Modifying/creating Advanced Patches

Started by H3x, Aug 07 2007 01:27 PM

I would like to encourage forum big ones to help, sharing their knowledge. There are LOTS of Russian sites containing stuff like that and I would like to make a small step forward placing some elements here at SE-NSE for people who would like to look at patching using a little bigger arsenal than just HEX editor...

Original tutorial was published by **Joker XT** at JokerXT.3dn.ru, I'm only translating it (adding some hints) for further discussion at SE-NSE. If there are some unclear translating issues here is the source in Russian:

JokerXT.3dn.ru.htm **247.12K**

Credits go to:
BLiZZARD777 for starting THIS THREAD, which enabled finding Joker's sources;
redguy for translating a huge part of this.
I think that thread wasn't clear enough, so I thought it would be a good idea to make things more "friendly" creating this one...

Things marked with:(??????????????????) are fragments that might not been translated correctly though understanding the meaning is possible from the context.

Porting patches using their sources

© Joker XT (2007)
E-mail: Hitman35@rambler.ru
ICQ: 340-884-365
Website: JokerXT.3dn.ru visit, I would be pleased to see you on my site

To start with, we need to download necessary apps like **IDA PRO**, **Smelter**, **FASMARM** and **two .idc** files to apply patches in IDA.

FASMARM - http://jokerxt.3dn.ru/files/armpc.rar
IDC scripts - http://jokerxt.3dn.ru/files/idc.rar
Smelter - http://avkiev.kiev.u...ter/Smelter.rar (right klick and "Save As…")*

*(**H3ct0R**): Smelter needed **Richtx32.ocx** file to run on my PC. You can download it from the net,
copy it to **Windows\system32** folder, then register: Start -> Run -> regsvr32 \windows\system32\Richtx32.ocx -> OK.

We will use "**Changing radiostation with side buttons**" patch as a example. Porting it from W800 SW-R1BC002 to K750 SW-R1CA021.

```
;W800 SW-R1BC002
;Переключение радиостанций боковыми кнопками
;"Громкость +" - +1 станция из списка
;"Volume +" - +1 station from the list
;"Громкость -" - -1 станция из списка
;"Volume -" - -1 station from the list
;(c) Joker XT
;(r) IronMaster
+44000000
27785c: 51AB1245 D14A1D45
277864: B1AB1245 D74A1D45
cdd094: A1D0CD44 E54A1D45
11d4ad0: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFB5002101E0FFB50121064F386856F7
11d4ae0: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF 9FFBFFBDFFB55568024E3560024EB047
11d4af0: FFFFFFFFFFFFFFFFFFFFFFFF FFBDFFFF9081034CA1D0CD44
```

Now we have to convert both .mbn MAINs to .raw format (using GExtract for example).

DISASSEMBLING

Open W800_R1BC002.raw in IDA
In first window choose Processor type: ARM processors: ARM710a
Click on Set button and OK.
In second window: ROM start address: 0x44020000 and Loading address: 0x44020000.*
Change both ROM size and Loading size from 0x011B4550 to 0x12B4550 for example.**
Click OK.

(H3ct0R)

***44020000 is base value. It's different across phone models, so you'll have to use this to choose a compatible one:**

jagheterfredrik, on 2007-04-21 14:06, said:

```
Flash ID 0x200D (ST Microelectronics), included in K750, W800, D750 etc, have base +44020000.
Flash ID 0x2019 (ST Microelectronics), included in W810, Z550 etc, have base +44140000.
Flash ID 0x897E (Intel), included in W810, Z550, DB2020 etc, have base +44140000.
Flash ID 0x890D (Intel), included in DB2010, have base +440A0000.
Flash ID 0x890D (Intel), included in DB2000, have base +200A0000.
```

In our example everything is fine as our phones are K750 and W800.

**It's done to increase the disassembling range. Thanks to this we can see the regions of firmware containing free space, where patch bodies are placed. These values are different in different firmwares and 0x011B4550 is the value appropriate for W800 R1BC002 firmware only. For W810i R4EA031 for example the size should be changed like this: 0x01574985 -> 0x01674985.

When '**Generating list of strings**' window appears, close it.

Go to Options -> General and set 'Number of opcode bytes' field to «4»
Go to Options -> general -> analysis -> processor specific analysis options -> disable pointer referencing and check this option. Then click on: Options -> general -> analysis -> reanalyze program.

The next step: Options -> Setup data types… and uncheck 1 byte and 2 word then **OK**.

Open K750_R1CA021.raw in **IDA**, and do the same actions mentioned above...

Go to **IDA** where W800_R1BC002.raw is open.
Go to File -> IDC file… and choose ApplyPatch.idc (from the downloaded pack).
In the opened window choose our patch "Changing radiostation with side buttons.vkp" and on 'Patch Now' question answer 'Yes'.
That's all, the firmware is patched...

Now, look at the patch. There is a first entrypoint at: **27785c: 51AB1245 D14A1D45**. We have to go to that address: 27785c + 44000000 = 4427785c*
(**H3ct0R**)
*Addition can be performed simply in the Windows calculator - fire it up, then choose View -> Scientific and mark 'Hex' field.
*44000000 is base taken from the patch code with + sign in front of it).

Press "G" and enter the address (4427785c), then press "O", and it MUST look like this:

```
4427785C D1 4A 1D 45 DCD unk_451D4AD1
```

Double click on unk_451D4AD1 - we jumped into the patch body... The code looks ugly as it's not Thumb (16 bit) but ARM (32 bit) view format. Move up to 451D4AD0 and press "Alt + G", choose **T** and Value 1, to set it to Thumb.
Then press "C".

Take the second and third entrypoint and perform the same operations...

Go to the address of the beginning of the patch: 11d4ad0 + 44000000 = 451d4ad0.
Look there, finding this:

```
ROM:451D4AD0; ---------------------------------------------------------------------------
ROM:451D4AD0                                                    CODE16
ROM:451D4AD0
ROM:451D4AD0                    loc_451D4AD0                    ; DATA XREF: ROM:4427785Co
ROM:451D4AD0 FF B5                                          PUSH {R0-R7,LR}
ROM:451D4AD2 00 21                                          MOV   R1, #0
ROM:451D4AD4 01 E0                                          B       loc_451D4ADA
ROM:451D4AD6; ---------------------------------------------------------------------------
ROM:451D4AD6
ROM:451D4AD6                    loc_451D4AD6                    ; DATA XREF: ROM:44277864o
ROM:451D4AD6 FF B5                                          PUSH {R0-R7,LR}
ROM:451D4AD8 01 21                                          MOV   R1, #1
ROM:451D4ADA
ROM:451D4ADA                    loc_451D4ADA                    ; CODE XREF: ROM:451D4AD4j
ROM:451D4ADA 06 4F                                          LDR   R7, dword_451D4AF4
ROM:451D4ADC 38 68                                          LDR   R0, [R7]
ROM:451D4ADE 56 F7 9F FB                                    BL      sub_4512B220
```

```
ROM:451D4AE2 FF BD                                          POP   {R0-R7,PC}
ROM:451D4AE4; ---------------------------------------------------------------------------
ROM:451D4AE4
ROM:451D4AE4                        loc_451D4AE4                    ; DATA XREF: ROM:44CDD094o
ROM:451D4AE4 FF B5                                          PUSH {R0-R7,LR}
ROM:451D4AE6 55 68                                          LDR   R5, [R2,#4]
ROM:451D4AE8 02 4E                                          LDR   R6, dword_451D4AF4
ROM:451D4AEA 35 60                                          STR   R5, [R6]
ROM:451D4AEC 02 4E                                          LDR   R6, off_451D4AF8
ROM:451D4AEE B0 47                                          BLX   R6
ROM:451D4AF0 FF BD                                          POP   {R0-R7,PC}
ROM:451D4AF0; ---------------------------------------------------------------------------
ROM:451D4AF2 FF                                            DCB 0xFF
ROM:451D4AF3 FF                                            DCB 0xFF
ROM:451D4AF4 90 81 03 4C dword_451D4AF4  DCD 0x4C038190; DATA XREF: ROM:loc_451D4ADAr
ROM:451D4AF4                                                    ; ROM:451D4AE8r
ROM:451D4AF8 A1 D0 CD 44 off_451D4AF8   DCD loc_44CDD0A0+1; DATA XREF: ROM:451D4AECr
```

Beautiful! :D

Now we have to make the source *.asm file:

Again go to 451D4AD0. Once moved, choose Edit -> Begin selection, and mark our entire code. Then go to File -> Produce file -> Create ASM file, and give a name to our *.asm file - "rad" for example. Open it, seeing:

```
;
; ЙННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННННН»
; є    This file is generated by The Interactive Disassembler (IDA)     є
; є    Copyright (c) 2004 by DataRescue sa/nv, <ida@datarescue.com>           є
; є          Licensed to: Lennart Reus, 1 user, std, 07/2003                 є
; ИНННННННННННННННННННННННННННННННННННННН
```

```
                                        ; ROM:451D4AE8r
off_451D4AF8 DCD loc_44CDD0A0+1  ; DATA XREF: ROM:451D4AECr
             DCB 0xFF
```

Remove all unnecessary stuff - it should look like this now:

```
loc_451D4AD0
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, #0
                                        B            loc_451D4ADA

loc_451D4AD6
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, #1

loc_451D4ADA
                                        LDR      R7, dword_451D4AF4
                                        LDR      R0, [R7]
                                        BL         sub_4512B220
                                        POP      {R0-R7,PC}

loc_451D4AE4
                                        PUSH    {R0-R7,LR}
                                        LDR      R5, [R2,#4]
                                        LDR      R6, dword_451D4AF4
                                        STR      R5, [R6]
                                        LDR      R6, off_451D4AF8
                                        BLX      R6
                                        POP      {R0-R7,PC}
                                        DCB 0xFF
                                        DCB 0xFF
dword_451D4AF4      DCD 0x4C038190
off_451D4AF8 DCD loc_44CDD0A0+1
```

1. Add ":" signs to items that start the blocks (loc_451D4AE4 -> loc_451D4AE4:);
2. Remove all "#";
3. The data at the end of the patch must be at the address multiple of 4. So add align 4 after the code to the data;
4. Command ADR must be changed to adr (not applicable to use this patch);
5. In file all addresses, which are beyond the patch (??????????????????), loc_address, sub_ address etc. must be changed to 0x address.
6. Add: include "x.inc" at the beginning;
7. Add 'org' - that means, you must add a piece of code to the respect address later. (??????????????????)

After these changes, our file MUST look like that:

```
include "x.inc"

org 0x451D4AD0
```

```
loc_451D4AD0:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 0
                                        B            loc_451D4ADA

loc_451D4AD6:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 1

loc_451D4ADA:
                                        LDR      R7, dword_451D4AF4
                                        LDR      R0, [R7]
                                        BL         0x4512B220
                                        POP      {R0-R7,PC}

loc_451D4AE4:
                                        PUSH    {R0-R7,LR}
                                        LDR      R5, [R2,4]
                                        LDR      R6, dword_451D4AF4
                                        STR      R5, [R6]
                                        LDR      R6, off_451D4AF8
                                        BLX      R6
                                        POP      {R0-R7,PC}
align 4
dword_451D4AF4      DCD 0x4C038190
off_451D4AF8 DCD 0x44CDD0A0+1
```

The file is almost done. Now all of 0x... addresses must be copied to the beginning:

```
include "x.inc"
address1             equ 0x4512B220
address2             equ 0x44CDD0A0
address3             equ 0x451D4AD0
ram                      equ 0x4C038190

org address3
loc_451D4AD0:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 0
                                        B            loc_451D4ADA

loc_451D4AD6:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 1

loc_451D4ADA:
                                        LDR      R7, dword_451D4AF4
                                        LDR      R0, [R7]
                                        BL         address1
                                        POP      {R0-R7,PC}

loc_451D4AE4:
                                        PUSH    {R0-R7,LR}
                                        LDR      R5, [R2,4]
                                        LDR      R6, dword_451D4AF4
                                        STR      R5, [R6]
                                        LDR      R6, off_451D4AF8
```

```
                                        BLX      R6
                                        POP      {R0-R7,PC}
align 4
dword_451D4AF4      DCD ram
off_451D4AF8 DCD address2+1
```

Now add pieces:

```
First: 4427785C D1 4A 1D 45                                                 DCD
loc_451D4AD0+1
Second: 44277864 D7 4A 1D 45                                               DCD
loc_451D4AD6+1
Third: 44CDD094 E5 4A 1D 45                                                 DCD
loc_451D4AE4+1
```

First:

```
org 0x4427785C
DCD loc_451D4AD0+1
```

Second:

```
org 0x44277864
DCD loc_451D4AD6+1
```

Third:

```
org 0x44CDD094
DCD loc_451D4AE4+1
```

Add to the source and change addresses according to this scheme:

```
include "x.inc"
address1             equ 0x4512B220
address2             equ 0x44CDD0A0
patch                equ 0x451D4AD0
ram                      equ 0x4C038190
hook1                             equ 0x4427785C
hook2                             equ 0x44277864
hook3                             equ 0x44CDD094

org hook1
DCD loc_451D4AD0+1

org hook2
DCD loc_451D4AD6+1

org hook3
DCD loc_451D4AE4+1

org patch
```

```
loc_451D4AD0:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 0
                                        B            loc_451D4ADA

loc_451D4AD6:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 1

loc_451D4ADA:
                                        LDR      R7, dword_451D4AF4
                                        LDR      R0, [R7]
                                         BL       address1
                                        POP      {R0-R7,PC}

loc_451D4AE4:
                                        PUSH    {R0-R7,LR}
                                        LDR      R5, [R2,4]
                                        LDR      R6, dword_451D4AF4
                                        STR      R5, [R6]
                                        LDR      R6, off_451D4AF8
                                        BLX      R6
                                        POP      {R0-R7,PC}
align 4
dword_451D4AF4      DCD ram
off_451D4AF8 DCD address2+1
```

Now we need to find addresses for K750 SW-R1CA021.
Open **Smelter**, go to Fullflash -> Open and choose our file K750_R1CA021.raw (since Smelter doesn't see '.raw' files, choose 'all files' below).
After opening, go to Fullflash -> Load base and set it to 44020000.

Porting address1:

```
address1             equ 0x4512B220
```

Go to 0x4512B220 and move a little bit lower (because making pattern from middle of the function is better IMHO) in the W800 flashfile and create a pattern:

```
ROM:4512B24C                    loc_4512B24C                    ; CODE XREF:
sub_4512B220+20j
ROM:4512B24C 2A 7C                                              LDRB R2, [R5,#0x10]
ROM:4512B24E 10 1C                                              ADD   R0, R2, #0
ROM:4512B250 12 30                                              ADD   R0, #0x12
ROM:4512B252 14 21                                              MOV   R1, #0x14
ROM:4512B254 3B F0 5E FC                                      BL       sub_45166B14
ROM:4512B258 03 1C                                              ADD   R3, R0, #0
ROM:4512B25A 00 2A                                              CMP   R2, #0
ROM:4512B25C 0F D1                                              BNE   loc_4512B27E
ROM:4512B25E 13 23                                              MOV   R3, #0x13
ROM:4512B260 0D E0                                              B       loc_4512B27E
```

2A7C101C12301421????????031C002A0FD1 – this is a pattern in our case.

Press a big "**B**" button in the Smelter's panel, copy the pattern here and press **OK**.
We found an address, click the line with address and **F3** to copy it.
Go to the same address in K750_R1CA021.
Press "Alt+B", in String fill **b5** then Search Up...
Ready? Got an address. Go to this address-1 and press "**C**".
Compare functions, they are equal aren't they? Yes!

```
address1              equ 0x4511C92C;0x4512B220 – W800
```

Find the second address the same way:

```
address2              equ 0x44CD2F54;0x44CDD0A0 – W800
```

The address of the patch and ram must not be changed:

```
patch                 equ 0x451D4AD0
ram                       equ 0x4C038190
```

(**H3ct0R**: they have to be changed when readdressing and porting patches across different platforms like W800/W810).

Now port the "hook's":

```
hook1                                 equ 0x4427785C
```

First: undo the patch in **IDA**.
Go to File > IDC file… and choose UndoPatch.idc.
In the opened window take our patch "Changing radiostation with side buttons.vkp" and answer 'Yes' to the question.

Go to the address of the first "hook" 0x4427785C, press "D" at the same address:

```
ROM:44277850 8D 13 00 00                                        DCD 0x138D
ROM:44277854 91 D7 12 45                                        DCD 0x4512D791
ROM:44277858 BE 07 00 00                                        DCD 0x7BE
ROM:4427785C 51 AB 12 45                                        DCD 0x4512AB51
ROM:44277860 BF 07 00 00                                        DCD 0x7BF
ROM:44277864 B1 AB 12 45                                        DCD 0x4512ABB1
ROM:44277868 06 00 00 00                                        DCD 6
ROM:4427786C 7D AD 12 45                                        DCD 0x4512AD7D
ROM:44277870 00 00 00 00                                        DCD 0
ROM:44277874 00 00 00 00                                        DCD 0
```

```
ROM:44277878 46 05 41 44                                        DCD 0x44410546
ROM:4427787C 00 00 00 00                                        DCD 0
ROM:44277880 30 78 27 44                                        DCD 0x44277830
ROM:44277884 05 00 00 00                                        DCD 5
```

Making pattern:
8D130000????????BE070000????????BF070000????????06000000????????
00000000000000000????????00000000

Found:

```
hook1                                   equ 0x44278324;0x4427785C – W800
```

Find other "hooks":

```
hook2                                   equ 0x4427832C;0x44277864 – W800
hook3                                   equ 0x44CD2F48;0x44CDD094 – W800
```

Get it all together:

```
include "x.inc"
address1              equ 0x4511C92C;0x4512B220 – W800
address2              equ 0x44CD2F54;0x44CDD0A0 – W800
patch                 equ 0x451D4AD0
ram                       equ 0x4C038190
hook1                             equ 0x44278324;0x4427785C – W800
hook2                             equ 0x4427832C;0x44277864 – W800
hook3                             equ 0x44CD2F48;0x44CDD094 – W800

org hook1
DCD loc_451D4AD0+1

org hook2
DCD loc_451D4AD6+1

org hook3
DCD loc_451D4AE4+1

org patch
loc_451D4AD0:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 0
                                        B            loc_451D4ADA

loc_451D4AD6:
                                        PUSH    {R0-R7,LR}
                                        MOV      R1, 1

loc_451D4ADA:
                                        LDR      R7, dword_451D4AF4
                                        LDR      R0, [R7]
                                        BL         address1
                                        POP      {R0-R7,PC}
```

```
loc_451D4AE4:
                                        PUSH    {R0-R7,LR}
                                        LDR      R5, [R2,4]
                                        LDR      R6, dword_451D4AF4
                                        STR      R5, [R6]
                                        LDR      R6, off_451D4AF8
                                        BLX      R6
                                        POP      {R0-R7,PC}
align 4
dword_451D4AF4      DCD ram
off_451D4AF8 DCD address2+1
```

That's all the file is ready :D. It has to be assembled with **FASMARM** now.

REASSEMBLING

Start Command prompt: Start -> Run -> cmd -> OK.

(On my PC the program is in **D:\armpc2**)

D: and **Enter**
Then, **cd D:\armpc2** and **Enter**
To assemble our file write: **make rad.asm K750 R1CA021.raw 44020000**

Where:
rad.asm – the name of our file;
K750 R1CA021.raw – the name of our firmware (the one that we are porting our patch to);
44020000 – base appropriate to our phone (the one that we are porting our patch to);

Press **Enter** and see:
Can not find D:\armpc2\rad.asm.vkp
flat assembler for ARM version 1.66
2 passes, 104 bytes.

Or something similar...
Go to the folder **D:\armpc2** and take our patch **rad.asm.vkp**

That's all!!!
Wrote all this in a simple manner, hope you understood it.

© Joker XT (2007)
----------THE END----------