Virginia Tech ❖ Bradley Department of Electrical and Computer Engineering

ECE 4984 / 5984 Linux Kernel Programming
Fall 2017

Small Project: Kernel Modules and Data Structures

1 Introduction

The goal of this project is to develop your first kernel module, and to study the manipulation of the following frequently-used kernel data structures: a linked list, red-black tree, hash table, radix tree, and bitmap.

The following concepts from the course will be put in practice in that project:

- Kernel module development;
- Kernel data structures.

2 Requirements

The requirements for this project are to write a *single* Linux kernel module manipulating the above-mentioned kernel data structures. The module takes one string parameter `int_str`, which is the arbitrary number of integers between 0 to 1000 (e.g., `insmod <module name> int_str='"4 98 45 984"'`). The module should compile with the Linux kernel v4.12.

The module should include functions manipulating the following data structures:

Linked lists : 1) create a linked list containing the integers in `int_str`; 2) print on the kernel log the content of the list using the list iteration functions; and 3) destruct the list and free its content.

Red-black trees : 1) create a rbtree, which is indexed by integer numbers; 2) insert integer numbers in `int_str` to the rbtree; 3) look up the inserted numbers and print them out; and 4) remove all inserted numbers in the rbtree.

Hash table : 1) create a hash table, of which the number of buckets is 2^{14}; 2) insert integer numbers in `int_str` to the hash table; 3) iterate the hash table and print out all inserted integer numbers; 4) look up the inserted numbers and print them out; 5) remove all inserted numbers in the hash table; and 6) destruct the hash table.

Radix tree : 1) create a radix tree, which is indexed by integer numbers; 2) insert integer numbers in `int_str` to the radix tree; 3) look up the inserted numbers and print them out; 4) tag all odd numbers in the radix tree; 5) look up all tagged odd number using `radix_tree_gang_lookup_tag`; and 6) remove all inserted numbers in the radix tree.

Bitmap : 1) create a bitmap, which is large enough to represent numbers between 0 to 1000; 2) set bits corresponding to integer numbers in `int_str`; 3) print all bits which are turned on; and 4) clear all bits in the bitmap.

3 Results to be handed

The deadline is **09/27 11:59 PM EDT**. A *tarball* containing source code and `Makefile` should be submitted. The tarball should follow the naming convention: `<VT ID>.project2.tar.gz` (e.g., `johndoe.project2.tar.gz`)

4 Project TA office hours

- *TA*: Anthony Carno
- *Date and time*: Monday:3-5pm, Wednesday:3-5pm
- *Email*: `acarno@vt.edu`
- *Location:* 460 Durham