Operating Systems Project Document
Project2, Part2

AmirMahdi Daraei
Student ID: **99105431**

June 29, 2024

Add Syscall to Kernel

1 Introduction

This document outlines the requirements and implementation details of the functionalities related to network management in the shell developed. The implementation covers various system call functionalities, such as toggling network interface status and measuring traffic, to enhance our custom shell capabilities.

2 Requirements

The project demands the development of a system call to control and measure network metrics directly from our shell. This includes:

- Incoming and outgoing traffic volume
- Number of connections (incoming and outgoing)
- Duration of activity since the last reset

These metrics are essential for understanding the network interactions of the system, providing real-time data to the user through the shell environment.

3 System Call Implementation

The system call will facilitate direct communication with the network interface, allowing the shell to:

- Display network traffic statistics.
- Clear and restart traffic measurements.
- Disconnect and reconnect the network interface.

This functionality is achieved by interfacing directly with the kernel through added system calls, which handle specific network-related tasks.

3.1 Setup for Development

Before implementing the new system calls, the development environment is prepared by updating the system and installing the necessary packages for compiling the Linux kernel.

```
sudo apt update && sudo apt upgrade -y
sudo apt install build-essential libncurses-dev libssl-dev libelf-dev bison flex -y
sudo apt autoclean && sudo apt autoremove -y
```

These commands ensure that the system is up-to-date and has all the necessary tools to build and modify the Linux kernel.

3.2 Downloading and Preparing the Kernel

Download the kernel source code, and necessary preparations are made to add custom system calls. first of all, extract your kernel version by using `uname -r` command and then download the nearest version to yours.

```
wget -P ~/ https://mirrors.edge.kernel.org/pub/linux/kernel/v6.x/linux-6.5.1.tar.gz
tar -xvf ~/linux-6.5.1.tar.gz -C ~/
```

This section involves downloading the latest stable version of the Linux kernel and extracting it to a working directory.

3.3 Adding System Calls

System calls are added to the kernel source. This involves modifying source files to include new syscall definitions, and updating syscall tables to reflect these changes.

```
vim kernel/fork.c  % Add custom syscall implementation here

SYSCALL_DEFINE1(netcall, int, functionality){
    switch (functionality) {
        case 1:
            pr_emerg("NETWORK DOWN!");
            break;
        case 2:
            pr_emerg("NETWORK UP!");
            break;
        case 3:
            pr_emerg("TIMER RESET!");
            break;
        case 4:
            pr_emerg("NETWORK INFO:");
            break;
            }
    return 0;
    }
```

Now we should modify some files of the kernel to identify our syscall

Add this line

```
asmlinkage long sys_netcall(int);
```

to this path:

```
include/linux/syscalls.h
```

for adding the prototype of your syscall to the kernel.

Add this line

```
548 common  netcall         sys_netcall
```

to this file

```
arch/x86/entry/syscalls/syscall_64.tbl
```

for adding your syscall to the syscall table of the kernel.

Note that this file is different depending on the architecture of the processor.
Also, the number 548 may be different depending on the kernel version.

3.4 Compiling and Installing the Kernel

Once the system calls are defined, by running the following commands, the kernel will compile and install.

```
make menuconfig  % Save and exit configuration
make -j $(nproc) && sudo make modules_install install
```

This process compiles the new kernel with the added functionalities and installs it to the system. **A reboot is required to apply the changes.**