

Análisis de firmwares

Hardware Hacking Day 2024

Contexto

- Hemos realizado un dump de una memoria o hemos descargado un firmware de la página de un fabricante...
- Tenemos un solo fichero con los contenidos de toda la memoria flash/todo el firmware...
- ¿Qué hacemos?

Contexto

- Hemos realizado un dump de una memoria o hemos descargado un firmware de la página de un fabricante...
- Tenemos un solo fichero con los contenidos de toda la memoria flash/todo el firmware...
- ¿Qué hacemos?
 - Binwalk? :D

Firmware 1: Ejercicio 1

- Identificación del formato del fichero proporcionado

Firmware 1: Ejercicio 1

- Identificación del formato del fichero proporcionado

```
% binwalk firmware_1
```

```
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
31064127    0x1DA003F    CFE boot loader
32101704    0x1E9D548    Sega MegaDrive/Genesis raw ROM dump,
Name: "5 17 19 08 62 1", "01 73 29 65 71 5",
35895853    0x223BA2D    StuffIt Deluxe Segment (data): f.hT.|
```

Firmware 1: Ejercicio 1

- Identificación del formato del fichero proporcionado

```
% binwalk firmware_1
```

```
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
31064127    0x1DA003F    CFE boot loader
32101704    0x1E9D548    Sega MegaDrive/Genesis raw ROM dump,
Name: "5 17 19 08 62 1", "01 73 29 65 71 5",
35895853    0x223BA2D    StuffIt Deluxe Segment (data): f.hT.|
```

Conclusiones

- Las herramientas son herramientas, debemos usarlas con cautela y adecuación...



Contexto

- Hemos realizado un dump de una memoria o hemos descargado un firmware de la página de un fabricante...
- Tenemos un solo fichero con los contenidos de toda la memoria flash/todo el firmware...
- ¿Qué hacemos?
 - Investigar cuales son los formatos más comunes de dumps...

Formatos de dump: Intel HEX

- Es formato texto. Se puede abrir con un editor de texto!
- Todas las líneas comienzan por “:”
- Típico en microcontroladores como PIC, AVR, algunos chips de ARM como los nRF (Nordic Semiconductor), algunas EEPROMs...

Formatos de dump: Intel HEX

- [https://es.wikipedia.org/wiki/HEX_\(Intel\)](https://es.wikipedia.org/wiki/HEX_(Intel))

```
:10010000214601360121470136007EFE09D2190140
:100110002146017EB7C20001FF5F16002148011988
:10012000194E79234623965778239EDA3F01B2CAA7
:100130003F0156702B5E712B722B732146013421C7
:00000001FF
```

■ Código de inicio
■ Longitud
■ Dirección
■ Tipo de registro
■ Datos
■ Checksum

Formatos de dump: Hexdump

- Es formato texto. Se puede abrir con un editor de texto!
- Pueden variar las columnas dependiendo de la herramienta...
- Si existe una columna con direcciones, normalmente serán consecutivas pero pueden no serlo!!
- Siempre encontraremos la sección principal con los datos en formato hexadecimal
- NUNCA usar la columna de texto para convertir a formato binario! Pérdida de información!
- Típico en volcados desde un bootloader a través de una UART...

Formatos de dump: Hexdump

- https://en.wikipedia.org/wiki/Hex_dump

```
00000000  30 31 32 33 34 35 36 37  38 39 41 42 43 44 45 46  |0123456789ABCDEF|
00000010  0a 2f 2a 20 2a 2a 2a 2a  2a 2a 2a 2a 2a 2a 2a 2a  |./* *****|
00000020  2a 2a 2a 2a 2a 2a 2a 2a  2a 2a 2a 2a 2a 2a 2a 2a  |*****|
00000030  2a 2a 2a 2a 2a 2a 2a 2a  2a 2a 2a 2a 2a 2a 2a 2a  |*****|
00000040  2a 2a 20 2a 2f 0a 09 54  61 62 6c 65 20 77 69 74  |** */..Table wit|
00000050  68 20 54 41 42 73 20 28  30 39 29 0a 09 31 09 09  |h TABs (09)..1..|
00000060  32 09 09 33 0a 09 33 2e  31 34 09 36 2e 32 38 09  |2..3..3.14.6.28.|
00000070  39 2e 34 32 0a                                     |9.42.|
00000075
```

Formatos de dump: Binario RAW

- No tiene porque ser un archivo de texto!
- Debemos usar un editor hexadecimal
- Si se trata como texto podemos perder información!
- No vamos a ver una estructura definida, dependerá del contenido...
- Es el formato más común, formato por defecto de muchas herramientas de volcado de memorias
- Es el formato común esperado por muchas herramientas de análisis como Binwalk!

Formatos de dump: Binario RAW

```
File Edit View Layout Extras Help ImHex - foto1.jpg
foto1.jpg
Hex editor
Address 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F ASCII
00000000: FF D8 FF E0 00 10 4A 46 49 46 00 01 01 01 00 60 .....JFIF.....
00000010: 00 60 00 00 FF E1 2D EA 45 78 69 66 00 00 4D 4D .....=,Exif..MM
00000020: 00 2A 00 00 00 08 00 06 00 0B 00 02 00 00 00 26 ..*.....&
00000030: 00 00 08 62 01 12 00 03 00 00 00 01 00 01 00 00 ...b.....
00000040: 01 31 00 02 00 00 00 26 00 00 08 88 01 32 00 02 ..1.....&....2..
00000050: 00 00 00 14 00 00 08 AE 87 69 00 04 00 00 00 01 .....i.....
00000060: 00 00 08 C2 EA 1C 00 07 00 00 08 0C 00 00 00 56 .....V
00000070: 00 00 11 46 1C EA 00 00 00 08 00 00 00 00 00 00 ...F.....
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
Page: 0x01 / 0x01 Region: 0x00000000 - 0x00019C6D (0 - 105581)
Selection: None Data Size: 0x00019C6E (0x19C6E | 103.11 kiB)
Aa abc ¶ Data visualizer: Little Hexadecimal 16
```

Firmware 1: Ejercicio 2

- Identificación del formato del fichero proporcionado. Otra vez...

Firmware 1: Ejercicio 2

- Identificación del formato del fichero proporcionado. Otra vez...

```
% cat firmware_1 | less
```

```
00000000  8b 01 00 00 cc 78 0d 00 01 01 00 70 00 70 01 00 |.....x.....p.p..|
00000010  00 00 00 00 00 00 00 00 00 04 00 00 00 00 01 a1 |.....|
00000020  02 01 d0 67 02 00 00 00 5b 00 00 00 68 00 00 00 |...g....[...h...|
00000030  ff 5f 2d e9 c1 02 00 fa 00 00 a0 e3 ff 9f bd e8 |._-.....|
00000040  fe 1f 2d e9 36 0f 07 ee fe 1f bd e8 1e ff 2f e1 |..-.6...../..|
```

```
...
```

Firmware 1: Ejercicio 2

- Identificación del formato del fichero proporcionado. Otra vez...
 - Es un fichero de texto!
 - Las líneas no comienzan por “:”
 - Hay una columna que parecen direcciones y un cuerpo principal en hexadecimal

Firmware 1: Ejercicio 2

- Identificación del formato del fichero proporcionado. Otra vez...
 - Es un fichero de texto!
 - Las líneas no comienzan por “:”
 - Hay una columna que parecen direcciones y un cuerpo principal en hexadecimal
 - Es un Hexdump! Binwalk esperaba un binario raw!

Conclusiones

- Casi todas las herramientas de análisis que vamos a usar trabajan con el formato binario
- Si nuestro dump está en formato hexadecimal o Intel HEX habrá que transformarlo

Conclusiones

- Las herramientas son herramientas y bien usadas nos hacen felices...



Intel HEX > Binario RAW

- Existen infinidad de herramientas:

- SRrecord: <https://github.com/sierrafoxtrot/srecord>  

```
srec_cat inputFile.hex -Intel -output outputFile.bin -binary
```

- Binex: <http://www.nlsw.nl/software/> 

```
binex.exe /B inputFile.hex
```

- HEX2BIN: <https://www.keil.com/download/docs/7.asp> 

```
hex2bin inputFile.hex outputFile.bin
```

Hexdump > Binario RAW

- Lo más común es xxd, en muchos sitios se recomienda este comando pero ESTÁ MAL:

```
xxd -r -p inputHexdump.txt outputBinary.bin
```

- xxd puede no llevarse bien con los números de las direcciones, es importante quitar la columna de las direcciones:

```
cut -d' ' -f3-19 inputHexdump.txt | xxd -r -p >  
outputBinary.bin
```

- Ojo porque puede haber saltos de dirección en el dump!

Firmware 1: Ejercicio 3

- Convertir el fichero a binario e identificarlo

```
% cut -d' ' -f3-19 firmware_1 | xxd -r -p > firmware_1.bin
```

```
% md5sum firmware_1.bin
```

```
c2ca9f8a3c011c87007a34c567311aea  firmware_1.bin
```

Firmware 1: Ejercicio 3

- Convertir el fichero a binario e identificarlo

```
% binwalk firmware_1.bin | less
```

```
DECIMAL          HEXADECIMAL      DESCRIPTION
```

```
-----  
139553          0x22121          Certificate in DER format (x509 v3), header length: 4,  
sequence length: 832  
...  
731273          0xB2889          LZ0 compressed data  
874485          0xD57F5          ESP Image (ESP32): segment count: 11, flash mode: QUIO,  
flash size: 1MB, entry address: 0xb8000000  
...  
2120508         0x205B3C         xz compressed data  
6292304         0x600350         LZ4 compressed data  
6293502         0x6007FE         LZ4 compressed data  
6294527         0x600BFF         Zip multi-volume archive data, at least PKZIP v2.50 to  
extract  
...
```

Conclusiones

- Aun usando las herramientas con cuidado...



Conclusiones

- Tenemos solo una estrategia de búsqueda en binarios
- Es deseable tener al menos un plan B...

Herramientas para binarios

- Búsqueda de firmas:
 - Binwalk - <https://github.com/OSPFG/binwalk>
 - Unblob - <https://github.com/onekey-sec/unblob>
- Estadísticos: Entropía - Con Binwalk/ImHex
- Editor hex: ImHex - <https://github.com/WerWolv/ImHex>

Herramientas para binarios: búsqueda de firmas

- ¿Que es una firma/magic number?
 - Algunos formatos de archivo usan un “magic number” para identificar su inicio
 - Es una constante que no **debería** variar
 - Cuidado: Algunos fabricantes cambian las firmas de sus archivos porque los customizan o para no ser detectados

Herramientas para binarios: búsqueda de firmas

- file (Utilidad de linux)
- Binwalk

`binwalk fichero.bin` (Solo busca firmas)

`binwalk -e fichero.bin` (Extrae)

- Unblob

`unblob fichero.bin` (Extrae)

Herramientas para binarios: entropía

- ¿Qué es la entropía?

Herramientas para binarios: entropía

- Es una medida de “densidad” de información
- La entropía será máxima cuando haya una buena compresión
- La entropía será alta con el cifrado
- La entropía será media cuando haya formatos estructurados
- La entropía es mínima en regiones vacías

Herramientas para binarios: entropía

- ¿Qué entropía tiene el siguiente fichero?

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000010:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000020:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000050:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000060:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000090:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000A0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000B0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000D0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000E0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000F0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Herramientas para binarios: entropía

- ¿Este tendrá más o menos?

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000010:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000020:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000050:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000060:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080:	FF															
00000090:	FF															
000000A0:	FF															
000000B0:	FF															
000000C0:	FF															
000000D0:	FF															
000000E0:	FF															
000000F0:	FF															

Herramientas para binarios: entropía

- ¿Este tendrá más o menos?
- ¿Es aleatorio o predecible?

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000:	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000010:	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
00000020:	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	!"#\$%&'()*+,-./
00000030:	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F	0123456789;<=>?
00000040:	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	@ABCDEFGHIJKLMNO
00000050:	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	PQRSTUVWXYZ[\]^_
00000060:	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	`abcdefghijklmno
00000070:	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F	pqrstuvwxyz{ }~.
00000080:	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
00000090:	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
000000A0:	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
000000B0:	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
000000C0:	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
000000D0:	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
000000E0:	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
000000F0:	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

Herramientas para binarios: entropía

- ImHex:

View > Data information > Analyze

- Binwalk:

```
binwalk -E fichero.bin
```

- Cyberchef: <https://gchq.github.io/CyberChef/>

Open file > Add recipe > Entropy

Herramientas para binarios: editor hex

- Es la herramienta más poderosa
- Es posiblemente la que más esfuerzo y tiempo requiere
- Útil para afinar los pequeños detalles de los datos que extraemos de las otras herramientas...

Herramientas para binarios: editor hex

- ImHex - <https://github.com/WerWolv/ImHex> 
- Hobbits - <https://github.com/Mahlet-Inc/hobbits>
- 101 Hex Editor - <https://www.sweetscape.com/010editor/>
- HxD - <https://mh-nexus.de/en/hxd/>
- Okteta - <https://apps.kde.org/es/okteta/>
- Vix - <https://github.com/batchDrake/vix>

Firmware 1: Ejercicio 4

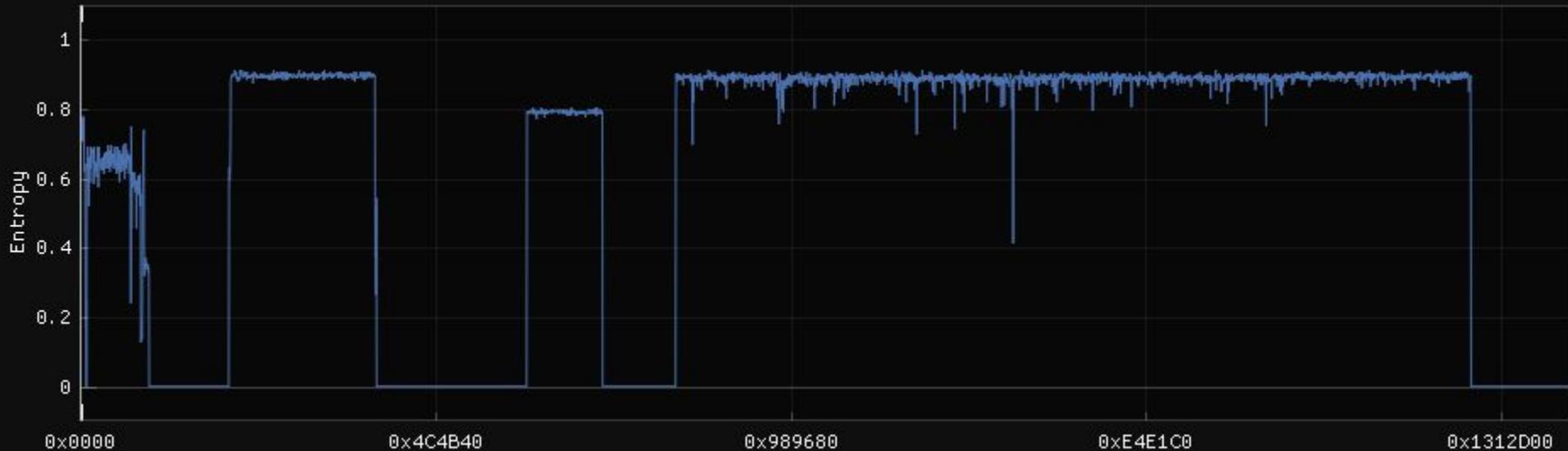
- Identificar el firmware. Una vez más...
 - ¿Qué estrategia podemos usar?

Firmware 1: Ejercicio 4

- Identificar el firmware. Una vez más...
 - Análisis de entropía del firmware

Firmware 1: Ejercicio 4

- Análisis de entropía del firmware



Firmware 1: Ejercicio 4

- Análisis de entropía del firmware



Firmware 1: Ejercicio 4

- Análisis de entropía del firmware



¿Cómo está organizada una flash?

- Depende de la complejidad del dispositivo
 - Micros pequeños solo tendrán una sección
 - Dispositivos más complejos tendrán varias secciones
- ¿Cómo organizada nuestra flash?

¿Cómo está organizada una flash?

- Depende de la complejidad del dispositivo
 - Micros pequeños solo tendrán una sección
 - Dispositivos más complejos tendrán varias secciones
- ¿Cómo organizada nuestra flash?
 - Particiones o secciones
 - Separadas por espacios en blanco

¿Cómo partimos un binario?

- Hay editores hex que lo permiten
- Utilidades clásicas como dd

```
dd if=input.bin of=output.bin bs=1 skip=$offset count=$size
```

- Ojo, dd no entiende los números en hexadecimal!

```
dd if=input.bin of=output.bin bs=1 skip=$((0x100))  
count=$((0xa00))
```

Firmware 1: Ejercicio 5

- Vamos a partir nuestro firmware en las distintas secciones!
 - Queremos que el fichero de salida tenga el inicio bien alineado.
 - El final del fichero de salida no tiene porque estar perfectamente alineado, podemos incluir espacio en blanco...

```
dd if=input.bin of=output.bin bs=1 skip=$((0x100))  
count=$((0xa00))
```

Firmware 1: Ejercicio 5

- Vamos a partir nuestro firmware en las distintas secciones!

```
% dd if=firmware_1.bin of=part_1.bin bs=1  
skip=$((0x000000)) count=$((0x200000))
```

```
% dd if=firmware_1.bin of=part_2.bin bs=1  
skip=$((0x200000)) count=$((0x400000))
```

```
% dd if=firmware_1.bin of=part_3.bin bs=1  
skip=$((0x600000)) count=$((0x200000))
```

```
% dd if=firmware_1.bin of=part_4.bin bs=1  
skip=$((0x800000)) count=$((0xc00000))
```

Firmware 1: Ejercicio 5

- Vamos a partir nuestro firmware en las distintas secciones!

```
% md5sum *
```

```
01f13d25e24ed64f60999c5aa9af5d39  firmware_1
```

```
0e73b9cf5098e5c8b6266a85ce3b4c17  firmware_1.bin
```

```
54d63b8457097e38d2e4e7bb0d8ff218  part_1.bin
```

```
82cb9c087aa2c112ceaa67c2f2765702  part_2.bin
```

```
26221f07d78a60ab78953811ce308bea  part_3.bin
```

```
b219ad14b1ac91695f91f5baefde90eb  part_4.bin
```

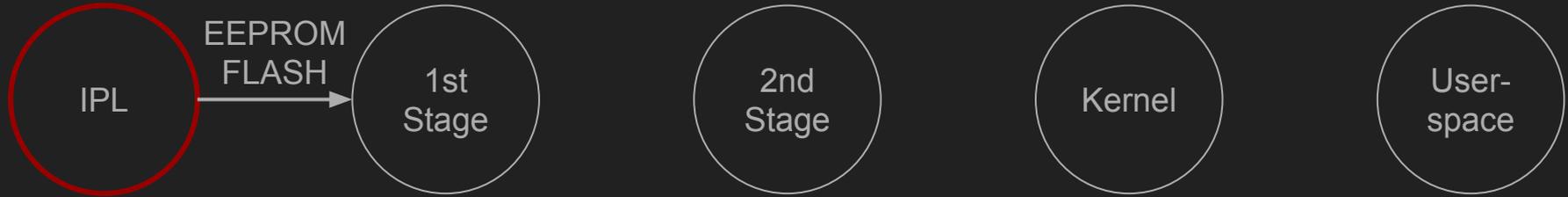
¿Cómo está organizada una flash?

- Con la entropía y búsqueda de regiones en blanco identificamos particiones/secciones de la flash
- Con la entropía y búsqueda de firmas identificamos que particiones/secciones están comprimidas, cifradas u otros formatos
- Podemos entender parte del formato, pero ¿Por qué esto está organizado de esta manera?

Entendiendo el diseño de un dispositivo

- Se busca un coste reducido y ajustado (esto da más beneficio al fabricante)
- En parte se logra a través de la “modularidad”: elijo una CPU y la combino con una RAM y una flash
- Esto genera complejidad en el software porque este tiene que adaptarse a esa modularidad!

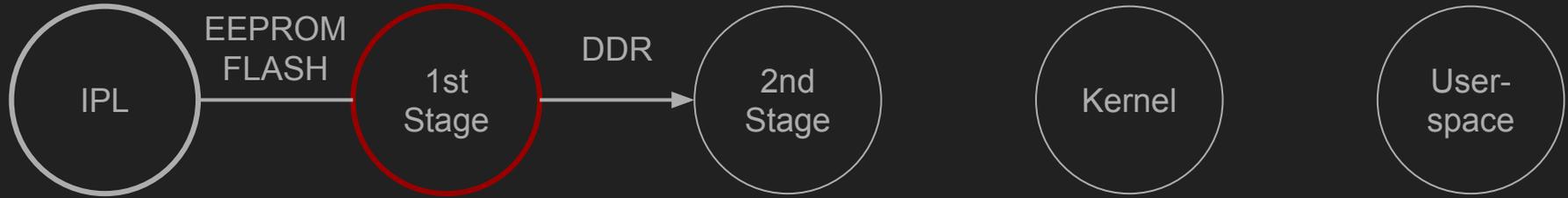
El proceso de arranque



Initial Program Loader o
Rom Boot Loader

- Muy limitado en tamaño, solo puede usar una SRAM mínima.
- Suele ser parte del SoC y usa recursos dentro del SoC.
- Inicializa mínimamente un medio de almacenamiento y copia el SPL a la SRAM.

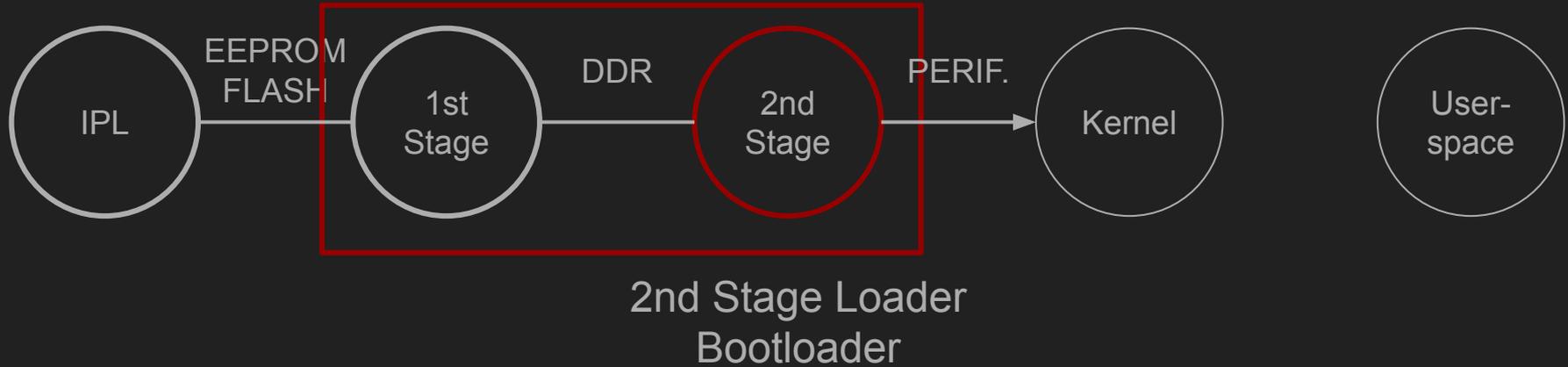
El proceso de arranque



Secondary Program Loader o
Memory Loader (MLO) o
1st Stage Loader

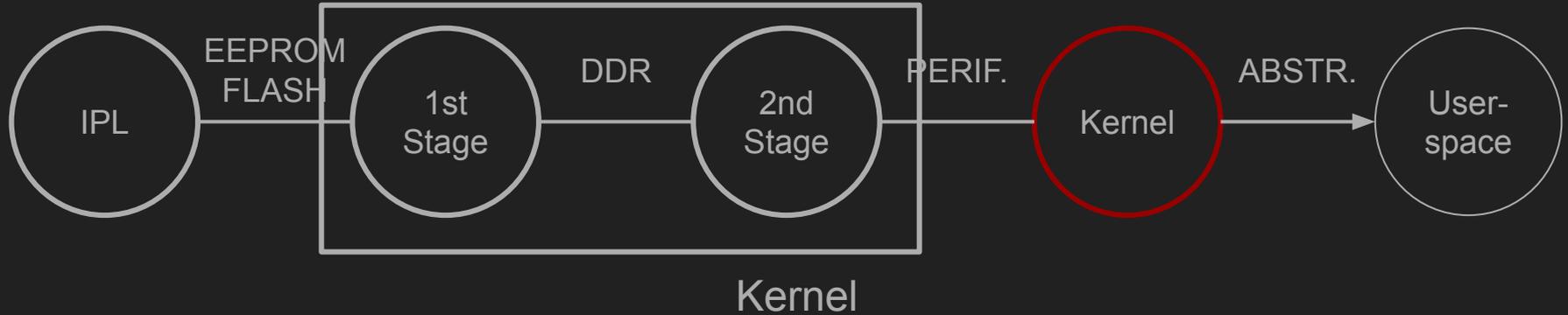
- Desde la SRAM inicializa la DDR de más tamaño.
- Casi siempre reconfigura el almacenamiento de arranque.
- Posiblemente configura algunos periféricos (UART para debug).
- Carga el 2nd stage de mucho mayor tamaño en la DDR.

El proceso de arranque



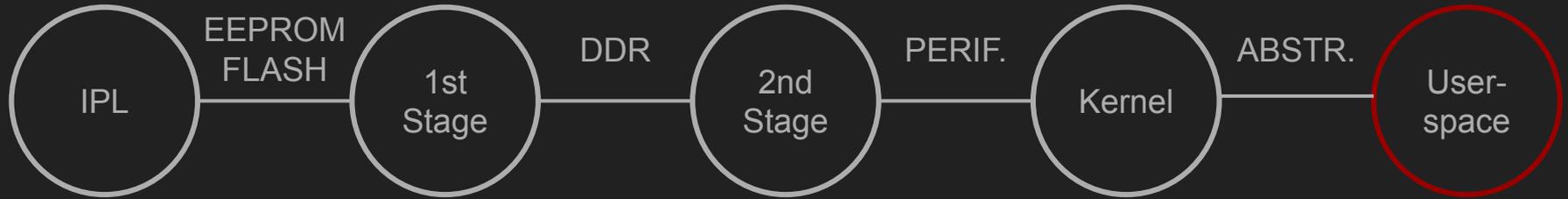
- Habitualmente llamamos bootloader al 1st stage + 2nd stage juntos.
- Los más típicos son U-Boot/CFE/Grub...
- Inicialización más completa del hardware. Puede tener mucha funcionalidad: Línea de comandos, boot de red, soporte para distintos almacenamientos...

El proceso de arranque



- Gestión de memoria, gestión de procesos, permisos, abstracción del hardware...
- Lo más común es una versión modificada de un Kernel obsoleto.
- Por motivos de licenciamiento el código propietario suele proveerse en módulos sin código fuente.

El proceso de arranque



Userspace

- Herramientas, servicios, configuraciones...

¿Cómo está organizada una flash?

- Particiones/secciones más comunes
 - Bootloader
 - Kernel
 - Sistema de archivos principal (userspace)
 - Otras particiones “custom” para traducciones, configuración u otros

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!
 - Ahora podemos usar la detección de firmas en cada una de las partes con el conocimiento que acabamos de adquirir.

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!
 - part_1: ??
 - part_2: ??
 - part_3: wtf???
 - part_4: squashfs - Sistema de archivos comprimido, será el userspace!

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!
 - part_1: ??
 - part_2: zImage - Es un kernel comprimido! ARM!
 - part_3: wtf???
 - part_4: squashfs - userspace

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!
 - part_1: bootloader! Tiene instrucciones de ARM!
 - part_2: zImage - kernel
 - part_3: wtf???
 - part_4: squashfs - userspace

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!
 - part_1: bootloader
 - part_2: zImage - kernel
 - part_3: custom? Partición para despistar!
 - part_4: squashfs - userspace

Firmware 1: Ejercicio 6

- Identificar cada una de las particiones!

```
% binwalk -Y part_1.bin
DECIMAL          HEXADECIMAL      DESCRIPTION
-----
0                0x0              ARM executable code, 32-bit, little endian, at least 720
valid instructions
```

```
% binwalk part_2.bin
DECIMAL          HEXADECIMAL      DESCRIPTION
-----
0                0x0              Linux kernel ARM boot executable zImage (little-endian)
...
2058728          0x1F69E8         Flattened device tree, size: 18269 bytes, version: 17
```

```
% binwalk part_4.bin
DECIMAL          HEXADECIMAL      DESCRIPTION
-----
0                0x0              Squashfs filesystem, little endian, version 4.0,
compression: gzip, size: 11189691 bytes, 1711 inodes, blocksize: 131072 bytes, created:
2024-03-02 17:13:55
```

¿Qué podemos hacer ahora?

- El análisis de un firmware completo puede requerir mucho tiempo...
- La estrategia es priorizar para poder obtener resultados con menor esfuerzo...
- Podemos empezar por lo más fácil!

Firmware 1: Ejercicio 7

- Extraer el sistema de archivos a una carpeta
 - Binwalk
 - Squashfs (unsquashfs)

Firmware 1: Ejercicio 7

- Extraer el sistema de archivos a una carpeta

```
% unsquashfs part_4.bin
```

```
Parallel unsquashfs: Using 2 processors
```

```
1628 inodes (1343 blocks) to write
```

```
create_inode: failed to create symlink
```

```
squashfs-root/bin/ash, because Operation not permitted
```

```
create_inode: failed to create symlink
```

```
squashfs-root/bin/cat, because Operation not permitted
```

```
...
```

¿Qué podemos hacer ahora?

- Firmware custom! Modificar el userspace, modificar o incluir algún fichero nuestro, reempaquetar y volver a flashear en nuestro router.
- Analizar el firmware que nos ha dado el fabricante en busca de cosas interesantes...

Análisis del userspace

- ¿Cuál es el primer proceso que se ejecuta?
 - /sbin/init - Normalmente apunta a busybox
- ¿Que hace?
 - /etc/inittab
 - /etc/init.d/
 - ...
- ¿Hay binarios interesantes?
- ¿Hay otras configuraciones interesantes?

Ejercicio 8

- Analizar el firmware!
 - ¿Qué nos falta por conocer de este firmware?
 - ¿Existe algún archivo interesante que debemos analizar?
 - ¿Qué necesitamos para analizar esos ficheros?

Ejercicio 8

- Analizar el firmware!
 - Es interesante entender los scripts de arranque y servicios, pueden darnos funcionalidad interesante o vulnerabilidades!
 - El binario “important_secret_service” parece interesante
 - Para analizarlo necesitamos un poco de información de reversing de binarios!

