Assignment 4

He Fu(SID: 012543440)
Rui Guo(SID: 012478973)

1. Which capabilities API (seccomp-bpf, AppArmor, or SELinux) did you choose? Why did you make that choice?
Answer:
We choose seccomp-bpf for our assignment.
In Linux, a large number of system calls are directly exposed to user-mode programs. However, not all system calls are needed, and misuse of system calls by unsafe code can pose a security threat to the system. The seccomp security mechanism enables a process to enter a "safe" running mode where it can only call the system calls from the white list, otherwise, the process will be terminated.

2. What was the process you used to ascertain the list of system calls required by each program?
Answer:
We applied strace at first in order to determine the system calls in the program, however, we found that the child process cannot be monitored. Thus we used strace -f finally and both parent and child processes can be monitored.

3. What system calls are needed by each?
Answer:
Server:
execve
brk
access
openat
fstat
mmap
close
read
arch_prctl
mprotect
munmap
write
socket
setsockopt
bind
listen
accept
clone
wait4

exit_group
seccomp
prctl
getcwd
chdir
chroot
setuid
sendto

Client:
execve
brk
access
openat
fstat
mmap
close
read
arch_prctl
mprotect
munmap
write
socket
exit_group
sendto
connect

4. What happens when your application calls the prohibited system call? What is the application behavior that results from the call?
Answer:
The seccomp-bpf would help to filter out system calls based on our white list, and when the application calls the prohibited system call, the application would stop and output “bad system call”, but would not specify which system call cause it.