Homework Chap. 9

1) (60 points) Hacking (a game) ROM

Question 1.1) Learn how to hack a game ROM from this link http://www.nintendoage.com/forum/messageview.cfm?catid=22&threadid=19733

Answer 1.1

Unfortunately the link in the question does not exist anymore. I have learnt how to hack a game ROM thanks to the site https://www.romhacking.net . This site has a collection of tools which can be downloaded. In this site there is also a forum, a rich community, a database of utilities, an educational section.

Question 1.2) Change 2 PLAYER GAME in menu to 2 Your Name GAME, e.g., I change the 2 PLAYER GAME to 2 EKARAT GAME. Capture and paste your change.
**** You can download the target game rom (Super Mario Adventure (SMB1 Hack).nes) at the course webpage.***

Answer 1.2

Unfortunately I did not find the course webpage where to download rom (Super Mario Adventure (SMB1 Hack).nes). I instead downloaded the original version Super Mario Bros ,you can download here (https://www.downloadroms.io/nintendo-rom-vs-super-mario-bros-vs-a1/) and I have made the changes requested in the question.

In order to do this change I have used two main tools :

1) *OpenSearch-0 (you cand download here https://www.romhacking.net/utilities/602/)*

*This tool is a more complete version , command line, of Search Relative 2 . If you type the string you are looking for in the ROM file, it will create an association table with the hexadecimal value and its character translation. In fact the characters in a ROM file are not represented in ASCII format. Therefore I put the Opensearch-0 executable in the same dos directory of the *.nes fil. It should be noted that a NES file is a game ROM created from an NES (Nintendo Entertainment System) video game. It contains the same data as the original NES cartridge and can be opened and played on a Mac or PC using an NES emulator.*
From the DOS command line I changed the actual position to the tool directory and I issued the following command:

```
C:\Users\Admin\Desktop\supermario\OpenSearch-0>opensearch-win32-0.1.exe "Super Mario Bros. (World).nes" *player -mktbl -autocaps G
```

The options are explained on the website and (-mktbl) means that the Table file output based on the results of a relative search.

*(note that *player stands for the string I am looking for into the super Mario nes file)*

The output of this command is the following:

```
OpenSearch 0.1  Copyright (C) 2009 mcpancakes
Input:  * 70 6C 61 79 65 72
ROM:   24 19 15 0A 22 0E 1B
Offset: 0x9FC7 (40903)
Creating table... Done!
```

This command created, as a result, the following table (the table is output in the command directory and it is displayed below) :

```
0A=a
0B=b
0C=c
0D=d
0E=e
0F=f
10=g
11=h
12=i
13=j
14=k
15=l
16=m
17=n
18=o
19=p
1A=q
1B=r
1C=s
1D=t
1E=u
1F=v
20=w
21=x
22=y
23=z
1E=A
1F=B
20=C
21=D
```

```
22=E
23=F
24=G
25=H
26=I
27=J
28=K
29=L
2A=M
2B=N
2C=O
2D=P
2E=Q
2F=R
30=S
31=T
32=U
33=V
34=W
35=X
36=Y
37=Z
```

The term “Player” I am looking for is therefore translated in the following hexadecimal value:

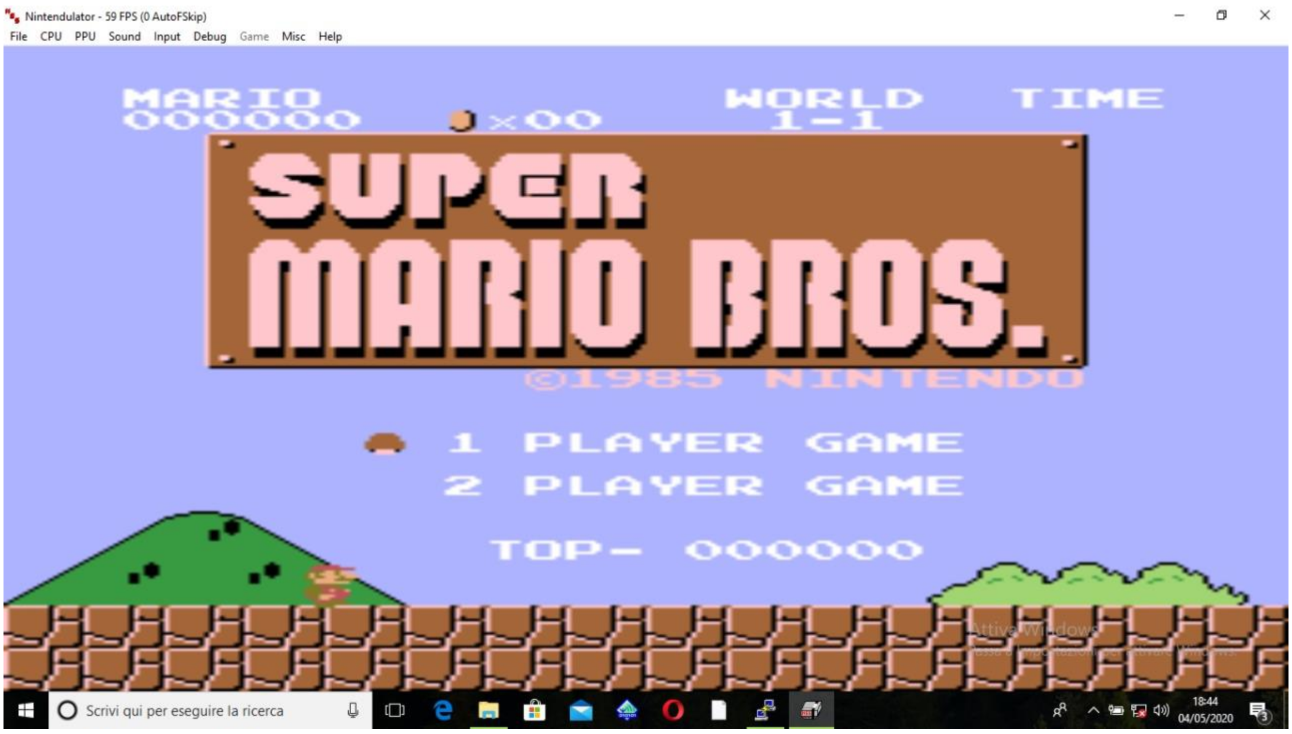
24 19 15 0A 22 0E 1B

And the term “Ekarat”, which I want to substitute to the term “player”, can be coded in :

0E 14 0A 1B 0A 1D.

(according to the table listed above)

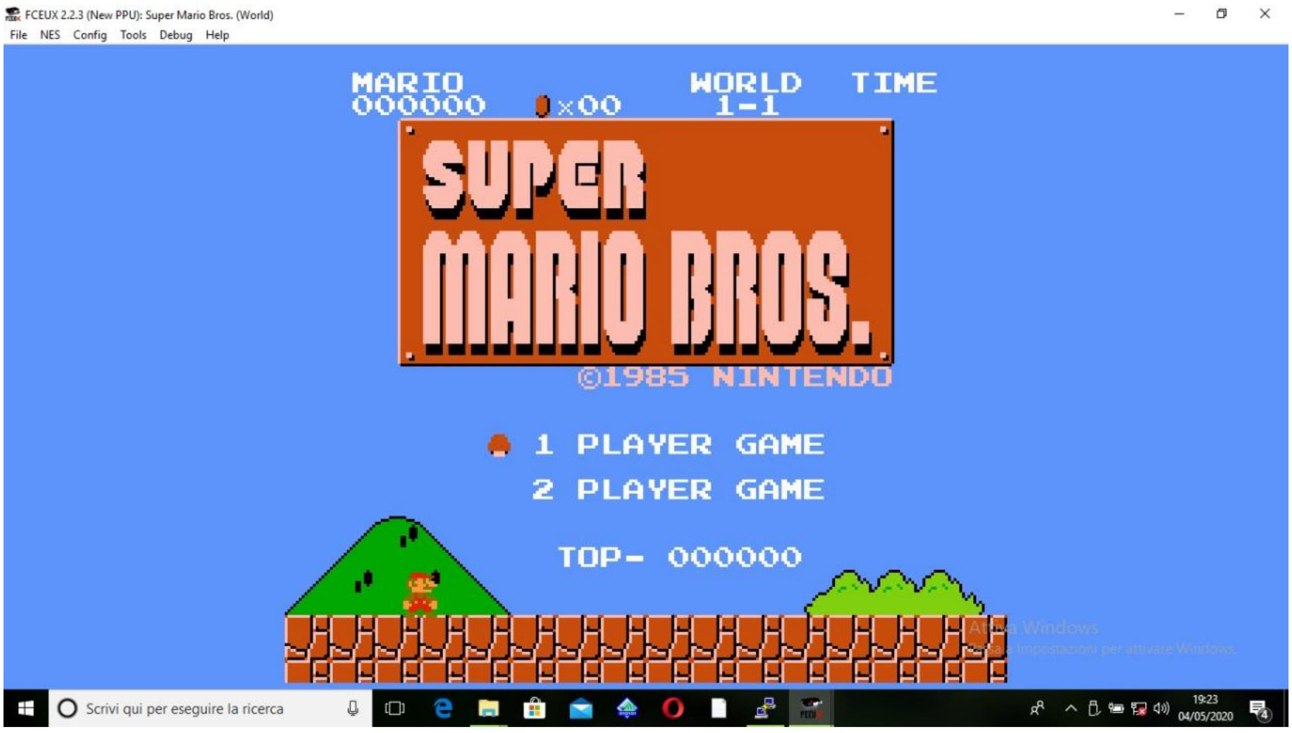
In order to make the modification to the nes code I will need another tool : FCEUX 2.2.3. This tool can be downloaded from the romahacking.net website, and has the following graphical interface (after opening the nes super Mario file).

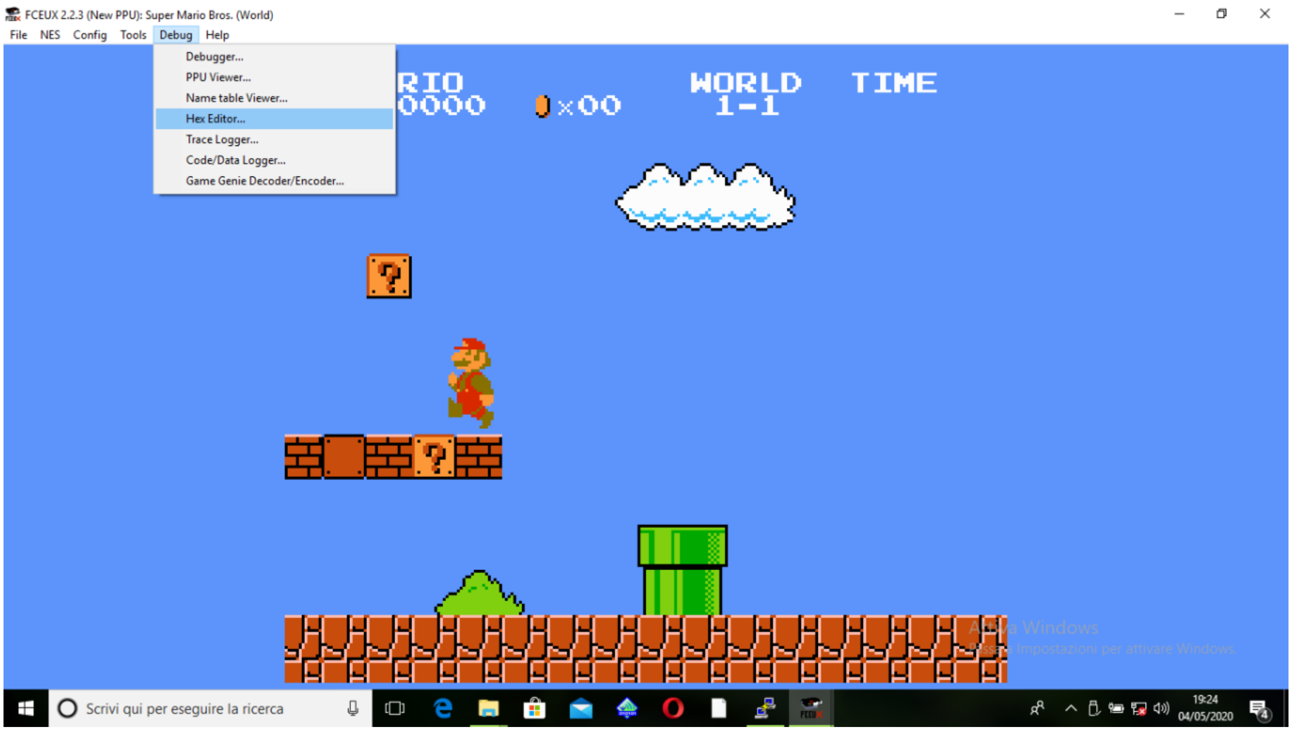

Nintendulator - 59 FPS (0 AutoFSkip)
File CPU PPU Sound Input Debug Game Misc Help
MARIO 000000 ×00 WORLD 1-1 TIME
SUPER MARIO BROS.
©1985 NINTENDO
1 PLAYER GAME
2 PLAYER GAME
TOP- 000000
Scrivi qui per eseguire la ricerca
18:44 04/05/2020

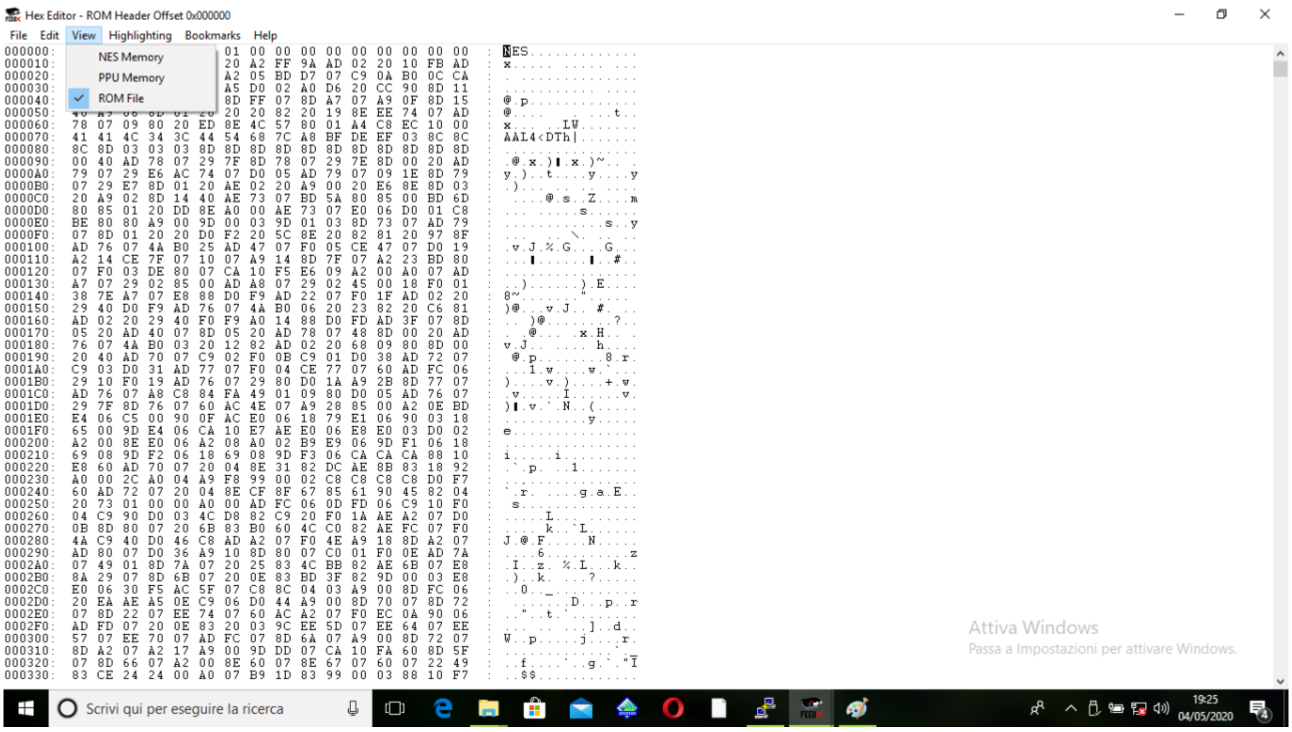
From the menu I click on debug, then I click on hex editor and I enter the hexcode **24 19 15 0A 22 0E 1B.** I selected nes memory view and I clicked on find the hexadecimal string :

24 19 15 0A 22 0E 1B

which corresponds to the player term.
See the screenshots below:

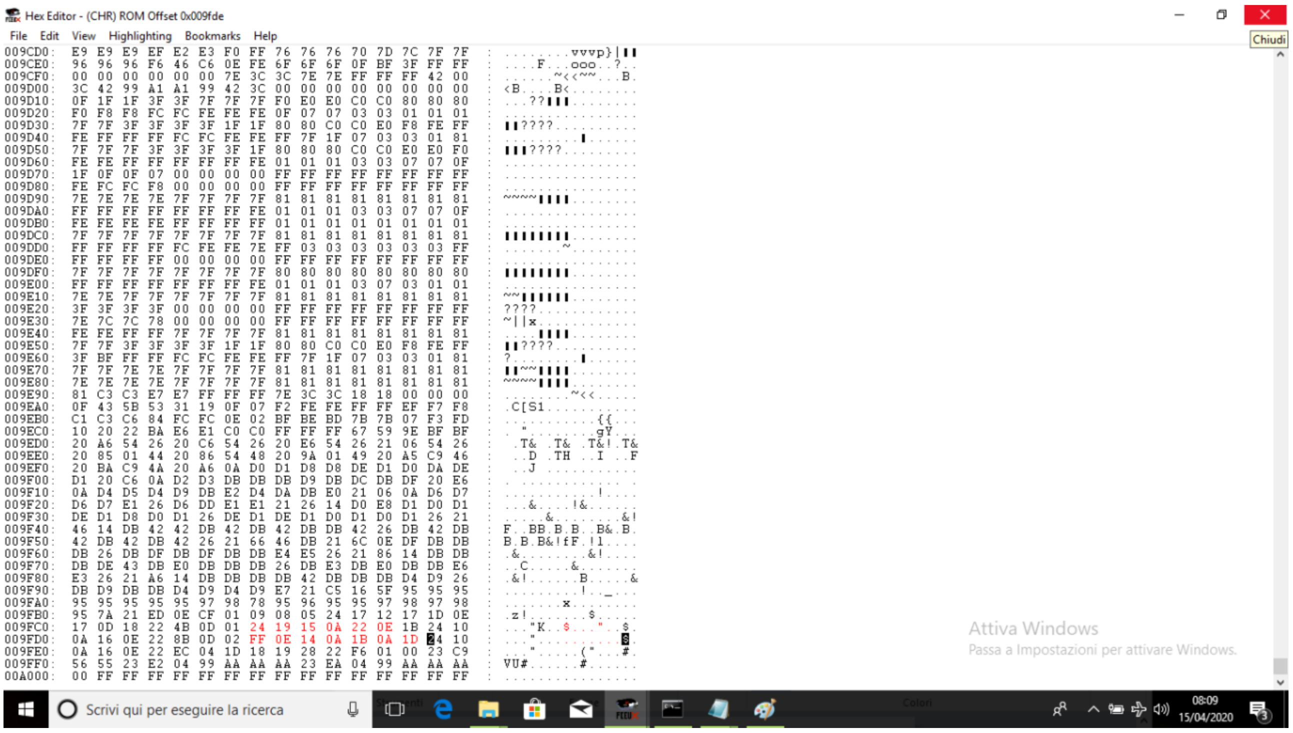

FCEUX 2.2.3 (New PPU): Super Mario Bros. (World)
File NES Config Tools Debug Help
MARIO 000000 ×00 WORLD 1-1 TIME
SUPER MARIO BROS.
©1985 NINTENDO
1 PLAYER GAME
2 PLAYER GAME
TOP- 000000
Scrivi qui per eseguire la ricerca
19:23 04/05/2020


FCEUX 2.2.3 (New PPU): Super Mario Bros. (World)
File NES Config Tools Debug Help
Debugger...
PPU Viewer...
Name table Viewer...
Hex Editor...
Trace Logger...
Code/Data Logger...
Game Genie Decoder/Encoder...
RIO 0000
×00
WORLD 1-1
TIME
Scrivi qui per eseguire la ricerca
19:24 04/05/2020


Hex Editor - ROM Header Offset 0x000000
File Edit View Highlighting Bookmarks Help
NES Memory
PPU Memory
✓ ROM File
Scrivi qui per eseguire la ricerca
19:25 04/05/2020

```
000000:                   01 00 00 00 00 00 00 00 00 00  : NES.............
000010:                   20 A2 FF 9A AD 02 20 10 FB AD  : x...............
000020:                   A2 05 BD D7 07 C9 0A B0 0C CA  : ................
000030:                   A5 D0 02 A0 D6 20 CC 90 8D 11  : ................
000040:                   8D FF 07 8D A7 07 A9 0F 8D 15  : @.p.............
000050: 40 A9 06 8D 01 20 20 20 82 20 19 8E EE 74 07 AD  : @............t..
000060: 78 07 09 80 20 ED 8E 4C 57 80 01 A4 C8 EC 10 00  : x......LW.......
000070: 41 41 4C 34 3C 44 54 68 7C A8 BF DE EF 03 8C 8C  : AAL4<DTh|.......
000080: 8C 8D 03 03 03 8D 8D 8D 8D 8D 8D 8D 8D 8D 8D 8D  : ................
000090: 00 40 AD 78 07 29 7F 8D 78 07 29 7E 8D 00 20 AD  : .@.x.)▌.x.)~....
0000A0: 79 07 29 E6 AC 74 07 D0 05 AD 79 07 09 1E 8D 79  : y.)..t....y....y
0000B0: 07 29 E7 8D 01 20 AE 02 20 A9 00 20 E6 8E 8D 03  : .)..............
0000C0: 20 A9 02 8D 14 40 AE 73 07 BD 5A 80 85 00 BD 6D  : .....@.s..Z....m
0000D0: 80 85 01 20 DD 8E A0 00 AE 73 07 E0 06 D0 01 C8  : .........s......
0000E0: BE 80 80 A9 00 9D 00 03 9D 01 03 8D 73 07 AD 79  : ............s..y
0000F0: 07 8D 01 20 20 D0 F2 20 5C 8E 20 82 81 20 97 8F  : ........\.......
000100: AD 76 07 4A B0 25 AD 47 07 F0 05 CE 47 07 D0 19  : .v.J.%.G....G...
000110: A2 14 CE 7F 07 10 07 A9 14 8D 7F 07 A2 23 BD 80  : ...▌.......▌..#..
000120: 07 F0 03 DE 80 07 CA 10 F5 E6 09 A2 00 A0 07 AD  : ................
000130: A7 07 29 02 85 00 AD A8 07 29 02 45 00 18 F0 01  : ..)......).E....
000140: 38 7E A7 07 E8 88 D0 F9 AD 22 07 F0 1F AD 02 20  : 8~.......".......
000150: 29 40 D0 F9 AD 76 07 4A B0 06 20 23 82 20 C6 81  : )@...v.J...#....
000160: AD 02 20 29 40 F0 F9 A0 14 88 D0 FD AD 3F 07 8D  : ...)@........?..
000170: 05 20 AD 40 07 8D 05 20 AD 78 07 48 8D 00 20 AD  : ...@......x.H...
000180: 76 07 4A B0 03 20 12 82 AD 02 20 68 09 80 8D 00  : v.J.........h...
000190: 20 40 AD 70 07 C9 02 F0 0B C9 01 D0 38 AD 72 07  :  @.p.......8.r.
0001A0: C9 03 D0 31 AD 77 07 F0 04 CE 77 07 60 AD FC 06  : ...1.w....w.`...
0001B0: 29 10 F0 19 AD 76 07 29 80 D0 1A A9 2B 8D 77 07  : )....v.)....+.w.
0001C0: AD 76 07 A8 C8 84 FA 49 01 09 80 D0 05 AD 76 07  : .v.....I......v.
0001D0: 29 7F 8D 76 07 60 AC 4E 07 A9 28 85 00 A2 0E BD  : )▌.v.`.N..(.....
0001E0: E4 06 C5 00 90 0F AC E0 06 18 79 E1 06 90 03 18  : ..........y.....
0001F0: 65 00 9D E4 06 CA 10 E7 AE E0 06 E8 E0 03 D0 02  : e...............
000200: A2 00 8E E0 06 A2 08 A0 02 B9 E9 06 9D F1 06 18  : ................
000210: 69 08 9D F2 06 18 69 08 9D F3 06 CA CA CA 88 10  : i.....i.........
000220: E8 60 AD 70 07 20 04 8E 31 82 DC AE 8B 83 18 92  : .`.p....1.......
000230: A0 00 2C A0 04 A9 F8 99 00 02 C8 C8 C8 C8 D0 F7  : ..,.............
000240: 60 AD 72 07 20 04 8E CF 8F 67 85 61 90 45 82 04  : `.r......g.a.E..
000250: 20 73 01 00 00 A0 00 AD FC 06 0D FD 06 C9 10 F0  :  s..............
000260: 04 C9 90 D0 03 4C D8 82 C9 20 F0 1A AE A2 07 D0  : .....L..........
000270: 0B 8D 80 07 20 6B 83 B0 60 4C C0 82 AE FC 07 F0  : .....k..`L......
000280: 4A C9 40 D0 46 C8 AD A2 07 F0 4E A9 18 8D A2 07  : J.@.F.....N.....
000290: AD 80 07 D0 36 A9 10 8D 80 07 C0 01 F0 0E AD 7A  : ....6..........z
0002A0: 07 49 01 8D 7A 07 20 25 83 4C BB 82 AE 6B 07 E8  : .I..z..%.L..k...
0002B0: 8A 29 07 8D 6B 07 20 0E 83 BD 3F 82 9D 00 03 E8  : .)..k....?......
0002C0: E0 06 30 F5 AC 5F 07 C8 8C 04 03 A9 00 8D FC 06  : ..0.._..........
0002D0: 20 EA AE A5 0E C9 06 D0 44 A9 00 8D 70 07 8D 72  : ........D..p..r
0002E0: 07 8D 22 07 EE 74 07 60 AC A2 07 F0 EC 0A 90 06  : .."..t.`........
0002F0: AD FD 07 20 0E 83 20 03 9C EE 5D 07 EE 64 07 EE  : ...........]..d.
000300: 57 07 EE 70 07 AD FC 07 8D 6A 07 A9 00 8D 72 07  : W..p.....j....r.
000310: 8D A2 07 A2 17 A9 00 9D DD 07 CA 10 FA 60 8D 5F  : .............`._
000320: 07 8D 66 07 A2 00 8E 60 07 8E 67 07 60 07 22 49  : ..f....`..g.`."I
000330: 83 CE 24 24 00 A0 07 B9 1D 83 99 00 03 88 10 F7  : ..$$............
```

The first occurrence is not my target string, since I wish to modify the second "player" occurrence. So I found the hex value in the row after the first occurrence , as you can see in the following screen dump (the values in red are the values that I edited):



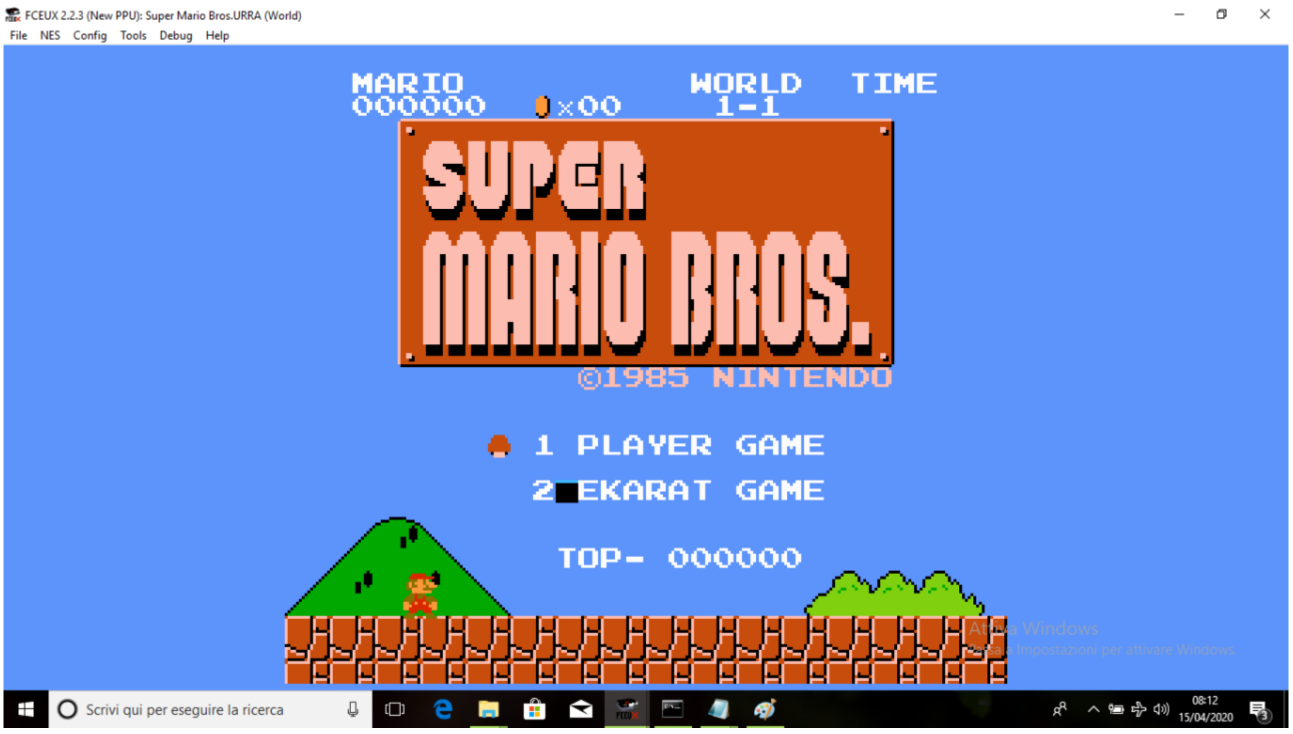
Hex Editor - (CHR) ROM Offset 0x009fde

File Edit View Highlighting Bookmarks Help

```
009CD0:  E9 E9 E9 EF E2 E3 F0 FF 76 76 76 70 7D 7C 7F 7F  : ........vvvp}|▌▌
009CE0:  96 96 96 F6 46 C6 0E FE 6F 6F 6F 0F BF 3F FF FF  : ....F...ooo..?..
009CF0:  00 00 00 00 00 00 7E 3C 3C 7E 7E FF FF FF 42 00  : ......~<<~~...B.
009D00:  3C 42 99 A1 A1 99 42 3C 00 00 00 00 00 00 00 00  : <B....B<........
009D10:  0F 1F 1F 3F 3F 7F 7F 7F F0 E0 E0 C0 C0 80 80 80  : ...??▌▌▌........
009D20:  F0 F8 F8 FC FC FE FE FE 0F 07 07 03 03 01 01 01  : ................
009D30:  7F 7F 3F 3F 3F 3F 1F 1F 80 80 C0 C0 E0 F8 FE FF  : ▌▌????..........
009D40:  FE FF FF FF FC FC FE FE FF 7F 1F 07 03 03 01 81  : .........▌......
009D50:  7F 7F 7F 3F 3F 3F 3F 1F 80 80 80 C0 C0 E0 E0 F0  : ▌▌▌????.........
009D60:  FE FE FF FF FF FF FF FE 01 01 01 03 03 07 07 0F  : ................
009D70:  1F 0F 0F 07 00 00 00 00 FF FF FF FF FF FF FF FF  : ................
009D80:  FE FC FC F8 00 00 00 00 FF FF FF FF FF FF FF FF  : ................
009D90:  7E 7E 7E 7E 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ~~~~▌▌▌▌........
009DA0:  FF FF FF FF FF FF FF FE 01 01 01 03 03 07 07 0F  : ................
009DB0:  FE FE FE FE FF FF FF FF 01 01 01 01 01 01 01 01  : ................
009DC0:  7F 7F 7F 7F 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ▌▌▌▌▌▌▌▌........
009DD0:  FF FF FF FF FC FE FE 7E FF 03 03 03 03 03 03 FF  : .......~........
009DE0:  FF FF FF FF 00 00 00 00 FF FF FF FF FF FF FF FF  : ................
009DF0:  7F 7F 7F 7F 7F 7F 7F 7F 80 80 80 80 80 80 80 80  : ▌▌▌▌▌▌▌▌........
009E00:  FF FF FF FF FF FF FF FE 01 01 01 03 07 03 01 01  : ................
009E10:  7E 7E 7F 7F 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ~~▌▌▌▌▌▌........
009E20:  3F 3F 3F 3F 00 00 00 00 FF FF FF FF FF FF FF FF  : ????............
009E30:  7E 7C 7C 78 00 00 00 00 FF FF FF FF FF FF FF FF  : ~||x............
009E40:  FE FE FF FF 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ....▌▌▌▌........
009E50:  7F 7F 3F 3F 3F 3F 1F 1F 80 80 C0 C0 E0 F8 FE FF  : ▌▌????..........
009E60:  3F BF FF FF FC FC FE FE FF 7F 1F 07 03 03 01 81  : ?........▌......
009E70:  7F 7F 7E 7E 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ▌▌~~▌▌▌▌........
009E80:  7E 7E 7E 7E 7F 7F 7F 7F 81 81 81 81 81 81 81 81  : ~~~~▌▌▌▌........
009E90:  81 C3 C3 E7 E7 FF FF FF 7E 3C 3C 18 18 00 00 00  : ........~<<.....
009EA0:  0F 43 5B 53 31 19 0F 07 F2 FE FE FF FF EF F7 F8  : .C[S1...........
009EB0:  C1 C3 C6 84 FC FC 0E 02 BF BE BD 7B 7B 07 F3 FD  : ...........{{...
009EC0:  10 20 22 BA E6 E1 C0 C0 FF FF FF 67 59 9E BF BF  : . "........gY...
009ED0:  20 A6 54 26 20 C6 54 26 20 E6 54 26 21 06 54 26  :  .T& .T& .T&!.T&
009EE0:  20 85 01 44 20 86 54 48 20 9A 01 49 20 A5 C9 46  :  ..D .TH ..I ..F
009EF0:  20 BA C9 4A 20 A6 0A D0 D1 D8 D8 DE D1 D0 DA DE  :  ..J ...........
009F00:  D1 20 C6 0A D2 D3 DB DB DB D9 DB DC DB DF 20 E6  : . ............ .
009F10:  0A D4 D5 D4 D9 DB E2 D4 DA DB E0 21 06 0A D6 D7  : ...........!....
009F20:  D6 D7 E1 26 D6 DD E1 E1 21 26 14 D0 E8 D1 D0 D1  : ...&....!&......
009F30:  DE D1 D8 D0 D1 26 DE D1 DE D1 D0 D1 D0 D1 26 21  : .....&........&!
009F40:  46 14 DB 42 42 DB 42 DB 42 DB DB 42 26 DB 42 DB  : F..BB.B.B..B&.B.
009F50:  42 DB 42 DB 42 26 21 66 46 DB 21 6C 0E DF DB DB  : B.B.B&!fF.!l....
009F60:  DB 26 DB DF DB DF DB DB E4 E5 26 21 86 14 DB DB  : .&........&!....
009F70:  DB DE 43 DB E0 DB DB DB 26 DB E3 DB E0 DB DB E6  : ..C.....&.......
009F80:  E3 26 21 A6 14 DB DB DB DB 42 DB DB DB D4 D9 26  : .&!......B.....&
009F90:  DB D9 DB DB D4 D9 D4 D9 E7 21 C5 16 5F 95 95 95  : .........!._....
009FA0:  95 95 95 95 95 97 98 78 95 96 95 95 97 98 97 98  : .......x........
009FB0:  95 7A 21 ED 0E CF 01 09 08 05 24 17 12 17 1D 0E  : .z!.......$.....
009FC0:  17 0D 18 22 4B 0D 01 24 19 15 0A 22 0E 1B 24 10  : ..."K..$..."..$.
009FD0:  0A 16 0E 22 8B 0D 02 FF 0E 14 0A 1B 0A 1D 24 10  : ..."..........$.
009FE0:  0A 16 0E 22 EC 04 1D 18 19 28 22 F6 01 00 23 C9  : ..."......("..#.
009FF0:  56 55 23 E2 04 99 AA AA AA 23 EA 04 99 AA AA AA  : VU#......#......
00A000:  00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  : ................
```

Chiudi
Scrivi qui per eseguire la ricerca
Attiva Windows
Passa a Impostazioni per attivare Windows.
08:09
15/04/2020

After this modification , I saved this new Rom version with a new name, and uploaded it again in the emulator. The result was what I wanted (see the following screen dump):



FCEUX 2.2.3 (New PPU): Super Mario Bros.URRA (World)

Question 2) (20 points) Use your Hex editor to modify any programs you want, and tell us
2.1 What is the target program?
2.2 What is your modification? Show the captured screen of the result.

This question can be considered a duplicate version of what has been showed in Answers 1.1 and 1.2 where the target program is the hex file (nes) and the modification is the string which I have substituted in the hex file.

Question 3) (20 points) Do a research. What are the difference between PlayStation4 and PlayState3 in terms of hardware security aspects?

PS3 General Info.

The PlayStation 3, registered with the PLAYSTATION 3 ™ brand and abbreviated with the abbreviation PS3 ™, is a video game console produced by Sony Computer Entertainment, equipped with various multimedia functions in addition to those of video game entertainment.
The console is available from November 11, 2006 in Japan, November 17, 2006 in North America and March 23, 2007 in Europe, the Middle East, Australia, Asia, Latin America and North Africa.

PS3 Architecture

(Source: Wikipedia https://en.wikipedia.org/wiki/PlayStation_3_technical_specifications)

Central processing unit

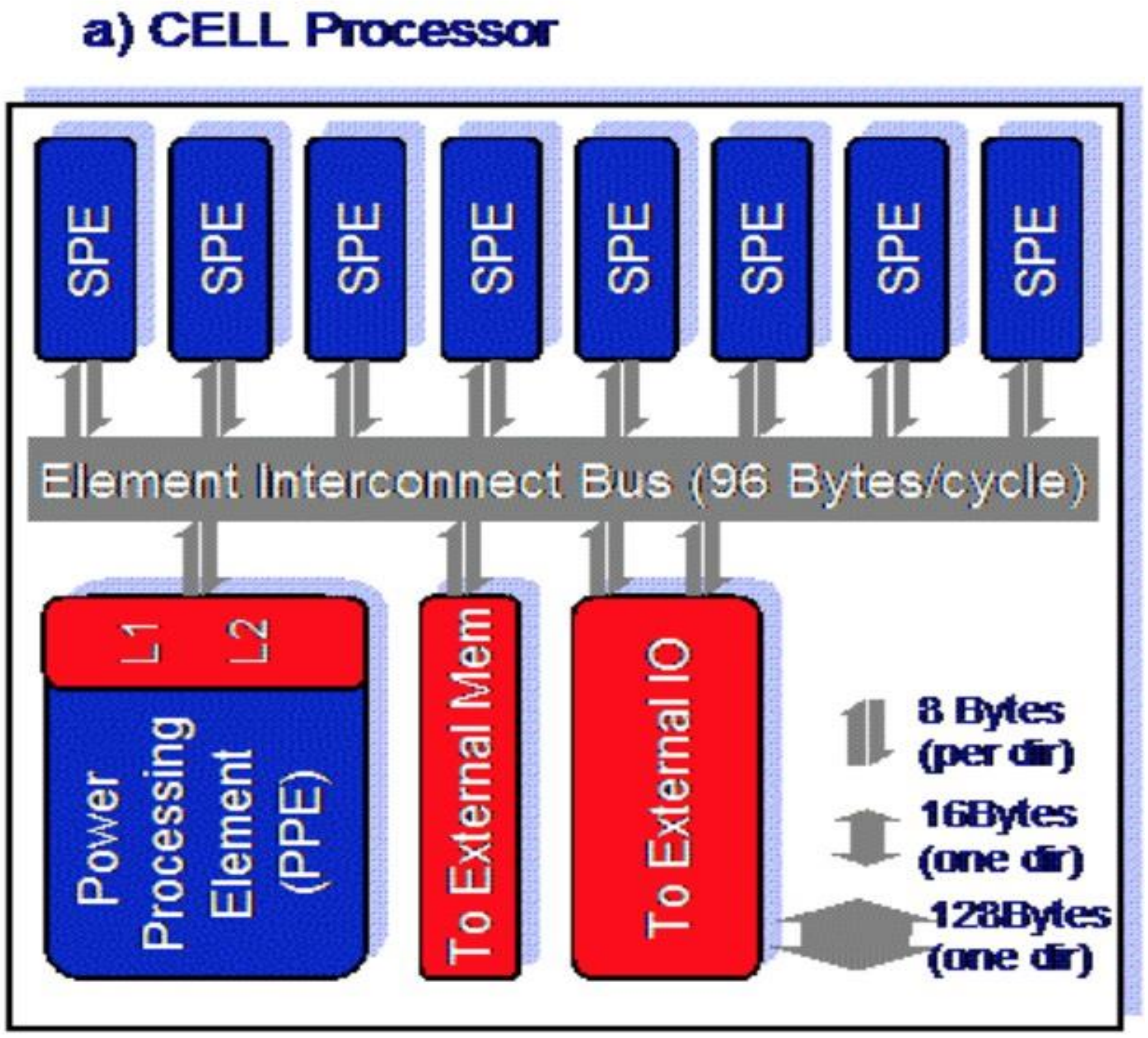
Cell (microprocessor)

PS3 CPU-"Cell Broadband Engine"

The PS3 uses the Cell microprocessor, which is made up of one 3.2 GHz PowerPC-based "Power Processing Element" (PPE) and six accessible Synergistic Processing Elements (SPEs). A seventh runs in a special mode and is dedicated to aspects of the OS and security, and an eighth is a spare to improve production yields. PlayStation 3's Cell CPU achieves a theoretical maximum of 230.4 GFLOPS in single precision floating point operations and up to 15 GFLOPS double precision[1]

The PS3 has 256 MB of Rambus XDR DRAM, clocked at CPU die speed. The PPE has 64 KB L1 cache and 512 KB L2 cache, while the SPEs have 2 MB local memory

(256 KB per SPE), connected by the Element Interconnect Bus (EIB) with up to 307.2 Gb/s bandwidth.



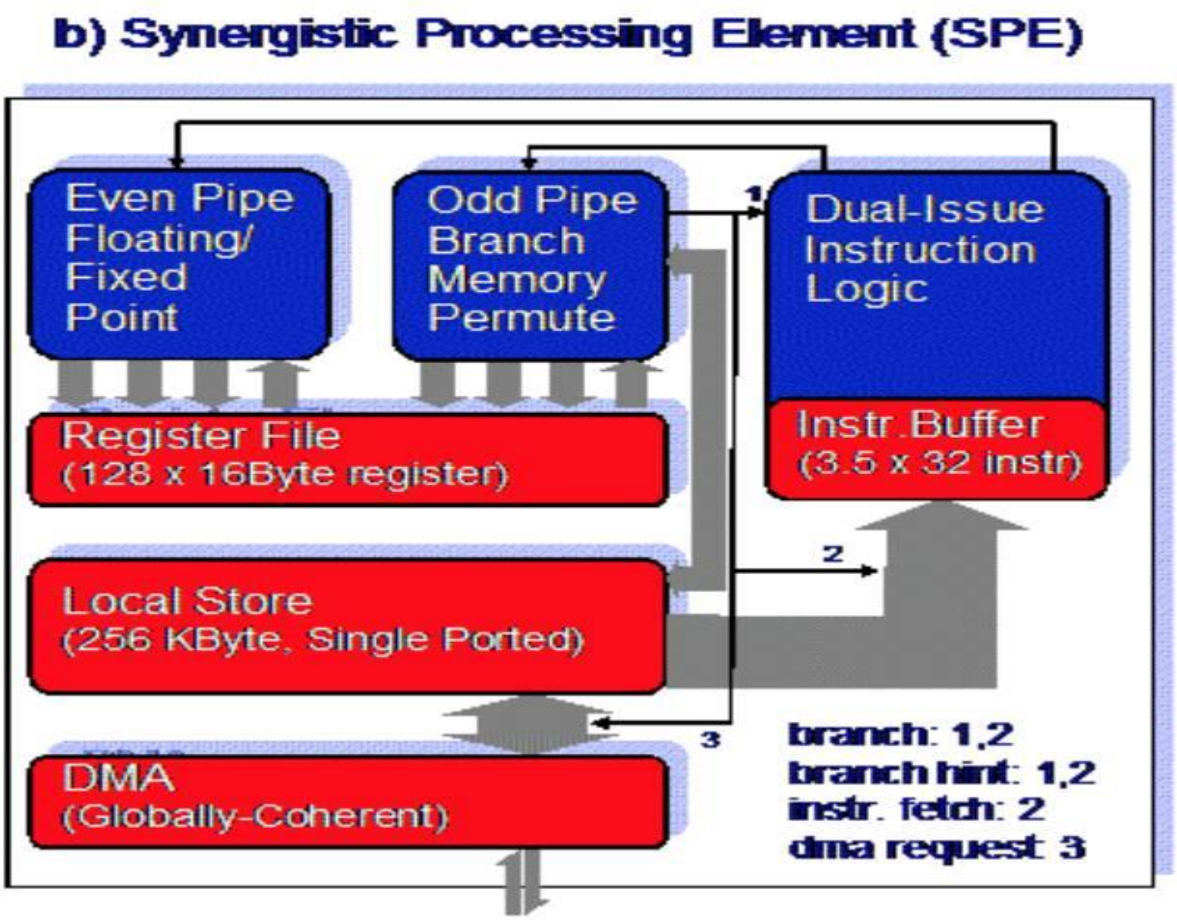

b) Synergistic Processing Element (SPE)
Even Pipe Floating/ Fixed Point
Odd Pipe Branch Memory Permute
Dual-Issue Instruction Logic
Register File (128 x 16Byte register)
Instr.Buffer (3.5 x 32 instr)
Local Store (256 KByte, Single Ported)
DMA (Globally-Coherent)
1
2
3
branch: 1,2
branch hint: 1,2
instr. fetch: 2
dma request: 3

Figure 2: SPE [10]

Graphic Processing Unit

RSX 'Reality Synthesizer'

PS3 GPU-RSX "Reality Synthesizer"

According to Nvidia, the RSX—the graphics processing unit (GPU)—is based on the NVIDIA G70 (previously known as NV47) architecture. The GPU is clocked at 500 MHz and makes use of 256 MB GDDR3 RAM clocked at 650 MHz with an effective transmission rate of 1.3 GHz.[3] The RSX has a floating-point performance of 400.4 GFLOPS.[4]

For a more detailed info on Architecture read the article in the link below.

https://www.cs.uaf.edu/2012/fall/cs441/students/sc_ps3.pdf

Hacking the PS3

(Source : Wikipedia : George Hotz https://en.wikipedia.org/wiki/George_Hotz)

In December 2009, Hotz announced his initial intentions to breach security on the Sony PlayStation 3. Five weeks later, on January 22, 2010, he announced that he had performed his first theoretical achievement. This consists of the initial read and write access to the machine's system memory as well as hypervisor level access to the machine's CPU.

On January 26, 2010, Hotz released the exploit to the public. On March 28, 2010, Sony responded by announcing their intention to release a PlayStation 3 firmware update that would remove the OtherOS feature from all models, a feature that was already absent on the newer Slim revisions of the machine.

On July 13, 2010, never having achieved any method of reading, installing, or modifying software on the PS3, Hotz posted a message on his Twitter account stating that he had abandoned his efforts of trying to crack the PS3 any further due to the system security's extreme difficulty.

On December 29, 2010, notable hacking group ***fail0verflow***, known for the reverse engineering of security models found in consumer electronics devices, performed an academic presentation at the 27th Chaos Communications Congress technical conference, of their accomplishments with the PlayStation 3. **They presented the methods they'd devised for having successfully penetrated the device's security model, yielding the root signing and encryption keys. These keys are the essential element of a full (and even minimally usable) breach, capable of installing and running any new software on any PlayStation 3 unit.**

On January 2, 2011, Hotz posted a copy of the root keys of the PlayStation 3 on his website. These keys were later removed from his website as a result of legal action by Sony against fail0verflow and Hotz. In response to his continued publication of PS3 exploit information, Sony filed on January 11, 2011 for an application for a temporary restraining order (TRO) against him in the US District Court of Northern California. On January 14, 2011, Hotz appeared in an interview on G4's *The Loop*, where he explained his involvement with the PlayStation 3.

Fail0verflow's has released of PSJailbreak making the PS3 station a modified firmware host (jailbreaking) able to install a lot of not standard games (source : https://www.eurogamer.net/articles/digitalfoundry-ps3-security-in-tatters).

Since the Hypervisor technology was the CPU guardian that should have preserved from the unauthorized code execution the previous hacks were made without touching the Hypervisor.

Every PS3 executable file is encrypted, or signed, using private ciphers only available to Sony itself. It has long been established that brute-forcing the keys would take hundreds of thousands of computers hundreds of thousands of years to complete.

PlayStation 3 established that, due to the console's utilisation of AES encryption the PlayStation 3 adopts an AES 128 encryption format.

On the contrary , one of the console's security vulnerabilities, consists in that the PlayStation 3 TCP & UDP communications are unencrypted.

How the PS3 hypervisor was hacked

It was George Hotz in 2010 the person who announced that he hacked the Playstation 3 and then provided exploit details. The PS3 depends on a hypervisor for security enforcement. The PS3 allows users to run ordinary Linux if they wish, but it still runs under management by the hypervisor. The hypervisor does not allow the Linux kernel to access various devices, such as the GPU. If a way was found to compromise the hypervisor, direct access to the hardware is possible, and other less privileged code could be monitored and controlled by the attacker (Source : https://rdist.root.org/2010/01/27/how-the-ps3-hypervisor-was-hacked/).

Hacking the hypervisor is not the only step required to run pirated games. Each game has an encryption key stored in an area of the disc called ROM Mark. The drive firmware reads this key and supplies it to the hypervisor to use to decrypt the game during loading. The hypervisor would need to be subverted to reveal this key for each game. Another approach would be to compromise the Blu-ray drive firmware or skip extracting the keys and just slave the decryption code in order to decrypt each game. The hypervisor code runs on both the main CPU (PPE) and one of its seven Cell coprocessors (SPE). The root hardware keys used to decrypt the bootloader and then hypervisor are present only in the hardware. This could also mean that each Cell processor has some unique keys, and decryption does not depend on a single global root key.

George's hack compromises the hypervisor after booting Linux via the “OtherOS” feature. He has used the exploit to add arbitrary read/write RAM access functions and dump the hypervisor. Access to lv1 is a necessary first step in order to mount other attacks against the drive firmware or games.

This attack is called "**glitching attack**". This kind of hardware attack involves sending a carefully-timed voltage pulse in order to cause the hardware to misbehave in some useful way. It has long been used by smart card hackers to unlock cards. Typically, hackers would time the pulse to target a loop termination condition, causing a loop to continue forever and dump contents of the secret ROM to an accessible bus. George connected an FPGA to a single line on his PS3's memory bus. He programmed the chip with very simple logic: send a 40 ns pulse via the output pin when triggered by a pushbutton. This can be done with a few lines of Verilog. While the length of the pulse is relatively short (but still about 100 memory clock cycles of the PS3), the triggering is extremely imprecise. However, he used software to setup the RAM to give a higher likelihood of success than it would first appear (Source : https://www.maxconsole.com/threads/how-the-ps3-hypervisor-was-hacked-now-fully-explained.2723/).

His goal was to compromise the hashed page table (HTAB) in order to get read/write access to the main segment, which maps all memory including the hypervisor. The exploit is a Linux kernel module that calls various system calls in the hypervisor dealing with memory management. It allocates, deallocates, and then tries to use the deallocated memory as the HTAB for a virtual segment. If the glitch successfully desynchronizes the hypervisor from the actual state of the RAM, it will allow the attacker to overwrite the active HTAB and thus control access to any memory region.

The exploit then creates a virtual segment . Once the hypervisor switches to this virtual segment, the attacker now controls all of memory and thus the hypervisor itself. The exploit installs two sys-calls that give direct read/write access to any memory address, then returns back to the kernel. The low level access given to guest OS kernels means that any bug in the hypervisor is likely to be accessible to attacker code due to the broad API it offers.

PS4 General Info.

The **PlayStation 4** (officially abbreviated as **PS4**) is an eighth-generation home video game console developed by Sony Interactive Entertainment. Announced as the successor to the PlayStation 3 in February 2013, it was launched on November 15 in North America, November 29 in Europe, South America and Australia, and on

February 22, 2014 in Japan. It's the 4th best-selling console of all time. It competes with Microsoft's Xbox One and Nintendo's Wii U and Switch.

PS4 Architecture.

Processors and memory

The PlayStation 4 uses a semi-custom Accelerated Processing Unit (APU) developed by AMD in cooperation with Sony and is manufactured by TSMC on a 28 nm process node. Its APU is a single-chip that combines a central processing unit (CPU) and graphics processing unit (GPU), as well as other components such as a memory controller and video decoder/encoder. The console also includes secondary custom chips that handle tasks associated with downloading, uploading, and social gameplay. These tasks can be handled seamlessly in the background during gameplay or while the system is in sleep mode.

Though not much is publicly known of the PS4's audio capabilities, the console also contains a dedicated hardware audio module, which can support in-game chat with minimal external resources as well as "a very large number" of MP3 streams for use in in-game audio.

Hacking the PS4 (Source : https://cturt.github.io/ps4.html)

The PS4 features a custom AMD x86-64 CPU (8 cores), as well as having a well documented CPU architecture, much of the software used in the PS4 is open source. Most notably, the PS4's Orbis OS is based on FreeBSD (9.0), just like the PS3's OS was (with parts of NetBSD as well); and includes a wide variety of additional open source software as well, such as Mono VM, and WebKit.

WebKit is the open source layout engine which renders web pages in the browsers for iOS, Wii U, 3DS, PS Vita, and the PS4. In particular, the browser in PS4 firmware 1.76 uses a version of WebKit which is vulnerable to CVE-2012-3748, a heap-based buffer overflow in the JSArray::sort(...) method.

Several attacks on the PS4 exist. These include attacks on the hardware, as well as attacks on the software. Software attacks consisting of a WebKit exploit and a kernel exploit allow for arbitrary code execution with kernel privileges. A WebKit exploit enables code execution with limited privileges. Most vulnerabilities on the PS4 originate from flawed implementations.

There exist a full chain of exploits to ultimately gain kernel code execution on the PS4 by just visiting a web page on this Internet Browser.

In 2014 nas and Proxima announced that they had successfully been able to port an exploit using this vulnerability, originally written for Mac OS X Safari, to the PS4's internet browser, and released the PoC code publicly as the first entry point into hacking the PS4.

This gives us arbitrary read and write access to everything the WebKit process can read and write to, which can be used to dump modules, and overwrite return addresses on the stack, letting us control the instruction pointer register (rip) to achieve ROP execution.
Unlike in primitive devices like the DS, the PS4 has a kernel which controls the properties of different areas of memory. Pages of memory which are marked as executable cannot be overwritten, and pages of memory which are marked as writable cannot be executed; this is known as Data Execution Prevention (DEP).

This means that we can't just copy a payload into memory and execute it. However, we can execute code that is already loaded into memory and marked as executable.

It wouldn't be very useful to jump to a single address if we can't write our own code to that address, so we use ROP.

Return-oriented Programming (ROP) is just an extension to traditional stack smashing, but instead of overwriting only a single value which rip will jump to, we can chain together many different addresses, known as gadgets.

A gadget is usually just a single desired instruction followed by a ret.

In x86_64 assembly, when a ret instruction is reached, a 64 bit value is popped off the stack and rip jumps to it; since we can control the stack, we can make every ret instruction jump to the next desired gadget.

Although you may have to be creative with how you write ROP chains, it is generally accepted that within a sufficiently large enough code dump, there will be enough gadgets for Turing-complete functionality; this makes ROP a viable method of defeating DEP.

Address Space Layout Randomization (ASLR) is a security technique which causes the base addresses of modules to be different every time you start the PS4.

It has been reported that very old firmwares (1.05) don't have ASLR enabled, but it was introduced sometime before firmware 1.70. Note that kernel ASLR is not enabled (for firmwares 1.76 and lower at least), which will be proved later.

For most exploits ASLR would be a problem because if you don't know the addresses of the gadgets in memory, you would have no idea what to write to the stack.

Luckily this is not limited to just writing static ROP chains. It can be used JavaScript to read the modules table, which will tell us the base addresses of all loaded modules. Using these bases, it can then be calculated the addresses of all gadgets before it triggers ROP execution, defeating ASLR.

Using JavaScript to write and execute dynamic ROP chains gives a tremendous advantage over a traditional, static buffer overflow attack.

As well as being necessary to defeat ASLR, JavaScript also lets it be possible to readd the user agent of the browser, and provide different ROP chains for different browser versions, giving this exploit a greater range of compatibility.

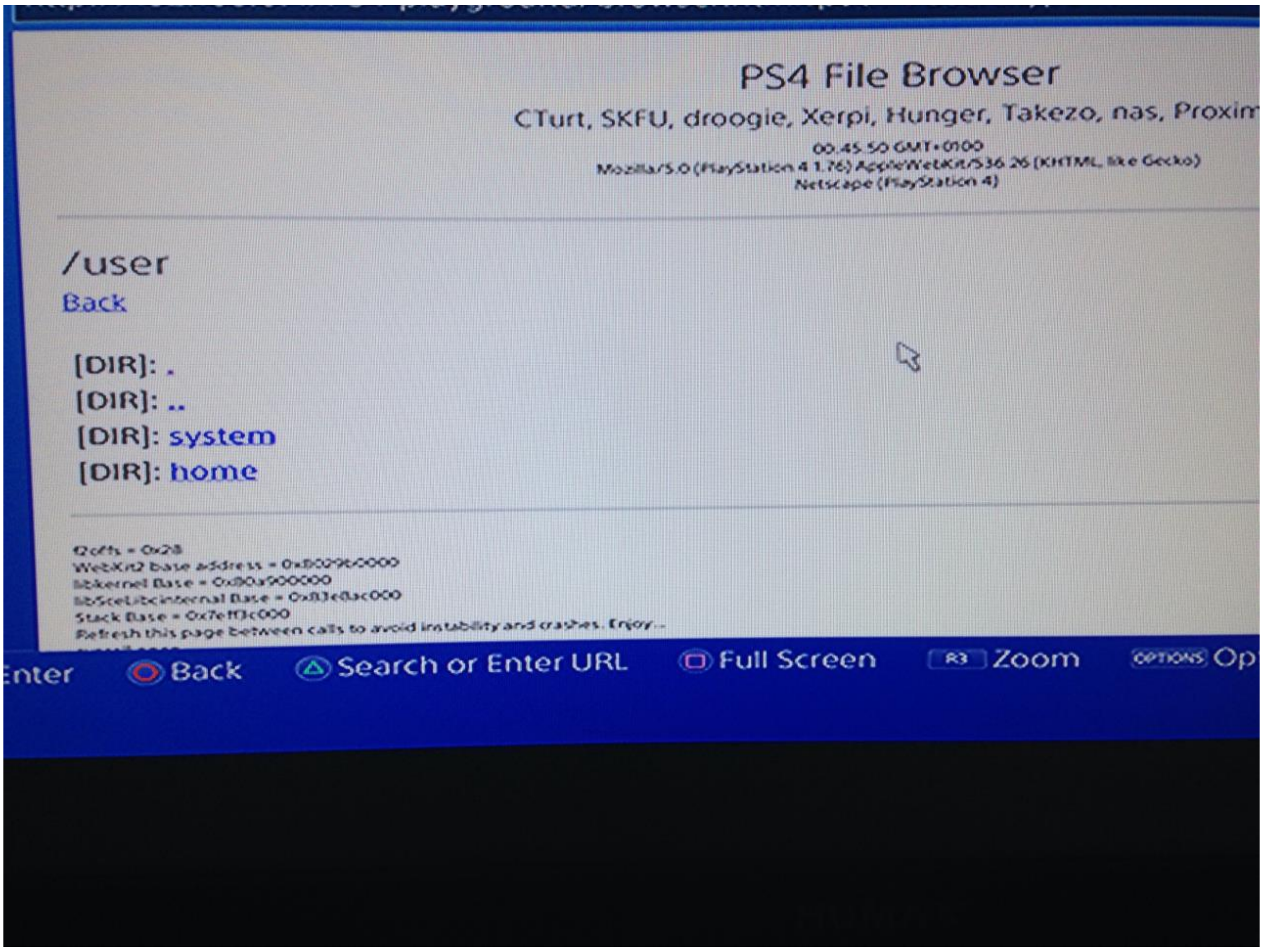
Browsing the filesystem

The PS4 uses the standard FreeBSD 9.0 system calls for reading files and directories.

However, whilst using read for some directories such as /dev/ will work, others, such as / will fail.

You can read some of these devices, for example: reading /dev/urandom will fill the memory with random data.

It is also possible to parse this memory to create a clean list of entries; look at browser.html in the repository for a complete file browser:



PS4 File Browser

CTurt, SKFU, droogie, Xerpi, Hunger, Takezo, nas, Proxim

```
00.45.50 GMT+0100
Mozilla/5.0 (PlayStation 4 1.76) AppleWebKit/536.26 (KHTML, like Gecko)
Netscape (PlayStation 4)
```

/user

Back

```
[DIR]: .
[DIR]: ..
[DIR]: system
[DIR]: home
```

```
Qoffs = 0x28
WebKit2 base address = 0x8029c0000
libkernel Base = 0x80a900000
libSceLibcinternal Base = 0x83e8ac000
Stack Base = 0x7efff3c000
Refresh this page between calls to avoid instability and crashes. Enjoy...
```

The hard drive of ps4 contained in the system appears encrypted while the main info is not stored on the disk but he information stored within the PlayStation Network (PSN).
We do have access to a lot of interesting stuff though including encrypted save data, trophies, and account information.

The best approach from here seems to be reverse engineering all of the modules which can be dumped, in order to document as many of Sony's custom system calls as possible.
Recently Jaicrab has discovered two UART ports on the PS4 which shows us that there are hardware hackers interested in the PS4. Although the role of hardware hackers has traditionally been to dump the RAM of a system, which we can already do thanks to the WebKit exploit, there's also the possibility of a hardware triggered kernel vulnerability being found, like geohot's original PS3 hypervisor hack.