Security I Supervisions

Example Sheet 2[*]

Petar Veličković

Easter Term 2016

Authentication, Access Control, Software Security

Once again, most of the questions presented here have been relevant when I did the course in 2013/14. If there's anything irrelevant feel free to skip it (but do let me know of such offenders, so I can remove them for the following years).

1. Consider the standard POSIX filesystem access control mechanism:
   a) Under which conditions can files and subdirectories be removed from a parent directory?
   b) Many Unix variants implement an extension known as the sticky bit. What is its function?
   c) On a POSIX system that lacks support for the sticky bit, how could you achieve an equivalent effect?
2. Explain the concept of a Trusted Computing Base and outline its meaning in the context of the access control provided by a typical Unix workstation.
3. How can you implement a Clark-Wilson policy under Unix?
4. If the Bell-LaPadula system is fully implemented, a malware process which acquires root privileges will be able to read all files on the system, but not be able to write to any files which regular users can read. However, there are many other ways of conveying information. For each of the following possible covert channels, discuss how a signal process (high privilege) might communicate secret bits to an untrusted spy process, the rough efficiency of the channel, and what steps the system might take to eliminate or narrow the covert channel.
   a) Network traffic volume
   b) Power consumption

[*]Special thanks to Markus Kuhn and Rubin Xu (from whose sheets this is heavily based on).

c) Processor cache contents

d) File system fragmentation

5. You and your roommates have had trouble keeping track of how much money is owed to whom, so you decide to build a computer system to keep track. For example, if Aisha pays £5 for Bharat's lunch, and Bharat pays £5 for Carlos' concert ticket, your system should allow Carlos to simply pay Aisha £5 and settle all debt. You are so awestruck by the Clark-Wilson security policy model that you decide to incorporate its principles into your system.

   a) What invariants should your system protect? How is this similar to the banking systems described in class?

   b) What items are unconstrained (UDIs) and which items are constrained (CDIs)?

   c) Describe a simple transaction procedure (TP) for recording a payment between two people.

   d) In the real world, what constraints might you place about which principals (roommates) can edit which UDIs?

6. Briefly describe *three* common software vulnerabilities, and explain a stack buffer overflow attack in greater detail.

7. Solve 2011 Paper 4 Question 8.

8. Little PetarV is a terrible programmer. Despite the fact he hasn't thoroughly followed his computer security course, he decided to implement his own secure authentication system using C. You've come across a snippet of his program's main function (given on the next page). Find as many ways as possible in which security of this system can be broken, outlining possible fixes (not necessarily only to the provided code fragment!).

```c
int main()
{
    char uname[150];
    char qry[150];
    char pwd[5];
    char master[25];
    int ok = 0;
    char *path_to_db = system("getdb '/etc/db_id.txt'");
    printf("Welcome to PetarV's totally secure system 9000!\n");
    printf("Now with two-step security!\n");
    printf("Enter your username.\n");
    scanf("%s", uname);
    sprintf(qry, "SELECT pwd FROM users WHERE username = %s", uname);
    char *ref = sql_execute(qry, path_to_db);
    printf("Enter your password.\n");
    gets(pwd);
    if (!strcmp(pwd, ref))
    {
        printf("Authentication successful\n");
        ok = 1;
        printf("Enter master password.\n");
        scanf("%s", master);
    }
    if (ok && !strcmp(master, "correcthorsebatterystaple"))
    {
        printf("Root privileges given to user.\n");
        give_root_privileges();
    }
    else printf("Authentication failed! Please try again.\n");
    return 0;
}
```