Network Security Assignment #2

JOHN THOMAS
P2CSN17017

PART A - Written Assignment -1

1. What is the difference between a binary file and a text file?

A text file stores data in ASCII characters i.e, with a maximum length of 255 characters . Each line of a text file is terminated (delimited) with a special character known as EOL (end of line) character. Some internal translations take place when this EOL character is read or written.

By default our files are created and opened in text mode.

A binary file contains information in the same format as it is held in memory i.e, (0 or 1). There is no delimeter for a line. Also, no translations occur in binary files. Binary files are faster and easier for a program to read and write than are text files.

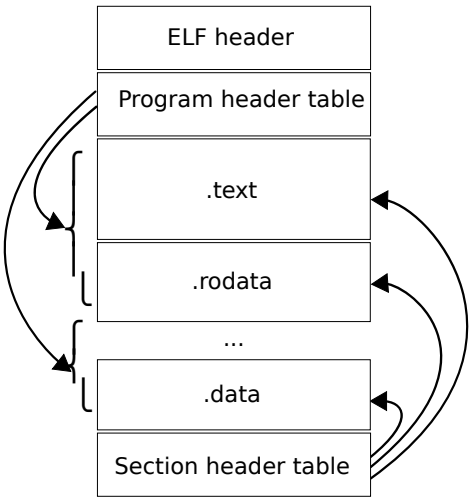
2. What is the ELF FILE Format?

In computing, the Executable and Linkable Format (ELF, formerly named Extensible Linking Format), is a common standard file format for executable files, object code, shared libraries, and core dumps.
Each ELF file is made up of one ELF header, followed by file data. The data can include:

Program header table, describing zero or more memory segment.

Section header table, describing zero or more section.

Data referred to by entries in the program header table or section header table.



3. What is a hexdump? What kind of information is stored in a hexdump?

In computing, a hex dump is a hexadecimal view (on screen or paper) of computer data, from RAM or from a file or storage device.In a hex dump, each byte (8-bits) is represented as a two-digit hexadecimal number.

Hex dumps are commonly organized into rows of 8 or 16 bytes, sometimes separated by whitespaces. Some hex dumps have the hexadecimal memory address at the beginning and/or a checksum byte at the end of each line.

4. Diagrammatically represent how the hex characters “DEADBEEF” will be represented in Big Endian and Little Endian formats.

Endianness means the order in which the bytes of a value larger than one byte are stored in memory.
Little-endian machines store the least significant byte on the lowest memory address (the word is stored little-end-first).
Big-endian machines store the most significant byte on the lowest memory address (the word is stored big-end-first).

Let the start address be 100 for storing hex characters.

In Big Endian

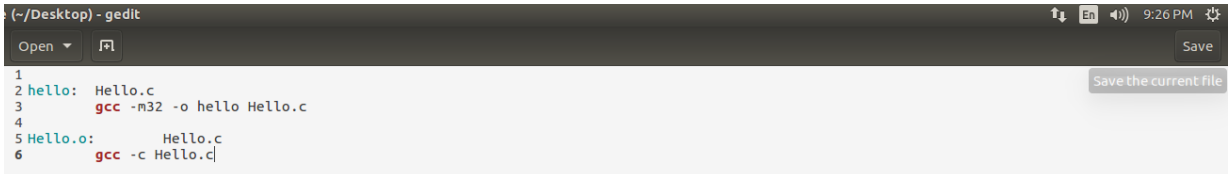
100	101	102	103	104	105	106
DE	AD	BE	EF			

In Little Endian

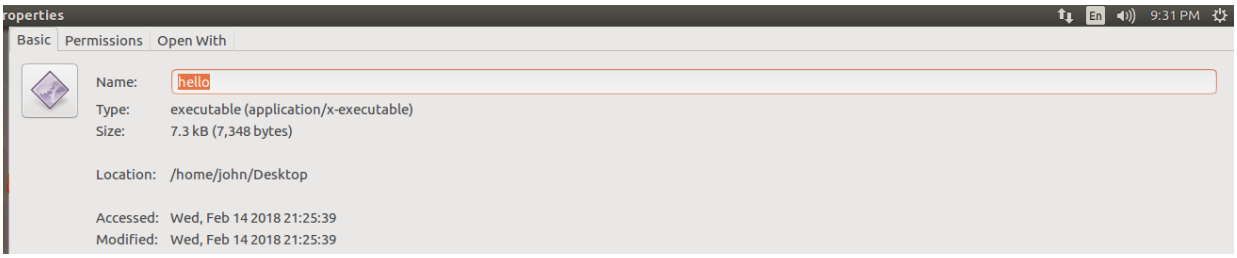
100	101	102	103	104	105	106
EF	BE	AD	DE			

PART B - Programming Assignment -2

1.Write a simple Hello World program in c. The program should have nothing but a #include and a print statement in main. Write a makefile to generate an executable called ‘hello’ Record the size in number of bytes of the hello world program

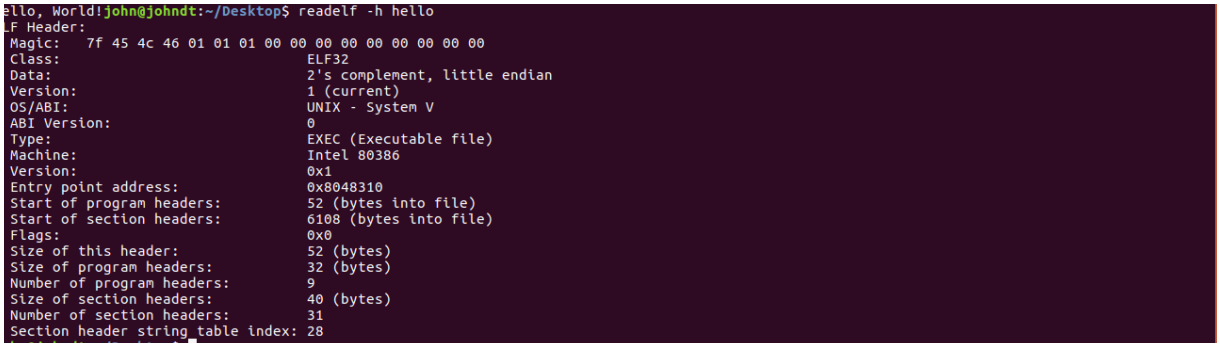


Size in bytes = 7.3 kB (7348 bytes)



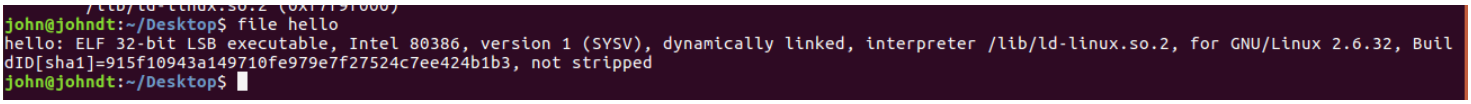
2. Read and dump the ELF header from the executable type the command

readelf -h hello

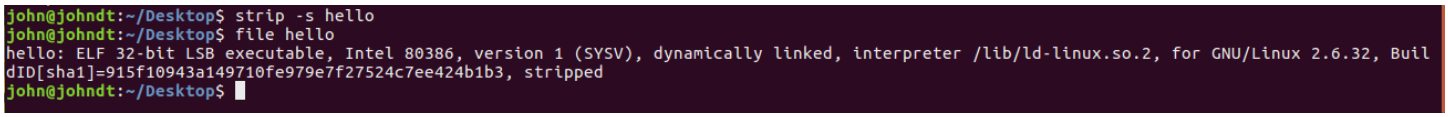


3. ldd hello

4. file hello



5.What does the strip command do?
strip –s hello

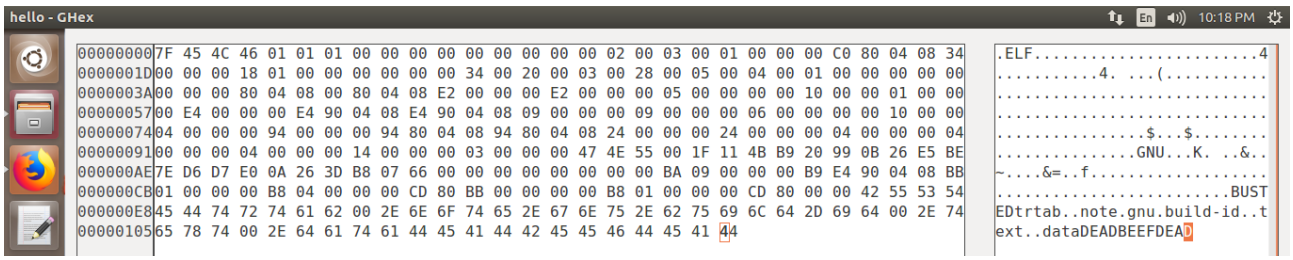


The symbol table can be stripped from an object file using -s option of strip command

6.,7.,8

```
john@johndt:~$ cd Desktop/
john@johndt:~/Desktop$ ls
a2.odt  a.out  example1a.tcl  hello  hello.asm  Hello.c  Makefile  out.nam
john@johndt:~/Desktop$ nasm -f elf hello.asmjohn@johndt:~/Desktop$ gcc -o m32 hello hello.o -nostartfiles -nostdlib -nodefaultlibs
hello.o: In function `main':
hello.asm:(.text+0x0): multiple definition of `main'
hello:(.text+0x0): first defined here
/usr/bin/ld: warning: Cannot create .note.gnu.build-id section, --build-id ignored.
/usr/bin/ld: i386 architecture of input file `hello' is incompatible with i386:x86-64 output
/usr/bin/ld: i386 architecture of input file `hello.o' is incompatible with i386:x86-64 output
/usr/bin/ld: warning: cannot find entry symbol _start; defaulting to 0000000000400110
collect2: error: ld returned 1 exit status
john@johndt:~/Desktop$ gcc -o -m32 hello hello.o -nostartfiles -nostdlib -nodefaultlibs
/usr/bin/ld: unrecognised emulation mode: 32
Supported emulations: elf_x86_64 elf32_x86_64 elf_i386 elf_iamcu i386linux elf_l1om elf_k1om i386pep i386pe
collect2: error: ld returned 1 exit status
john@johndt:~/Desktop$ gcc -m32 -o hello hello.o -nostartfiles -nostdlib -nodefaultlibs
/usr/bin/ld: warning: cannot find entry symbol _start; defaulting to 00000000080480c0
john@johndt:~/Desktop$ strip -s hello
strip: '-s': No such file
john@johndt:~/Desktop$ strip -s hello
john@johndt:~/Desktop$ ldd hello
        not a dynamic executable
```

9.



10.Record the number of bytes on the executable. It should have come down to less than 300 bytes

```
john@johndt:~/Desktop$ wc -c hello
282 hello
john@johndt:~/Desktop$
```

282 bytes