How to talk to inet server

Note: this docment is fully tested only on Minix3.1.2a.

In this document, we introduce a method to let user level program to talk to inet server.

I. Problem with system call

Recall the process of system call, refering to
http://www.cis.syr.edu/~wedu/seed/Labs/Documentation/Minix3/System_call_sequence.pdf
We can see the real system call happens in this function call:
_syscall(FS, CHMOD, &m)

This function executes 'INT 80' to trap into kernel. Look at the parameter it passs to kernel. 'CHMOD' is a macro which is merely the system call number. 'FS' is a macro which indicates the server which handles the chmod system call, in this case 'FS' is 1, which is the pid of file system server process.

Now your might ask 'why can we hard code the pid of the process? Won't it change?' Yes, normally the pid of process is unpredictable each time the system boots up. But for *fs, pm, rs, init* and other processes which is loaded from system image at the very beginning of booting time, it is not true. In minix, you can dump the content of system image by pressing 'F3', then dump the current running process table by pressing 'F1'. What do you find? The first 12 entries in current process table is exactly the ones in system image with the same order. So the pids of these 12 processes will not change.

Inet is different. It is not in the system image, so it is not loaded into memory in the very first time. Instead, it is started by script */usr/etc/rc*. So the pid of inet can change if we start an additional server before *inet*. Thus, we can not simply use _syscall(INET, CHMOD, &m) to communicate with *inet*. We need another way.

II. ioctl

What's ioctl, literally it is i/o control, it is a mechanism in Minix and also a system call's name. Almost every driver for a device has its function to handle ioctl request, in *inet* , it is *ip_ioctl()* in */usr/src/servers/inet/generic/ip_ioctl.c.* When system call ioctl() is called, fs will handle the system call and forward the ioctl request to corresponding server which is considered to be the driver of the device you want to do i/o control to. In our case, when *inet* starts up, it register itself as the driver for ip device, then if we do ioctl to ip device, and fs will forward the request to *inet*. You'll know this mechanism better by looking at */usr/src/commands/simple/pr_routes.c* and *ip_ioctl()* in */usr/src/servers/inet/generic/ip_ioctl.c*

III. Conclusion

Ioctl is a quite common tool in Unix like system. In Linux/Unix, we can use ioctl to let user to talk to LKM(loadable kernel module) .