**Prerequisite:** git clone git://github.com/mit-pdos/xv6-public.git xv6

$ cd xv6

….xv6 $

**Follow the below steps in Skill9 folder.**

**Step-1:** **Syscall.h**

**$ nano syscall.h**

#define SYS\_backtrace 23

**Step 2:** **Syscall.c**

**$ nano syscall.c**

extern int sys\_backtrace(void);

[SYS\_backtrace] sys\_backtrace,

**Step 3: Sysproc.c**

**$ nano sysproc.c**

int

sys\_backtrace(void)

{

//struct proc \*curproc = myproc();

uint ebp, ret\_addr, next\_addr;

int count = 0;

struct trapframe \*tf = myproc()->tf;

cprintf("eax : 0x%x\n", tf->eax);

cprintf("ebx : 0x%x\n", tf->ebx);

cprintf("ecx : 0x%x\n", tf->ecx);

cprintf("edx : 0x%x\n", tf->edx);

cprintf("edi : 0x%x\n", tf->edi);

cprintf("esi : 0x%x\n", tf->esi);

cprintf("esp : 0x%x\n", tf->esp);

cprintf("ebp : 0x%x\n", tf->ebp);

cprintf("eip : 0x%x\n", tf->eip);

ebp = tf->ebp;

next\_addr = tf->eip;

while(next\_addr && next\_addr != (uint)-1) {

ret\_addr = \*(uint \*) (ebp + 4);

cprintf("#%d 0x%x\n", count++, ret\_addr);

ebp = \*(uint \*)ebp;

next\_addr = ret\_addr;

}

return 0;

}

**Step 4: Usys.S**

**$ nano usys.s**

SYSCALL(backtrace)

**Step 5: User.h**

**$ nano user.h**

int backtrace(void);

**Step 6: defs.h**

**$ nano defs.h**

int backtrace(void);

**Step 7: bt.c**

**$ nano bt.c**

#include "types.h"

#include "stat.h"

#include "user.h"

int baz() \_\_attribute\_\_((noinline));

int baz() {

int a;

a = backtrace();

return a + uptime();

}

int bar() \_\_attribute\_\_((noinline));

int bar() {

int b;

b = baz();

return b + uptime();

}

int foo() \_\_attribute\_\_((noinline));

int foo() {

int c;

c = bar();

return c + uptime();

}

int

main(int argc, char \*argv[])

{

foo();

exit();

}

**Step 8: Makefile**

**$ nano Makefile**

**UPROGS:** \_bt\

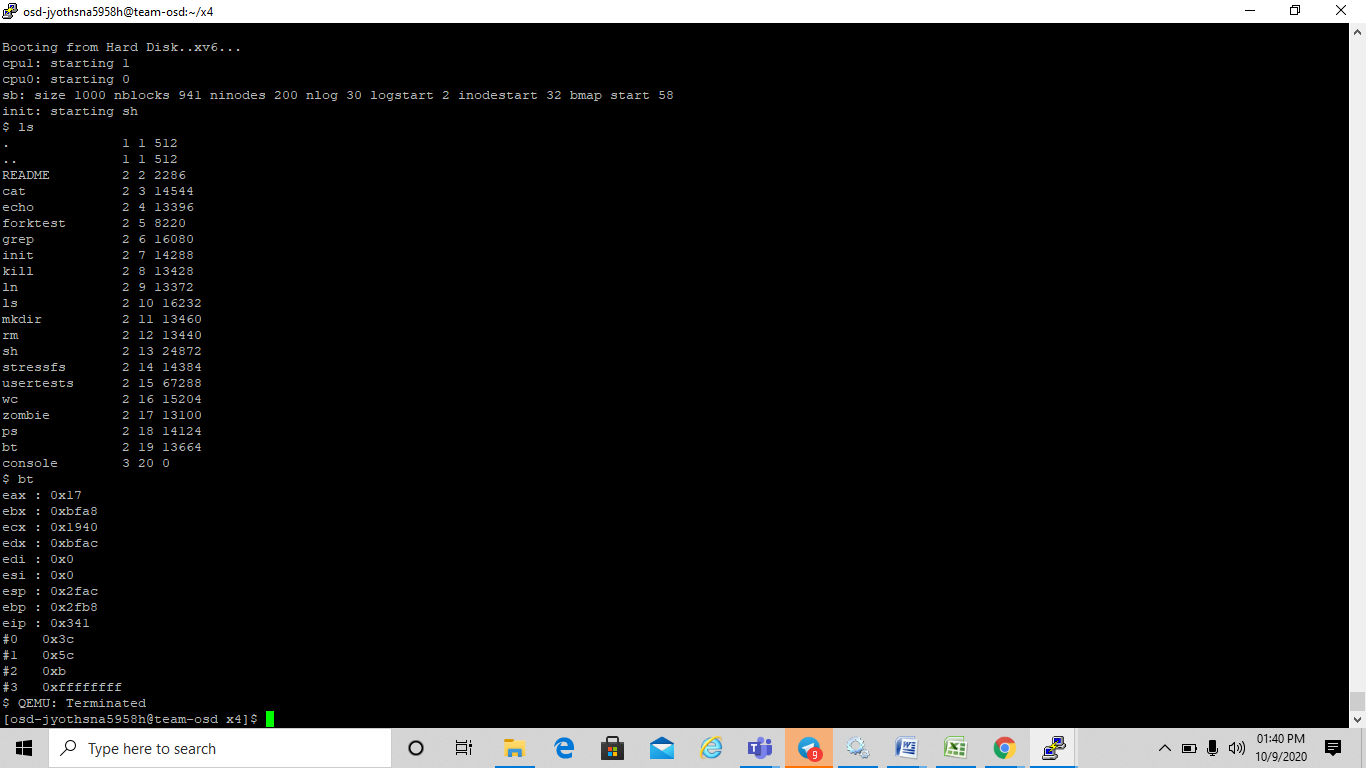
**EXTRAS:** bt.c

**Step 9:** make clean

**Step 10:** make

**Step 11:** make qemu-nox

**Output:**

****