

CSIS0230A Principles of Operating Systems (Class A)

Tutorial 13

Unix Security

Setup a simple message board system for your Linux machine. The requirements:

1. Every user can read the message board through the `msgboard` program;
2. Every user can append a message to the message board through “`msgboard`” program.

The message board is stored as a text file at `/var/local/message/data`, which is owned by `root` and cannot be written by any other user (i.e. its permission should be `-rw-r-r-`). The `msgboard` program is located at `/usr/local/bin/`, which any user can run. The program:

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <cstring>
#include <cerrno>
#include <unistd.h>
#include <wait.h>
using namespace std;
const char* DATA_FILE = "/var/local/message/data";
inline void die(char *msg) { cerr << msg << strerror(errno) << endl; exit(1); }

int main() {
    for (;;) {
        char c = 0;
        cout << "Read messages, Write a message or Quit (r/w/q)? ";
        cin >> c;
        while (cin && (c != 'r' && c != 'w' && c != 'q')) {
            cout << c << "Invalid input! Try again (r/w/q): ";
            cin >> c;
        }
        if (!cin) return 0;
        cin.ignore(1000000, '\n'); // eat the remainder of the line
        if (c == 'r') { // use "less" to display message board
            if (fork() == 0) { // child
                execlp("less", "less", DATA_FILE, 0);
                die("Cannot exec: ");
            }
            wait(0);
        } else if (c == 'w') {
            cout << "Type your message. Press enter 3 times when finished\n";
            ofstream of(DATA_FILE, ios::app); // in append mode
            if (!of.is_open())
                die("File open failed: ");
            for (int entercount=0; entercount<3; ++entercount) {
                if (!cin.get(c)) return 0;
                of.put(c);
                if (c!='\n')
                    entercount--;
            }
        } else if (c == 'q')
            return 0;
    }
}
```

Your tasks

1. Set up the message board system using the above program, which source can be found in the `/root` directory. The `root` password is the same as that of the `os` account.

(Hint: you are required to set the `setuid`-bit for the compiled `msgboard` program. Read the man page of `chmod(1)` to see how to do this.) The following steps check whether your system has been setup correctly:

- a. login as normal user (`os`)
 - b. try to overwrite the data file by `cat > /var/local/message/data`. (A normal user should not be able to arbitrarily write data file)
 - c. run the `msgboard` program and add a new message. (You should be able to add a new message while using a normal user account)
 - d. use the same program to view the message. (You should find your message have been added to the message board)
2. There is a security hole in the system: the program executed might not be the `less` program of the system, and a normal user can subsequently execute any command with root privilege. Where is the security hole, and how to effect an attack? How to make sure that the correct `less` is executed?
 3. There is another problem in the system: `less` is too powerful a program to be used this way. Read the man page of `less`, in particular the section about security. What type of attacks can be done to the current system? How to solve the problem? (Hint: is special privilege really needed for reading the message board?)

Further discussions (think after the tutorial):

- The current system has a race condition: if two users tries to write a message at the same time, the message board can get corrupted. Discuss how to deal with the problem by using file `flock(2)`.
- Naively using `flock(2)` may make users wait unnecessarily long if they want to write to the file. Explain why, and suggest how temporary files can be used to deal with the problem.
- How you can avoid two users using the same temporary file? (See man page of `mktemp(3)` and `mkstemp(3)`.)