LAB 1 Solution

To find out documentation of specific tools use man <tool> . Use '/<txt> to search for a particular keyword. For next occurance use 'N'.

Part 1: The OS view

A. Tools

1. **top:** Top shows all the linux running processes. "-U | -u" is a command line option of top that display only processes with a user id or user name matching that given.To find a particular process use command "top -U <Username>"
2. **ps:** ps command shows the snapshot of the current process. "-e | -A " option can be used to select all processes. "-f" option for full-format listing (to find which user). To find all processes use the command "ps -ef".
3. **iostat:** iostat is a command useful for monitoring and reporting CPU and statistics related to devices. "%system" shows the percentage of CPU utilization that occurred while executing at the system level (kernel). Command - "iostat %system".
4. **strace:** strace command trace all system calls and signals. "-c" and "-e" option is useful for determining which system calls and signals are made by command respectively. "strace -c ls" shows all the system calls. "strace -e 'signal' top" shows signals made by the top command.
5. **lsof:** lsof is a tool used to list open files. "-u" option selects the listing of all open files of the user. "lsof -u <username>"
6. **lsblk:** list information about all available block devices such as hard disk, flash drives, CD-ROM etc. "-m" output info about device owner, group and mode. "lsblk -m" shows the permissions.

B. The proc file system -

The proc file system is a mechanism provided by Linux, for communication between user space and the kernel using the file system interface. Files in the /proc directory report values of several kernel parameters and also can be used for configuration and (re)initialization. The proc filesystem is nicely documented in the man pages, — man proc. Understand the system-wide procfiles such as meminfo, cpuinfo, etc and process related files such as status, stat, limits, maps etc.System related proc files are available in the directory /proc, and process related proc files are available at /proc/<process-id>/

1.
    a. "cat /proc/cpuinfo" : provides model name, address size -physical and virtual size. "lscpu" : provides architecture and byte order
    b. "cat /proc/cpuinfo" : provides #cores and #CPUs machine have. "lscpu" : provides #sockets
    c. "lscpu" : provides information of L1, L2, L3 cache sizes
    d. "cat /proc/meminfo" : MemTotal(main memory), MemFree(free main memory), SwapTotal(secondary memory), SwapFree(free secondary memory)

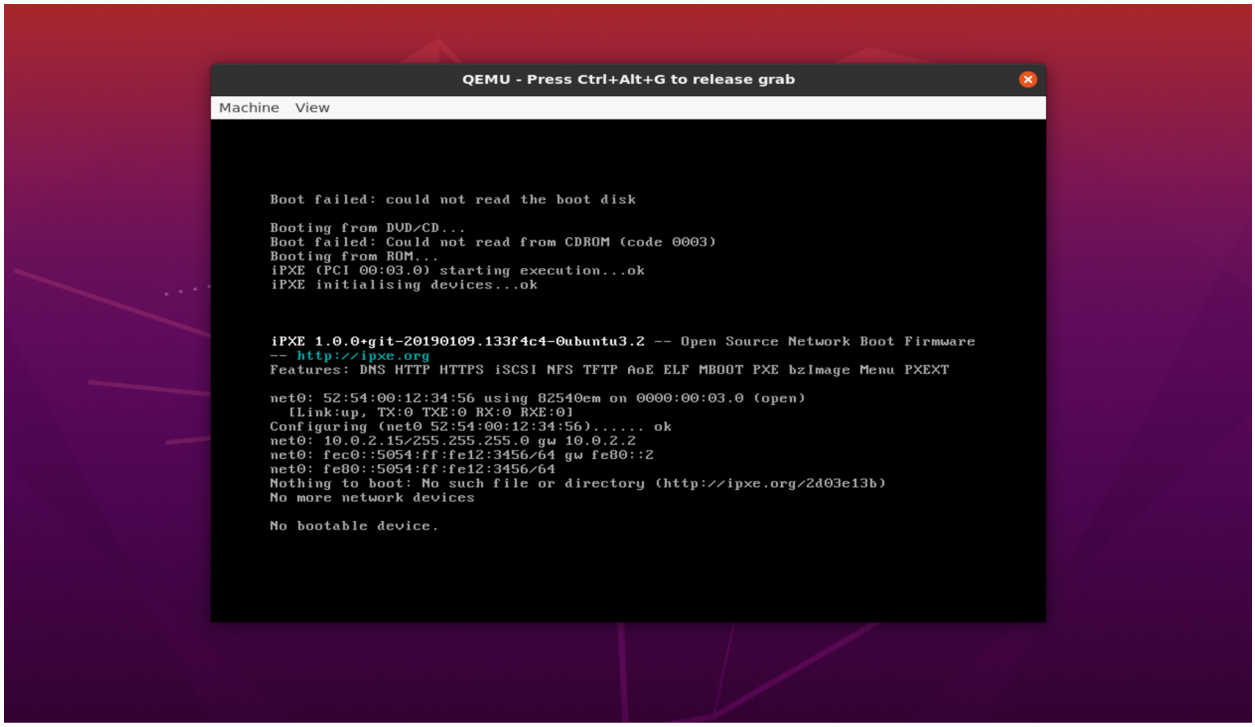
e. “top” : provides information on running (R), sleeping (S), stopped (T),zombie (Z) processes. “cat /proc/stat” : provides number of running processes
f. :”cat /proc/stat” : ctxt -number of context switches happened since system boot

2. VMRSS is the total main memory that the program is using; VMSize also includes memory that is allocated but may not be used and shared libraries, etc. Steps - “./memory_i.c” and then find process id by using “ps -ef” and then use command “cat /proc/<processid>/status”
   Observation:
   - Memory_1.c - VMsize - 6560KiB, VMRSS - 5440KiB
   - Memory_1.c - VMsize - 10464KiB, VMRSS - 8780KiB
   - Memory_1.c - VMsize - 6552KiB, VMRSS - 4844KiB
   - Memory_1.c - VMsize - 6556KiB, VMRSS - 5376KiB
   - We see there to be a sizable difference in memory 2.c, compared to others. This is because the size of the array in memory 2.c is double the size of the array in the other programs. Also, from memory 3 to memory 4, we see an increase in VMRSS due to more array accesses since more data has to be loaded into the memory. (in contrast when pages are swapped out, this decreases).
3. “./subprocess <rollnumber> & echo $!” - outputs the process ID. “pgrep” - command looks through the currently running processes and lists the process IDs. “-P” option in “pgrep” command only match processes whose parent process ID is listed.Then use “pgrep -P <processid>” to get list of all subprocesses
4. “strace ./<executablefile>” or “strace -c ./<executablefile>” strace identifies all the system calls made by the program. The first part of both the programs is similar (with only memory address differences). In the second part, we see system calls for reading and writing, as well as getting the PID in hello, but not in empty. strace -c can also be used for a summary of the system calls.
5. “./<executablefile>” - obtain process ID using “ps -ef” command. Then “lsof -p processid” provides the information of list of files which are opened by the program. “-p” selects the listing of files for the process ID provided.
6. “lsblk -a” command list empty devices and RAM disk devices.

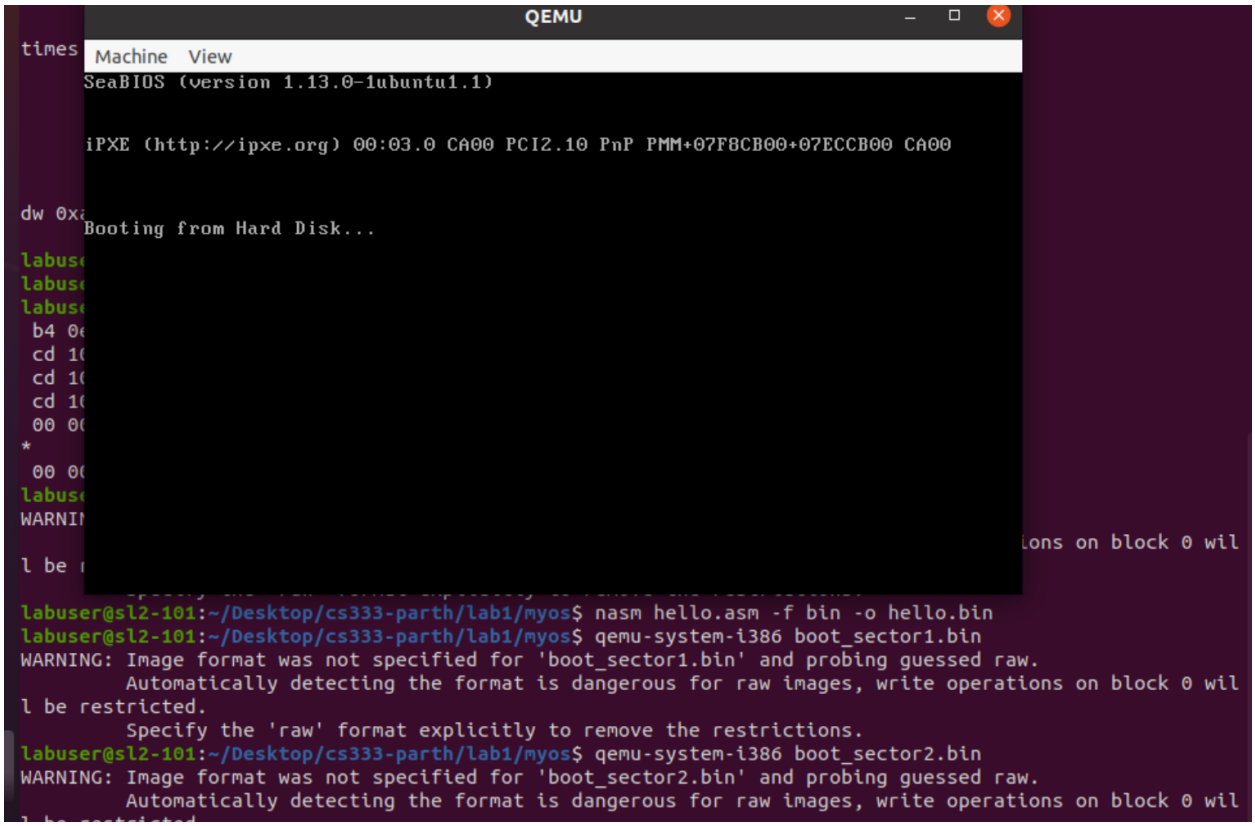
C. **Object files -** objdump -s <FileName> displays all the information from the object file . As password is hardcoded it will be present in read only data section can be found using : “objdump -s -j .rodata password.out” - “-j” option for displaying information only for section name, “-s” option for displaying the full contents of any sections requested.

Part 2: Booting unraveled:

1. boot_sector1.asm - output



boot_sector2.asm: output

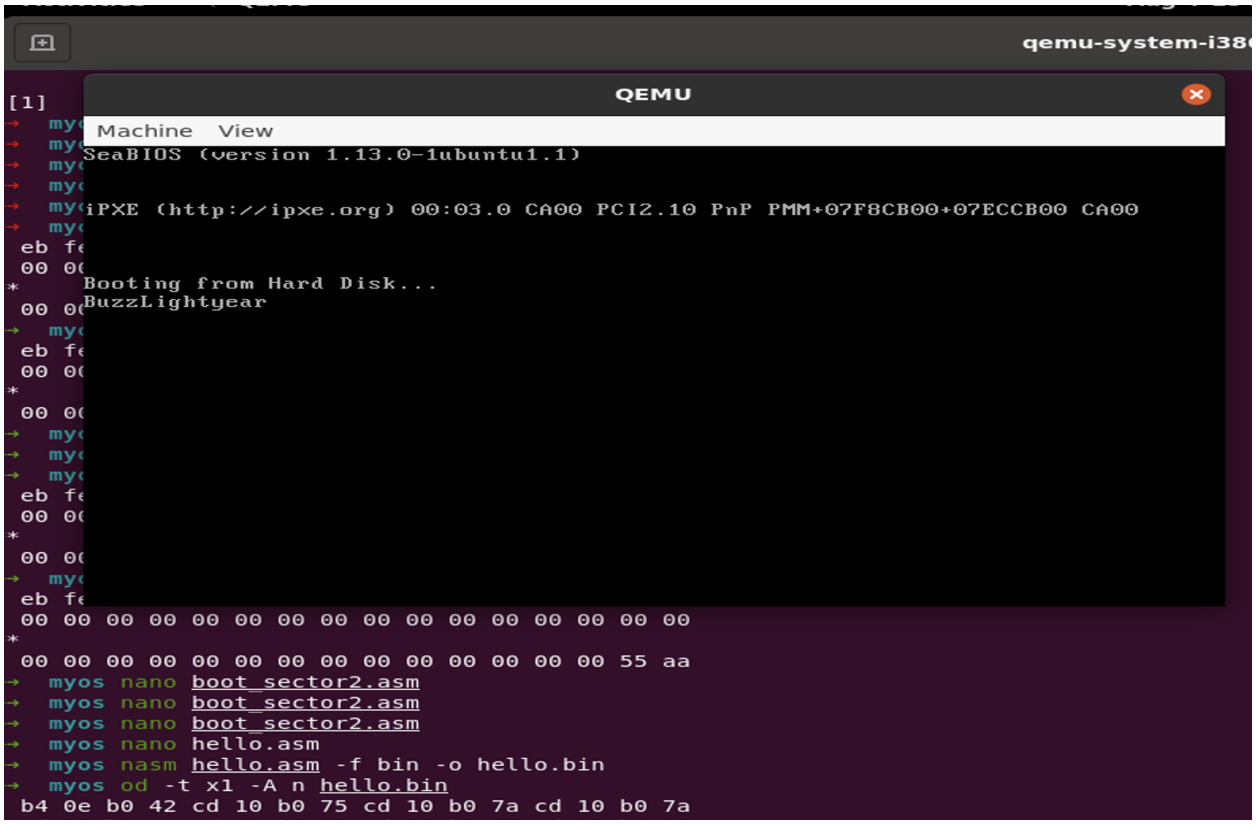


Converting the Assembly files into Binary files and then run:
$od -t x1 -A n boot_sector1.bin
$od -t x1 -A n boot_sector2.bin
See the difference between the 2 outputs. The first file ends at 00 00 and the second file ends in magic number i.e. aa 55(or 55 aa depending on little-endian or big-endian system), which tells the BIOS that the disk is a boot disk.

2. Writing the code:

```
 mov ah, 0x0e
 mov al, 'B'
 int 0x10
 mov al, 'u'
 int 0x10
 mov al, 'z'
 int 0x10
 mov al, 'z'
 int 0x10
 mov al, 'L'
 int 0x10
 mov al, 'i'
 int 0x10
 mov al, 'g'
 int 0x10
 mov al, 'h'
 int 0x10
 mov al, 't'
 int 0x10
 mov al, 'y'
 int 0x10
 mov al, 'e'
 int 0x10
 mov al, 'a'
 int 0x10
 mov al, 'r'
 int 0x10
 times 510 -( $ - $$ ) db 0
dw 0xaa55
```

Our program has to be 512 Bytes, with last bytes as magic number, so we fill all bytes till 510 as 00 and then, Define the magic number (0xaa55) at the end. Output -


qemu-system-i38
QEMU
Machine View
SeaBIOS (version 1.13.0-1ubuntu1.1)
iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+07F8CB00+07ECCB00 CA00
Booting from Hard Disk...
BuzzLightyear
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 55 aa
myos nano boot_sector2.asm
myos nano boot_sector2.asm
myos nano boot_sector2.asm
myos nano hello.asm
myos nasm hello.asm -f bin -o hello.bin
myos od -t x1 -A n hello.bin
b4 0e b0 42 cd 10 b0 75 cd 10 b0 7a cd 10 b0 7a