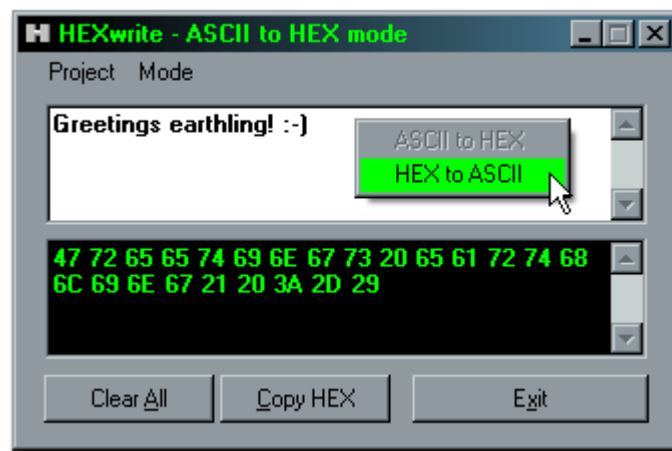
Well that is one part experience, one part trial and error, one part luck and one part black magic. Ok, I'm kidding about the last part. However, there *is* a mix of the first three involved.

One thing you should do as you work through your decoding is regularly manually check what your test phone is showing you versus your interpretation. This will help you validate your findings and remind you of other information you might have overlooked. It may also help you interpret the hex data by revealing patterns.

Patterns are something you need to develop a critical eye for. As you scan the sub keys of the area are examining, try to spot repetitions and duplications of hex data. You can then separate the data (I use tabs to isolate the hex) and work on it.

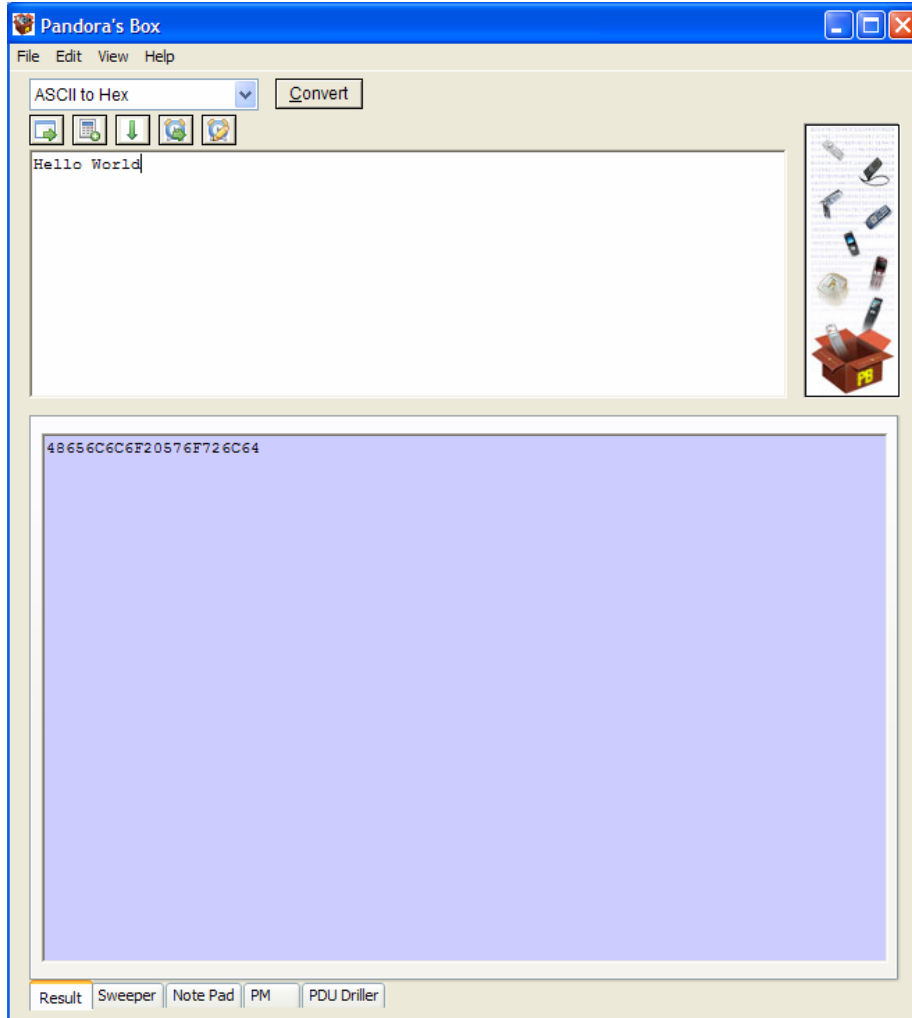
Absolutes are a different animal. You will need a hex editor capable of loading binary files. I prefer WinHex, a commercial tool. The thing you need to do with absolutes is to attempt to locate known data culled from a PM and go from there. Further discussion of interpreting Absolutes is beyond the scope of this paper.

I often use windows calculator to take the hex and change it into a decimal value; it is handy because it's built into Windows and it's free. Another useful piece of freeware is HEXwrite. This small stand alone utility (again free) is useful for converting ASCII to Hex and vice versa. It is available at <http://bluefive.pair.com/hexwrite.htm>.



You should also pick up a PDU encoder/decoder as some of the data you might be interested in might be PDU encoded (SMS). One of the tools I use is located here- <http://www.codeproject.com/vb/net/PDUDecoder.asp>.

There are commercial tools available for looking at PM files. Cell Phone Analyzer™ from BKForensics and Data Lifter is one such tool. Guidance Software currently has a tool in development called Neutrino™ that is intended harness the power of EnCase™ to interpret the hex data. I also have a tool in development that will be released shortly called Pandora's Box™. This tool has many functions built into it for working on the raw hex of a PM or Absolute File. Below is a screen capture of the tool.



This tool will load any standard PM and a variety of Absolutes. The intention is to provide the examiner with as many conversion functions and interpretation tools under one hood for an affordable price.

Lastly, the examiner will want a good ASCII chart reference. You can find one here <http://www.asciitable.com/>.

Decoding Calendar Data

As promised we are going to decode and interpret some data from the Calendar key of a Nokia 6015. First here is the raw hex data.

```
0=0000000107D6091808000000000000000000200000000007D6091808000008000005  
9FFFFFFFFFFFFFFF002000000014000000000047006500740020006D00610074006500720069  
0061006C00200074006F0020007200690063006B
```

1=0000000307D60A010800000000000000000000002000000000007D60A01080000008000005
9FFFFFFFFFFFFFFF0020000000100000000000050006F0077006500720070006F0069006E0074
007300200064006F006E0065

Here I've highlighted some patterns that appear in both sub keys

0=0000000107D609180800000000000000000000002000000000007D60918080000008000005
9FFFFFFFFFFFFFFF002000000014000000000047006500740020006D00610074006500720069
0061006C00200074006F0020007200690063006B

1=0000000307D60A010800000000000000000000002000000000007D60A01080000008000005
9FFFFFFFFFFFFFFF002000000010000000000050006F0077006500720070006F0069006E0074
007300200064006F006E0065

And now I begin to separate the data up for interpretation

0=
00000001
07D60918080000
000000000000000000200000000000
07D60918080000
0080000059FFFFFFFFFFFFFF0020
000000140000000000
47006500740020006D006100740065007200690061006C00200074006F002000720069006
3006B

1=
00000003
07D60A01080000
000000000000000000200000000000
07D60A01080000
0080000059FFFFFFFFFFFFFF0020
000000100000000000
50006F0077006500720070006F0069006E0074007300200064006F006E0065

Interpreted, we have two reminders in the calendar.

0=
9/24/2006 8:00:00 AM Get material to Rick

1=
10/1/2006 8:00:00 AM Powerpoints Done

Lastly, here is a breakdown of the areas

```
1=  
00000003  
07D60A01080000 Time Stamp  
0000000000000000200000000000  
07D60A01080000  
0080000059FFFFFFFF0020 Two Bytes indicating type  
00000010000000000000 Number of Bytes in the entry  
50006F0077006500720070006F0069006E0074007300200064006F006E0065  
Entry(Unicode)
```

Conclusion

In Part II we looked at how to obtain a PM and an Absolute PM. We also looked some research techniques and some tools for interpreting the data. Lastly, we walked through two calendar entries for a 6015 Nokia.

I hope that this two-part series on hex dumping helps you in your quest to acquire and understand the data from the memory of a cell phone. Please send comments and suggestions to me either linuxchimp at gmail dot com or on my website, www.mobile-examiner.com.

Acknowledgements

Once again, I wish to thank Det. Brian Roach of the Kansas City Police Department and all my good friends at phone-forensics.com.

Without your support, encouragement and expertise an undertaking like this wouldn't be possible.

Cogitationis poenam nemo patitur