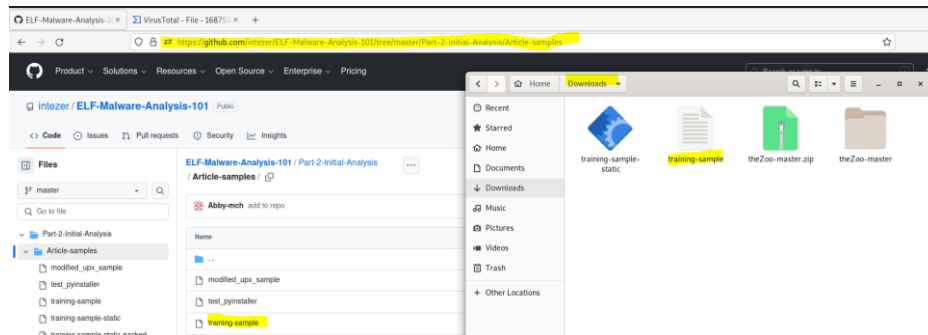
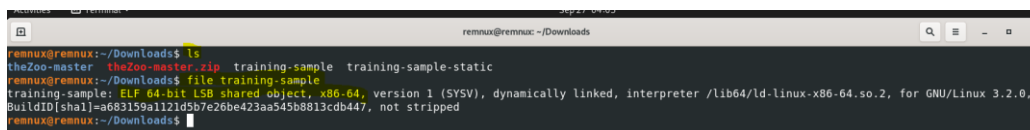
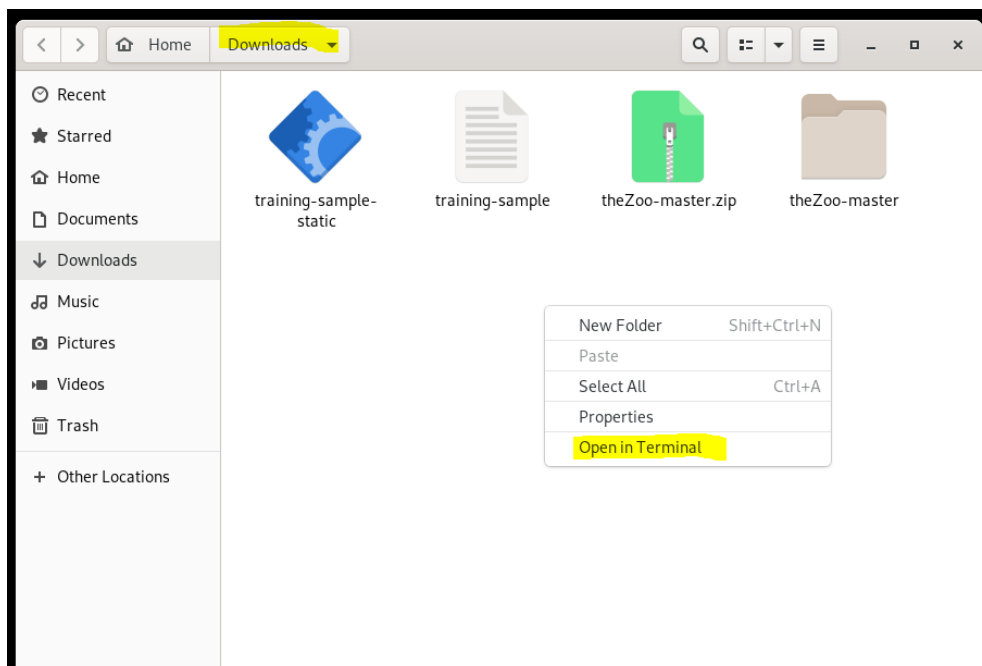


Static analysis using REMnux

- **Machine:** Launch the REMnux machine
- **Tools:** file, die(Detect it Easy), md5sum, readelf
- **Ransomware** – sample from (not a malware)
<https://github.com/intezer/ELF-Malware-Analysis-101/tree/master/Part-2-Initial-Analysis>
- Launch the browser and use the above link to download the sample files – training-sample

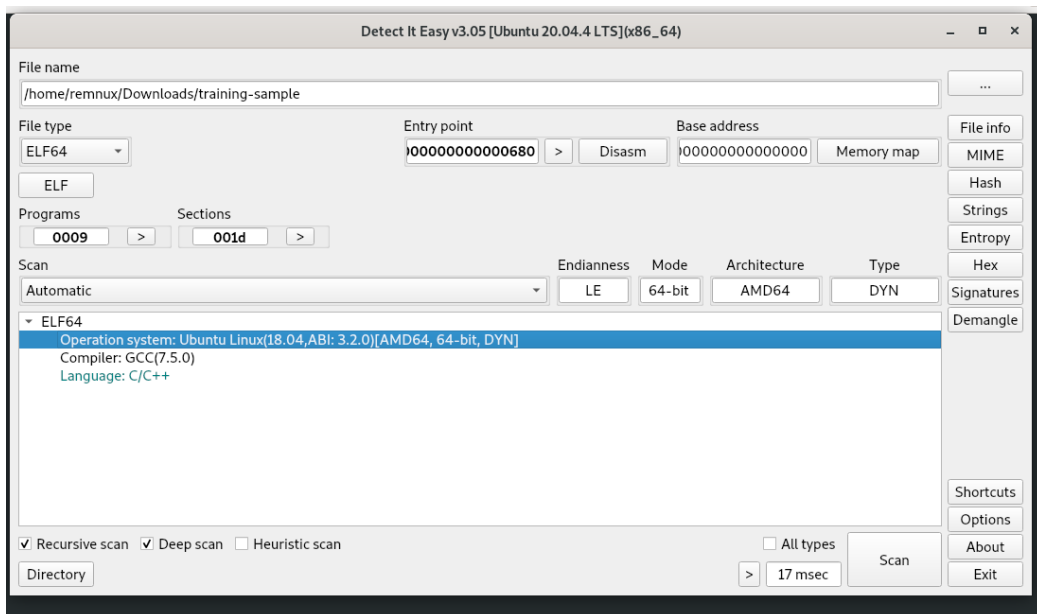
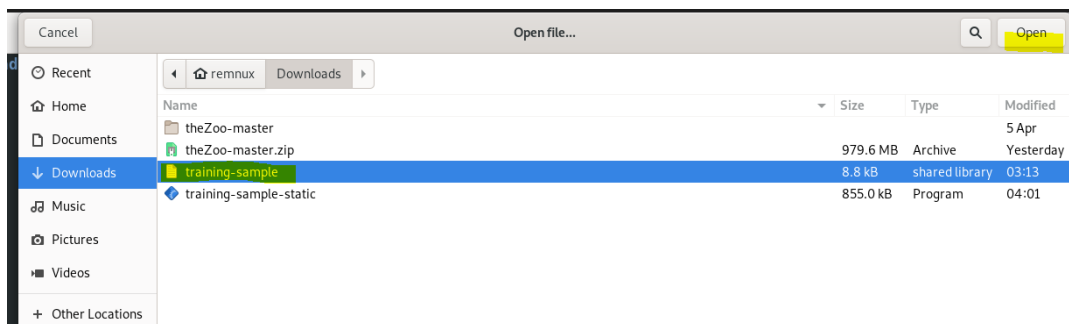
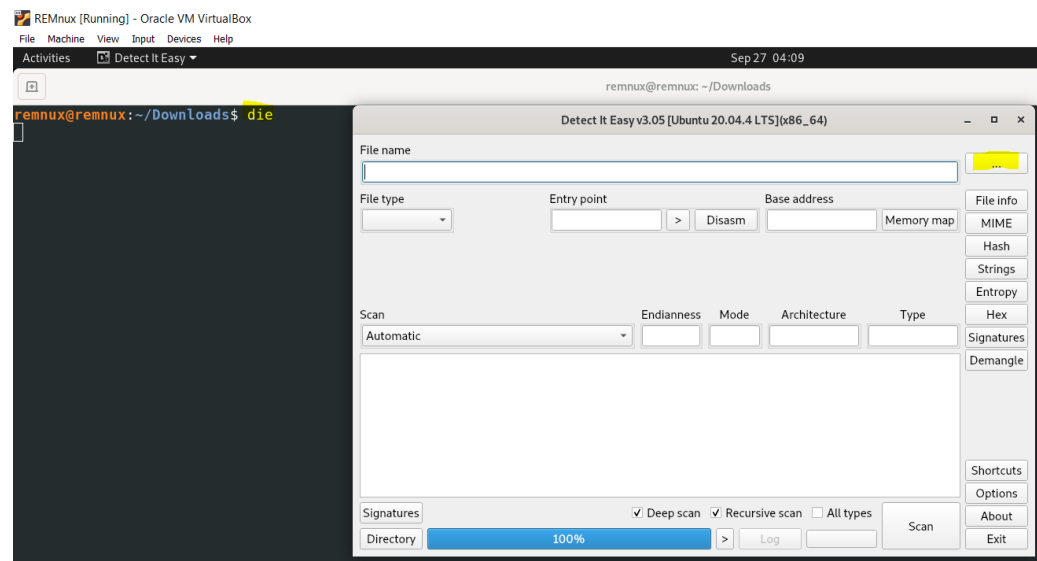


- Open the terminal and use the tool - file to identify the type of the file



- **Observations:** ELF – a binary file that can run on Linux

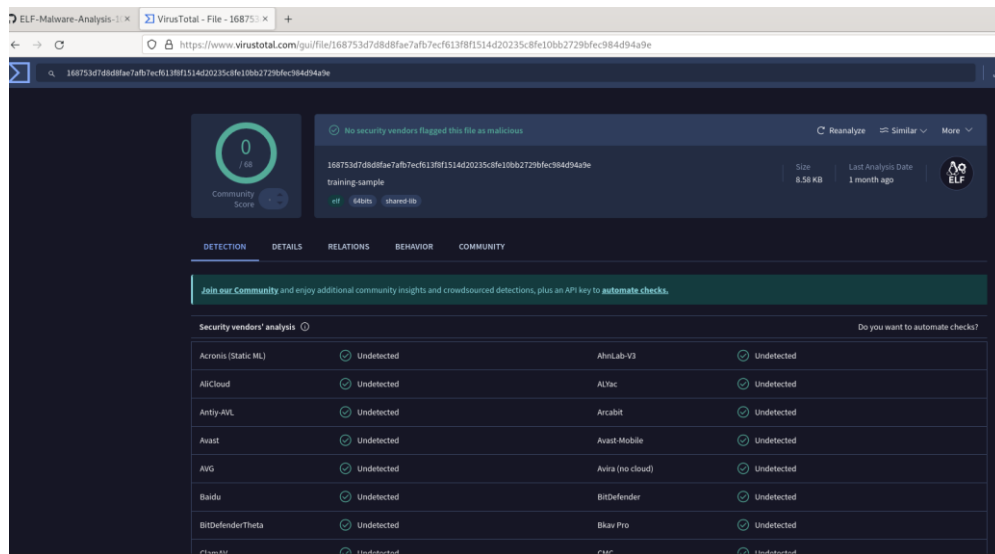
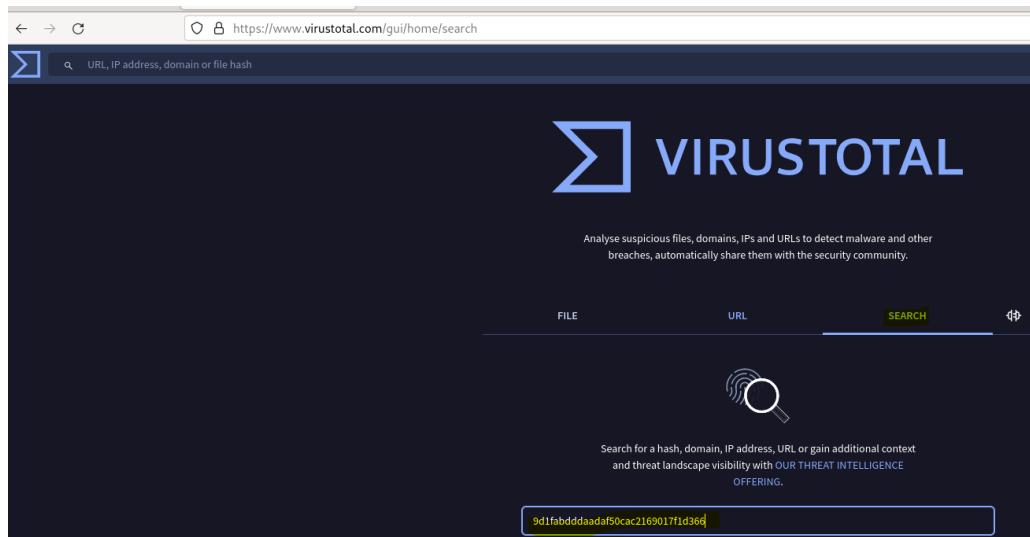
- Use the tool die(Detect it Easy) to know more about this file



- **Observations:** type of OS, name of the compiler and name of the programming language used can be detected

- Check if the malware file has already been analyzed. Find the hash of the malware sample and check in the virustotal web site

```
remnux@remnux:~/Downloads$ md5sum training-sample
9d1fabdddaadaf50cac2169017f1d366 training-sample
remnux@remnux:~/Downloads$
```



https://www.virustotal.com/gui/file/168753d7d8d8fae7afb7ec613f8f1514d20235c8fe10bb2729bfe984d949e/details

168753d7d8d8fae7afb7ec613f8f1514d20235c8fe10bb2729bfe984d949e

0 / 68
Community Score

No security vendors flagged this file as malicious

Reanalyze Similar More

168753d7d8d8fae7afb7ec613f8f1514d20235c8fe10bb2729bfe984d949e
training-sample
Size: 8.58 KB
Last Analysis Date: 1 month ago

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to [automate checks](#).

Basic properties

MD5	9d1fabdddaad50acc2169017f1d366
SHA-1	667f8fce9d587ade0d47d78e936c316a122cbbf
SHA-256	168753d7d8d8fae7afb7ec613f8f1514d20235c8fe10bb2729bfe984d949e
Virus	44a74e42221c45dbb76a967d770d0
SSDEEP	96.867GVH-86d5H4c4xgoOXY9C4Mhuj2Y7Zw70suu8Z8BQ5vP5d3.80xwee6McGpe3J/oz/n1KSids
TLSH	T13C027549F7918D3FD8657388DF743303270E890F0143232748E6796E028D8686EA09
File type	ELF [executable] [linux] [elf]
Magic	ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=a68315ba1121d5b7c26...
TeiHash	11c570090e300e00202030883802a063c0e0e000000200c080000e23049
Trid	ELF Executable and Linkable format (Linux) (50.1%) ELF Executable and Linkable format (generic) (49.8%)
DetectItEasy	ELF64 Operation system: Unix [DYN AMD64-64] Library: GLIBC (2.4) [DYN AMD64-64] Compiler: gcc ([Ubuntu 7.5.0-3ubuntu1-18.04] 7.5.0) [DYN AMD64-64]
Magika	ELF
File size	8.58 KB (8704 bytes)

- **Observation:** not a malware (since the file is written for training purpose). Click the details tab and look for useful information

- To analyse the ELF binary, use the tool – readelf

```
remnux@remnux: ~/Documents
remnux@remnux:~/Documents$ readelf
Usage: readelf <option(s)> elf-file(s)
Display information about the contents of ELF format files
Options are:
-a --all                Equivalent to: -h -l -S -s -r -d -V -A -I
-h --file-header        Display the ELF file header
-l --program-headers    Display the program headers
--segments              An alias for --program-headers
-S --section-headers    Display the sections' header
--sections              An alias for --section-headers
-g --section-groups      Display the section groups
-t --section-details    Display the section details
-e --headers            Equivalent to: -h -l -S
-s --syms               Display the symbol table
--symbols               An alias for --syms
--dyn-syms              Display the dynamic symbol table
-n --notes              Display the core notes (if present)
-r --relocs             Display the relocations (if present)
-u --unwind             Display the unwind info (if present)
-d --dynamic            Display the dynamic section (if present)
-V --version-info       Display the version sections (if present)
-A --arch-specific      Display architecture specific information (if any)
-c --archive-index      Display the symbol/file index in an archive
-D --use-dynamic        Use the dynamic section info when displaying symbols
-x --hex-dump=<number|name>
                        Dump the contents of section <number|name> as bytes
-p --string-dump=<number|name>
                        Dump the contents of section <number|name> as strings
-R --relocated-dump=<number|name>
                        Dump the contents of section <number|name> as relocated bytes
-z --decompress         Decompress section before dumping it
-w[llianrmfEsOBTtUuTqAckKl] or
```

```
remnux@remnux: ~/Downloads
remnux@remnux:~/Downloads$ readelf -s training-sample | more
Symbol table '.dynsym' contains 11 entries:
Num:  Value          Size Type  Bind  Vis      Ndx Name
0: 0000000000000000  0 NOTYPE LOCAL DEFAULT UND
1: 0000000000000000  0 NOTYPE WEAK  DEFAULT UND _ITM_deregisterTMCloneTab
2: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND puts@GLIBC_2.2.5 (2)
3: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND __stack_chk_fail@GLIBC_2.4 (3)
4: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND system@GLIBC_2.2.5 (2)
5: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND _libc_start_main@GLIBC_2.2.5 (2)
6: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND fgets@GLIBC_2.2.5 (2)
7: 0000000000000000  0 NOTYPE WEAK  DEFAULT UND gmon_start_
8: 0000000000000000  0 FUNC  GLOBAL DEFAULT UND popen@GLIBC_2.2.5 (2)
9: 0000000000000000  0 NOTYPE WEAK  DEFAULT UND _ITM_registerTMCloneTable
10: 0000000000000000  0 FUNC  WEAK  DEFAULT UND __cxa_finalize@GLIBC_2.2.5 (2)

Symbol table '.symtab' contains 70 entries:
Num:  Value          Size Type  Bind  Vis      Ndx Name
0: 0000000000000000  0 NOTYPE LOCAL DEFAULT UND
1: 0000000000000238  0 SECTION LOCAL DEFAULT 1
2: 0000000000000254  0 SECTION LOCAL DEFAULT 2
3: 0000000000000274  0 SECTION LOCAL DEFAULT 3
4: 0000000000000298  0 SECTION LOCAL DEFAULT 4
5: 00000000000002b8  0 SECTION LOCAL DEFAULT 5
6: 00000000000003c0  0 SECTION LOCAL DEFAULT 6
7: 0000000000000470  0 SECTION LOCAL DEFAULT 7
8: 0000000000000488  0 SECTION LOCAL DEFAULT 8
9: 00000000000004b8  0 SECTION LOCAL DEFAULT 9
10: 0000000000000578  0 SECTION LOCAL DEFAULT 10
11: 00000000000005f0  0 SECTION LOCAL DEFAULT 11
12: 0000000000000610  0 SECTION LOCAL DEFAULT 12
13: 0000000000000670  0 SECTION LOCAL DEFAULT 13
```

- Symbols helps to uncover the name of the functions and source codes used in a malware.
- .dynsym – contains symbols that are dynamically linked
- .symtab – contains symbols that are actual functions. Looking into the name we may be able to identify the function's task at this point

```

remnux@remnux: ~/Downloads
39: 000000000200da8 0 OBJECT LOCAL DEFAULT 21 _DYNAMIC
40: 000000000200d98 0 NOTYPE LOCAL DEFAULT 19 __init_array_start
41: 000000000000908 0 NOTYPE LOCAL DEFAULT 17 GNU_EH_FRAME_HDR
42: 000000000200f98 0 OBJECT LOCAL DEFAULT 22 GLOBAL_OFFSET_TABLE
43: 0000000000008d0 2 FUNC GLOBAL DEFAULT 14 __libc_csu_fini
44: 0000000000000000 0 NOTYPE WEAK DEFAULT UND ITM_deregisterTMCloneTab
45: 000000000201000 0 NOTYPE WEAK DEFAULT 23 data_start
46: 0000000000000000 0 FUNC GLOBAL DEFAULT UND puts@GLIBC_2.2.5
47: 0000000002010c4 0 NOTYPE GLOBAL DEFAULT 23 _edata
48: 0000000000008d4 0 FUNC GLOBAL DEFAULT 15 _fini
49: 0000000000000000 0 FUNC GLOBAL DEFAULT UND __stack_chk_fail@GLIBC_2
50: 000000000201020 50 OBJECT GLOBAL DEFAULT 23 google_dns_ping
51: 0000000000000000 0 FUNC GLOBAL DEFAULT UND system@GLIBC_2.2.5
52: 0000000000000000 0 FUNC GLOBAL DEFAULT UND __libc_start_main@GLIBC_
53: 0000000000000000 0 FUNC GLOBAL DEFAULT UND fgets@GLIBC_2.2.5
54: 000000000201000 0 NOTYPE GLOBAL DEFAULT 23 __data_start
55: 0000000000000000 0 NOTYPE WEAK DEFAULT UND __gmon_start__
56: 000000000201008 0 OBJECT GLOBAL HIDDEN 23 __dso_handle
57: 0000000000008e0 4 OBJECT GLOBAL DEFAULT 16 __IO_stdin_used
58: 000000000000860 101 FUNC GLOBAL DEFAULT 14 __libc_csu_init
59: 0000000002010c8 0 NOTYPE GLOBAL DEFAULT 24 __end
60: 000000000000680 43 FUNC GLOBAL DEFAULT 14 __start
61: 0000000002010c4 0 NOTYPE GLOBAL DEFAULT 24 __bss_start
62: 000000000201060 100 OBJECT GLOBAL DEFAULT 23 some_string
63: 00000000000080d 77 FUNC GLOBAL DEFAULT 14 main
64: 0000000000000000 0 FUNC GLOBAL DEFAULT UND popen@GLIBC_2.2.5
65: 0000000002010c8 0 OBJECT GLOBAL HIDDEN 23 __TMC_END__
66: 0000000000000000 0 NOTYPE WEAK DEFAULT UND ITM_registerTMCloneTable
67: 00000000000078a 131 FUNC GLOBAL DEFAULT 14 ping_google_dns
68: 0000000000000000 0 FUNC WEAK DEFAULT UND __cxa_finalize@GLIBC_2.2
69: 0000000000005f0 0 FUNC GLOBAL DEFAULT 11 __init
remnux@remnux: ~/Downloads$

```

- File – name of the source code file

```

remnux@remnux: ~/Downloads
27: 0000000000006b0 0 FUNC LOCAL DEFAULT 14 deregister_tm_clones
28: 0000000000006f0 0 FUNC LOCAL DEFAULT 14 register_tm_clones
29: 000000000000740 0 FUNC LOCAL DEFAULT 14 __do_global_ctors_aux
30: 0000000002010c4 1 OBJECT LOCAL DEFAULT 24 completed.7698
31: 000000000200da0 0 OBJECT LOCAL DEFAULT 20 __do_global_ctors_aux_fini
32: 000000000000780 0 FUNC LOCAL DEFAULT 14 frame_dummy
33: 000000000200d98 0 OBJECT LOCAL DEFAULT 19 frame_dummy_init_array
34: 0000000000000000 0 FILE LOCAL DEFAULT ABS training_sample.c
35: 0000000000000000 0 FILE LOCAL DEFAULT ABS crtstuff.c
36: 000000000000a74 0 OBJECT LOCAL DEFAULT 18 __FRAME_END__
37: 0000000000000000 0 FILE LOCAL DEFAULT ABS
38: 000000000200da0 0 NOTYPE LOCAL DEFAULT 19 __init_array_end
39: 000000000200da8 0 OBJECT LOCAL DEFAULT 21 _DYNAMIC
40: 000000000200d98 0 NOTYPE LOCAL DEFAULT 19 __init_array_start
41: 000000000000908 0 NOTYPE LOCAL DEFAULT 17 GNU_EH_FRAME_HDR
42: 000000000200f98 0 OBJECT LOCAL DEFAULT 22 GLOBAL_OFFSET_TABLE

```

- Analyse the program headers and sections

```

remnux@remnux: ~/Downloads$ readelf -l training-sample | more
Elf file type is DYN (Shared object file)
Entry point 0x680
There are 9 program headers, starting at offset 64

Program Headers:
Type           Offset             VirtAddr           PhysAddr
               FileSiz            MemSiz             Flags  Align
PHDR           0x0000000000000040 0x0000000000000040 0x0000000000000040
               0x0000000000001f8 0x0000000000001f8  R      0x8
INTERP         0x000000000000238 0x000000000000238 0x000000000000238
               0x00000000000001c 0x00000000000001c  R      0x1
[Requesting program interpreter: /lib64/ld-linux-x86-64.so.2]
LOAD           0x0000000000000000 0x0000000000000000 0x0000000000000000
               0x000000000000a78 0x000000000000a78  R E    0x200000
LOAD           0x000000000000d98 0x000000000200d98 0x000000000200d98
               0x00000000000032c 0x000000000000330  RW     0x200000
DYNAMIC        0x000000000000da8 0x000000000200da8 0x000000000200da8
               0x0000000000001f0 0x0000000000001f0  RW     0x8
NOTE           0x000000000000254 0x000000000000254 0x000000000000254
               0x000000000000044 0x000000000000044  R      0x4
GNU_EH_FRAME   0x000000000000908 0x000000000000908 0x000000000000908
               0x000000000000044 0x000000000000044  R      0x4
GNU_STACK      0x0000000000000000 0x0000000000000000 0x0000000000000000
               0x0000000000000000 0x0000000000000000  RW     0x10
GNU_RELRO      0x000000000000d98 0x000000000200d98 0x000000000200d98
               0x000000000000268 0x000000000000268  R      0x1

Section to Segment mapping:
Segment Sections...
00

```

- **Observations:** look for LOAD – actual location where the actual executable code is loaded. The code with RE permission will execute the malware. The code with RW stores file's global variable data and dynamic linking library is stored.
- Analyse the section to segment mapping

Section to Segment mapping:

```
Segment Sections...
00
01      .interp
02      .interp .note.ABI-tag .note.gnu.build-id .gnu
t.got .text .fini .rodata .eh_frame_hdr .eh_frame
03      .init_array .fini_array .dynamic .got .data
04      .dynamic
05      .note.ABI-tag .note.gnu.build-id
06      .eh_frame_hdr
07
08      .init_array .fini_array .dynamic .got
remnux@remnux:~/Downloads$
```

- **Observation:** .text section contains the executable code.