Operating Systems
Semester 1, 2015
Eike Ritter

The University of Birmingham
30 October, 2015

Assessed Exercise 3

Deadline: Friday 20 November, 4pm

The Task

Write an extension to the linux firewall which makes it possible to specify which programs are allowed use which outgoing port.

More precisely, you should write a user space program and a kernel module.

- A firewall rule consists of a port number and a filename (the full path) of a program separated by a space, meaning that the corresponding program is allowed to make outgoing connections on this TCP-port. If there is no rule for a given port, any program should be allowed to make outgoing connections on this port. If a connection is not allowed, it should be immediately terminated.

- The kernel module processes the packets and maintains the firewall rules, and displays the firewall rules via `printk` in `/var/log/kern.log`. The output should be:

  ```
  Firewall rule: <port> <program>
  ```

- When the kernel module is unloaded, the firewall extensions should be deleted.

- The user space program, which must be called `firewallSetup`, has commands firstly for triggering the listing of the firewall rules in `/var/log/kern.log`, and secondly for setting the firewall rules. A new set of firewall rules overrides the old set (no appending). You should use the file `/proc/firewallExtension` for communication between the user program and the kernel.

- If replacing the set of firewall rules fails for any reason, the old set of firewall rules should be retained.

- To make marking easier, there should be two ways of calling the user space program. The first one is

  ```
  firewallSetup L
  ```

  where `firewallSetup` is the program name. This way of calling the user space program causes the firewall rules to be displayed in `/var/log/kern.log` as specified above.

  The second way of calling the program is

```
firewallSetup W <filename>
```

where `<filename>` is the name of the file containing the firewall rules. This file contains one firewall rule per line. `firewallSetup` should check (via the `stat`-system call) whether the filename in the firewall rule denotes an existing executable file. If there is any error in the syntax or any filename does not an executable file, this program should abort with the messages `ERROR: Ill-formed file` and `ERROR: Cannot execute file` respectively.

- Your kernel code may assume that only well-formed files are written by `firewallSetup`.
- Only one process should be allowed to open the `/proc/firewallExtension`-file at any give time. If a second process tries to open this file it should receive the error message `-EAGAIN`.
- You need to ensure that you handle concurrency in the kernel correctly. In particular, any number of outgoing connections may be started at any time, hence several instances of the procedures handling the packets may be executed at the same time. It is very important that you maximize the degree of parallelism. In particular, your critical sections should be as short as possible.
- We recommend using the APIS for linked lists, locking and the Basic C Library functions provided by the kernel.
- Please be careful to check all buffer boundaries (in particular for `strcpy` and `strcat`). Sometimes `strncpy` and `strncat` are the better alternatives.

Marking Scheme

Please use the canvas for submitting your code. Please submit only the source files you have written yourself. We will compile and run your code on the virtual machine and mark it accordingly. Please in particular note that we will use the compiler option introduced in the lecture and deduct 6 marks immediately if there is any compiler error or warning.

We will award marks as follows:

- 5 marks for correctly extending the firewall.
- 5 marks for correct handling the list of firewall rules.
- 5 marks for correct handling the concurrency.
- 5 marks for the correct functioning of the user space program.

A basic test script is provided via the module webpage. Further tests will be carried out during marking.