2017-09-28

Operating Systems, 5dv171, Fall 2017

Project Linux Kernel Module - part 1

Teacher
Jan-Erik Moström

Teaching assistent
Adam Dahlgren

Group members
Igor Ramazanov, ens17irv@cs.umu.se
Erik Ramos, c03ers@cs.umu.se
Bastien Harendarczyk, ens17bhk@cs.umu.se

Gitlab repository
https://git.cs.umu.se/c03ers/5dv171-project

Thursday 28th September, 2017

User's guide

The project consists of two parts - the kernel module and the user space test program. All files required to build both parts can be found in the gitlab repository.

To build the project, begin by cloning the repositorybyt typing the following in a terminal:

```
git clone https://git.cs.umu.se/c03ers/5dv171-project
```

Problem description

The goal with this project is to write a Linux Kernel Module (LKM) that allows different processes running on the same linux machine, to access shared data through key-value mappings. The system must be robust enough, that multiple processes, running on multiple processors, will not accidentally corrupt the stored data.

This first part of the project report describes how the LKM represents these mappings in kernel space, as well as how the user space applications can access them.

Solution

The following three methods of user to kernel space communication were considered for this project:

1. Device files
2. Proc files
3. Netlink sockets

Using one of the filesystems to solve the problem was appealing, primarly because much of the heavy lifting would be done by system calls aleady available in Linux. However, since device files usually repressent an actual, physical device

and proc files are meant to contain information about processes, none of these seemed like a really good fit.

Because the kernel module would provide user space applications with a service, in much the same way that an Internet server would to its clients, Netlink sockets seemed like a much better fit. They would however, be much more complicated to set up and use.

System limitations

Discussion