Project 1 Report

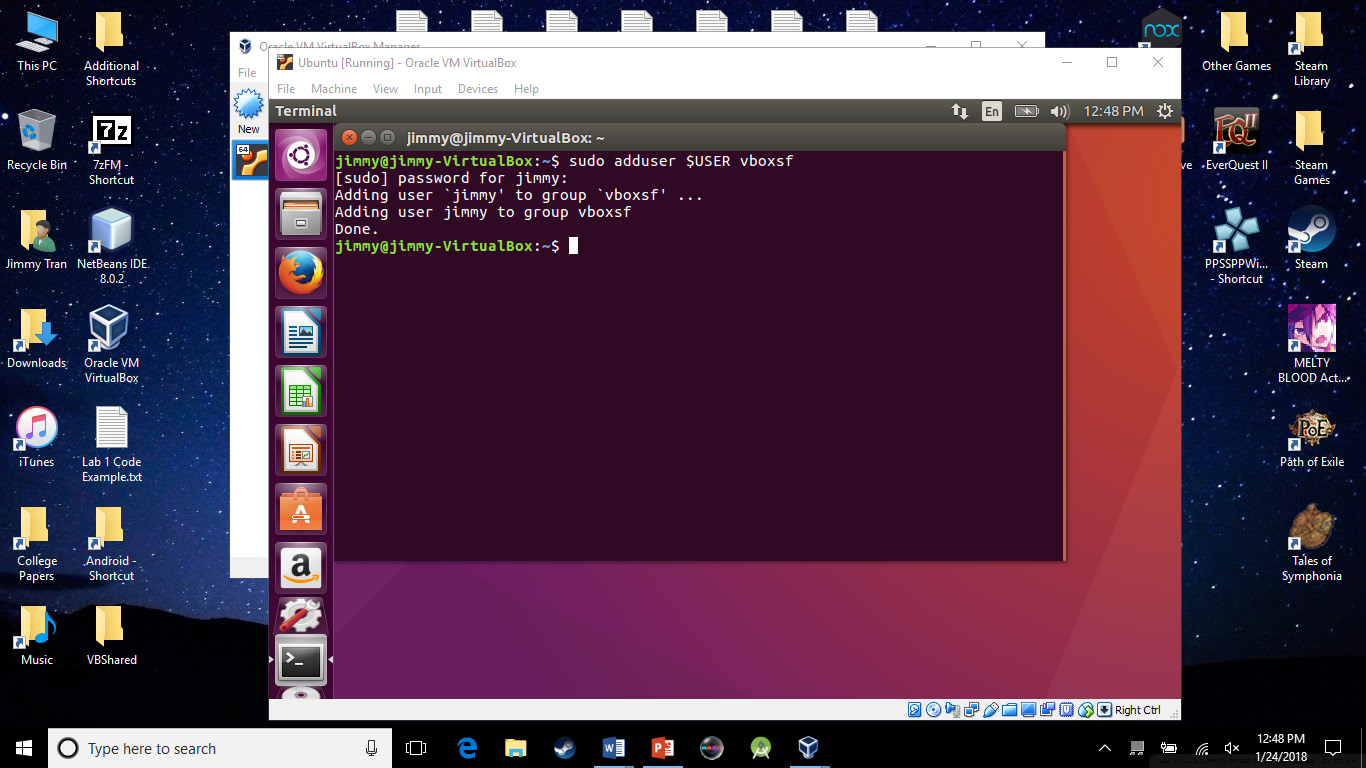
CSC 4320 Operating Systems

Spring 2018

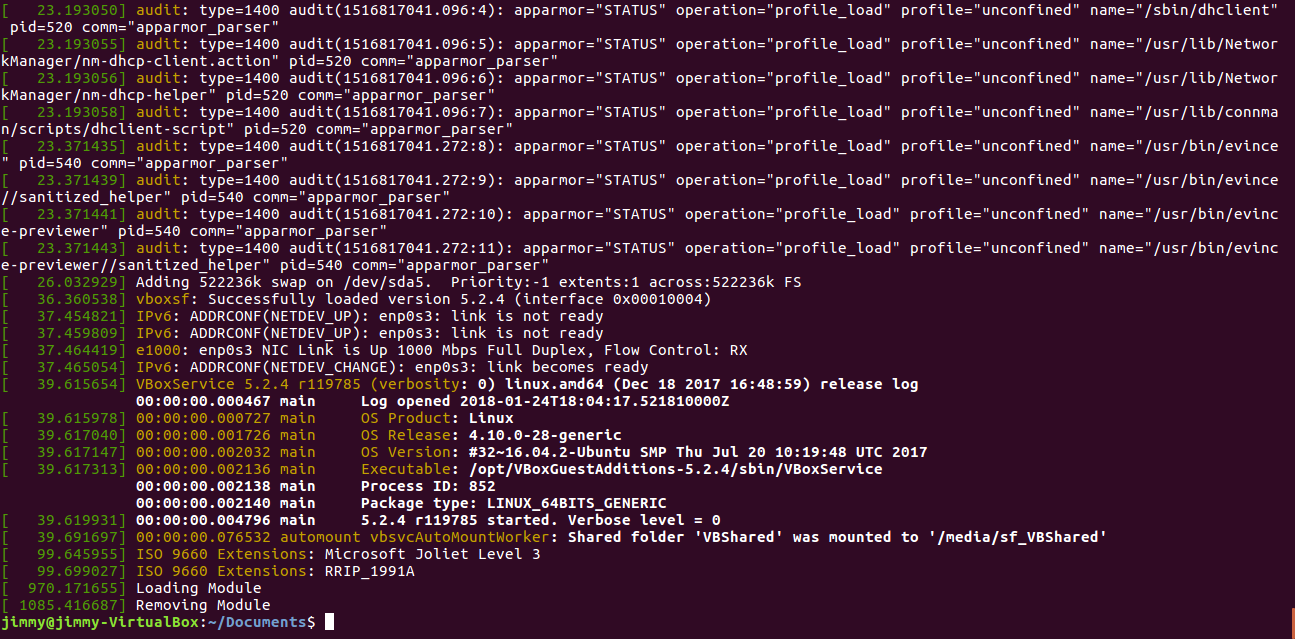
Name: Jimmy Tran

Email: jtran25@student.gsu.edu

Part 1: Screenshot below

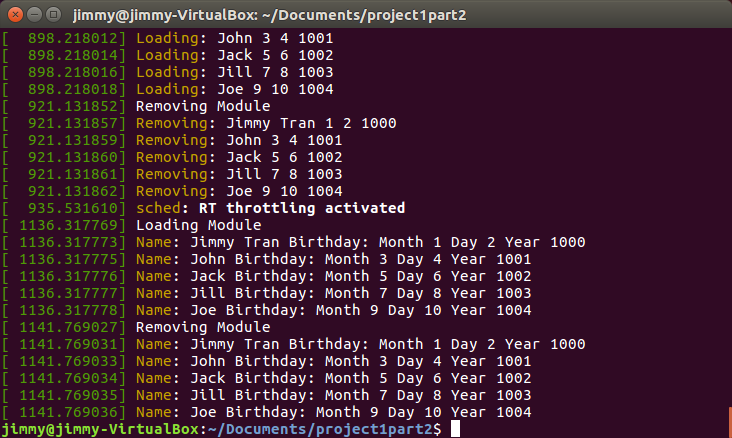


Part 2: Screenshot below for module loading and removing.



Part 3:

1 and 2) Screenshot for module load and removal combined . I know the years don’t make sense. Sue me for picking random numbers. It works in any case. The above loads and removes are there because I forgot the labels for the variables. Ignore those.



3) Source Code:

#include <linux/init.h>

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/list.h>

#include <linux/slab.h>

struct birthday

{

char \*name;

int month;

int day;

int year;

struct list\_head list;

};

/\*\*

\* The following defines and initializes a list\_head object named birthday\_list

\*/

static LIST\_HEAD(birthday\_list);

int simple\_init(void)

{

printk(KERN\_INFO "Loading Module\n");

/\* Create a linked list containing five struct birthday elements\*/

/\* NOTE:THE NAME OF FIRST STRUCT BIRTHDAY SHOULD BE YOUR OWN NAME \*/

struct birthday \*person1;

struct birthday \*person2;

struct birthday \*person3;

struct birthday \*person4;

struct birthday \*person5;

//first element

person1=kmalloc(sizeof(\*person1),GFP\_KERNEL);

person1->name="Jimmy Tran";

person1->month=1;

person1->day=2;

person1->year=1000;

INIT\_LIST\_HEAD(&person1->list);

//second element

person2=kmalloc(sizeof(\*person2),GFP\_KERNEL);

person2->name="John";

person2->month=3;

person2->day=4;

person2->year=1001;

INIT\_LIST\_HEAD(&person2->list);

//third element

person3=kmalloc(sizeof(\*person3),GFP\_KERNEL);

person3->name="Jack";

person3->month=5;

person3->day=6;

person3->year=1002;

INIT\_LIST\_HEAD(&person3->list);

//fourth element

person4=kmalloc(sizeof(\*person4),GFP\_KERNEL);

person4->name="Jill";

person4->month=7;

person4->day=8;

person4->year=1003;

INIT\_LIST\_HEAD(&person4->list);

//fifth element

person5=kmalloc(sizeof(\*person5),GFP\_KERNEL);

person5->name="Joe";

person5->month=9;

person5->day=10;

person5->year=1004;

INIT\_LIST\_HEAD(&person5->list);

//build linked list

list\_add\_tail(&person1->list,&birthday\_list);

list\_add\_tail(&person2->list,&birthday\_list);

list\_add\_tail(&person3->list,&birthday\_list);

list\_add\_tail(&person4->list,&birthday\_list);

list\_add\_tail(&person5->list,&birthday\_list);

/\* Traverse the linked list \*/

struct birthday \*ptr;

list\_for\_each\_entry(ptr,&birthday\_list,list){

/\* on each iteration ptr points \*/

/\* to the next birthday struct \*/

printk(KERN\_INFO "Name: %s Birthday: Month %d Day %d Year %d\n",ptr->name,ptr->month,ptr->day,ptr->year);

}

return 0;

}

void simple\_exit(void) {

/\* Remove the elements from the linked list and return the free memory back to the kernel \*/

printk(KERN\_INFO "Removing Module\n");

struct birthday \*ptr,\*next;

list\_for\_each\_entry\_safe(ptr, next, &birthday\_list,list){

/\* on each iteration ptr points \*/

/\* to the next birthday struct \*/

printk(KERN\_INFO "Name: %s Birthday: Month %d Day %d Year %d\n",ptr->name,ptr->month,ptr->day,ptr->year);

list\_del(&ptr->list);

kfree(ptr);

}

}

module\_init( simple\_init );

module\_exit( simple\_exit );

MODULE\_LICENSE("GPL");

MODULE\_DESCRIPTION("Kernel Data Structures");

MODULE\_AUTHOR("SGG");