Operating Systems Security – Assignment 4

2017/2018
Due Date: 14 Dec 2016 (23:59 CET)

Institute for Computing and Information Sciences,
Radboud University, The Netherlands.

1 Play around with the setuid `(suid)` bit

In Week 1 lecture, you were introduced to the **setuid** bit. In this exercise, you will learn how to carry out *privilege escalation* using `suid`.

Login to your (Kali) Linux system as a **non**-root user and download the program **`showdate`** from https://www.cs.ru.nl/~vmoonsamy/teaching/ossec2016/showdate (for 64-bit OS).

Then, change the owner

```
$ sudo chown root:root showdate
```

set the suid bit and make it executable

```
$ sudo chmod u+s,a+x showdate
```

Execute the program and verify it prints the date correctly

```
$ ./showdate
Fri Nov 18 xx:xx:xx EST 2016
```

Install the tool **`strace`**

```
$ sudo apt-get install strace
```

and run it to see system calls used by **`showdate`**

```
$ strace -f ./showdate
```

Objectives

a) Find out what the program does internally. What system calls does it use?

b) Assume the role of a non-privileged attacker. Use the program `showdate` to obtain a root shell. You can verify if you succeeded by looking at the output of **`id`**, it should be something like:

```
$ /usr/bin/id
uid=0(root) gid=0(root) groups=0(root),27(sudo),1001(test1)
```

Hand in the exact console commands you used to get this working.

c) Explain what a developer could do to overcome this issue. What explicit actions should a developer take when writing software that is intended to be used with `setuid-root` to avoid these types of problems?

2 Trust models

In this exercise we consider a reference monitor which uses Mandatory Access Control (MAC) to implement the Bell-LaPadula and the Biba model. The Bell-LaPadula model uses levels **unclassified** $\leq$ **confidential** $\leq$ **secret** $\leq$ **top secret**. The Biba model uses levels **untrusted** $\leq$ **user** $\leq$ **application** $\leq$ **system** $\leq$ **trusted**. The following objects with corresponding secrecy and trust levels are used in this exercise:

- `/home/peter/database` (confidential, user),
- `/etc/password` (confidential, trusted)
- `/etc/shadow` (top secret, trusted)
- `/usr/bin/someprog` (unclassified, application)
- `/usr/lib/somelib.so` (unclassified, system)
- Network socket to 203.0.113.42, port 80 (unclassified, untrusted)

Objectives

a) For each of the following steps determine whether the reference monitor will allow the action. If not, explain why not (if there are multiple reasons, state all).
   - i User peter logs in with clearance (secret, application) and tries to run `/usr/bin/someprog`.
   - ii The process dynamically loads (reads) `/usr/lib/somelib.so`.
   - iii The process reads `/home/peter/database`.
   - iv The process writes data to the network socket.
   - v The process reads `/etc/password`.
   - vi The process writes `/etc/shadow`.

b) The process from part a) now creates a new file `/home/peter/out`. What are the permitted pairs of trust and secrecy level for this output file?