Information Security:

Theory vs. Reality

Exercise 2

due date: 2 February 2012

Tel Aviv University

0368-4474-01, Winter 2011-2012

Lecturer: Eran Tromer

Teaching Assistant: Roei Schuster (royschuster@mail.tau.ac.il)

Question 1

In class we saw that the following broken code poses a considerable security risk:

```
void my_syslog(char *inp) {
  char s[128];
  strcpy(s, inp);
  modify(s);   // an internal function that changes parts of s
  syslog(LOG_WARNING, s);
}
```

1. The code is vulnerable to a stack overflow and a format string exploit. Write correct code that fixes both vulnerabilities.
2. Stackshield is a stack overflow defense similar to Stackguard and works as follows.When a function begins executing, copies the return address located in its stack frame to a “shadow stack” (a second stack managed by Stackshield, elsewhere in memory). When the function is about to return (i.e. just before calling the opcode “ret”) the program checks that the return address on the shadow stack is equal to the return address in its stack frame, and terminates the program otherwise. Like Stackguard, Stackshield adds its checking code during compile time. Assume the shadow stack is located in some fixed location on the heap known to the attacker.
   Give sample C code and a stack buffer overflow against that code that defeats Stackguard but not Stackshield.
3. Give sample C code and a stack buffer overflow against that code that defeats Stackshield but not Stackguard.
4. How would you strengthen Stackshield to defend against your attack?

Note: The only vulnerability in your code in questions 2,3 should be a stack overflow vulnerability.

Question 2

1. When using system call interposition explain, how the reference monitor must behave when the monitored program calls

   ```
   system("./helper_program");
   ```

   where *helper_program* is a helper program in the same directory as the monitored program.
2. In our discussion of virtual machine introspection we mentioned the “lie detector” test as a way to test for kernel-level malware. Suppose the malware author knew exactly how the lie detector test was implemented. Explain how he might design his malware to fool the test.
   Assume that the VMM Security assumption holds for the VMM used to implement introspection, i.e., malware cannot break out of its VM.
3. Explain how the author of the virtual machine introspection engine can strengthen the lie detector test to defeat your attack above.

Question 3

Consider a Bell & LaPadula style multilevel security setting in which all data is labeled with one of four security levels: Public (P), Quarantined (Q), Restricted (R), and Secret (S). These are ordered as follows: P< R, P< Q, Q< S, and R< S. In other words, Public is a lower classification than Quarantined or Restricted, and both of these are lower than Secret. However, neither Quarantined or Restricted is lower than the other, so information is not allowed to flow from Q to R or from R to Q.
We can apply this confidentiality model to code by labeling each variable with one of the four levels. A labeling is *correct* for a program if each labeled variable only depends on variables of equal or lower classifications. For example, if a has label P and b has label R, then an assignment b=a is correct but a=b is not because the second assignment allows restricted information to be stored in a public variable.
Consider the following program that calculates output variables v, w, x, y, and z from input variables p, q, r, and s. Assume that the input variables have security labels that correspond to their names (i.e. variable p has label P, etc.). Assume that all output variables are initially set to 0.

```
v = p + q;
if (r > 0) then {
   w = 1;
   if (v > 0) then {
      x = 1;
   }
} else {
   y = 1;
}
z = s - s;
```

For each output variable v, w, x, y, and z, state the minimal security label it can be given so that system is secure according to the information-flow policy. Each answer should be one of {P, Q, R, S}.
Explain your answer. Explain your choice of z's security label.

Question 4

Paranoid Bob bought a laptop containing a special tamper-resistant[1] motherboard, which includes (among other things) a TPM, processor and BIOS. Bob encrypts the data stored on his laptop's hard disk. To protect the disk, he seals the key with a TPM. Whenever an OS update occurs, the correct PCR values change, so Bob seals the key again and saves the blob in a new file (outside the encrypted partition, of course). Bob first installed his operating system 5 years ago, and has been diligently installing all security updates ever since (which is very fortunate, since one of these updates fixed a nasty buffer-overflow bug in the system login procedure). Bob had been examining his system carefully over the years[2], and has noticed that some PCRs have always been updated (extended) in the boot process with the same values. These are the PCRs extended by the BIOS code.

1. What can you learn about the updates to Bob's system? Where (physically) does the (binary) code influenced by these updates reside?
2. You found Bob's laptop, powered off. Describe a method for decrypting its content.
3. Suggest a (possibly new) TPM feature that Bob can use to prevent such an attack.

Question 5

1. ActiveX is a technology for packaging software components for embedding in larger applications. An ActiveX "control" is essentially a DLL file, with a special interface, that can be loaded into a host application such as a web browser, and can provide its own UI and functionality within that applications. To determine which ActiveX controls may be loaded into applications, the controls are signed using cryptographic certificates, similarly to the X.509 certificates used by SSL/TLS. An ActiveX program can be signed by a private key of the developer. A certificate authority can certify (also by digital signature, as we have learned in class) the ownership of the private key by the developer. The web browser contains a list of certificate authorities it trusts.
   BunnY is an (imaginary) ActiveX control which offers a "game engine" that plays games written in a proprietary programming language. Among other advantages, BunnY programming offers extensive libraries enabling quick and easy development of web games in which there are many graphical figures of bunnies.
   BunnY invokes some dangerous system calls (e.g. it needs to access the the printer in order to print screenshots upon users' request). Nevertheless, games developed in the BunnY programming language are sandboxed so they can't do much harm. Several sites offer BunnY games developers to upload their games. BunnY was developed by BuggS, a well-known and trustworthy vendor.
   Elmer, who is browsing the web, tries to play a BunnY-based web game. His web browser informs him that the game requires installing BunnY, which is signed by a private key certified as belonging

[1] Assume tampering with the motherboard, components inside or the interfaces between them it is not cost-effective.
[2] Assume Bob can always fetch and examine the current PCR values.

to BuggS Corporation.
What could go wrong, if Elmer agrees? (Assume BuggS is indeed well-known and trustworthy.)

2. 0xMallory is an independent security analyst from Mexico, who found a buffer overflow vulnerability in BunnY that can be used to break out of the sandbox and hijack control over the plugin code. She would like to grow his collection of other people's credit card numbers. What can she do?
3. After a credit card theft scandal, BuggS found a sample of 0xMallory's deeds, reversed engineered it, and deduced the way BunnY was misused. BuggS then quickly fixed the vulnerability in BunnY, signed the new version of the ActiveX control, and distributed it to the web sites that used BunnY. 0xMallory, deprived of her card-collection hobby, decided to cultivate his political aspirations. She has found out that the Mexican prime minister is very fond of bunnies, and will surely love being told about web-based bunny games. 0xMallory knows the prime minister's private email address. How can 0xMallory read all of his prime minister's email correspondence?

Question 6 – Ex1 Anti-Exercise

Here you have to describe attacks example answers[3] to sections from question 1 in the exercise. Describe at least 2 attacks for each answer (overall at least 6 different attacks).

Section 1:

a. SafeRing will not be able to access the internet, but the helper application will. The feedback format is assumed to be a mapping from question numbers to numbers between 1 and 7. An example question is "On a scale of 1 to 7, how accessible is the UI?".
SafeRing lets the user fill a form and sends the mapping to the helper application. The helper application is assumed trustworthy, and declassifies the data verifying conformity with the predefined format (a N→[0,..,7] mapping). If it conforms with the format, the filled form is sent to SafeRing's developer.

Section 3:

a. We assume there is a predefined and immutable set of rules which can be invoked by the application, and invoking a rule simply means a specific e-mail is sent to a specific address. Rules are numbered. We define an "invoke rule" permission. A helper application will expose an IPC interface allowing any application with "invoke rule" permission to send a message containing a rule number. The helper application will then invoke this rule. The helper application will not contain MalloryAds.
SmartMail will have no internet access (but just access to the inbox), but will be permitted to send the helper application requests to invoke rules.

Section 4:

a. Define the messages sent from SmartMailBack to SmartMailFront in order for it to keep the log, as "classified" objects. An application can choose to irreversibly become "classified", gaining the ability to read "classified" objects, but losing the privilege of internet access. Messages sent from any "classified" application are also "classified" objects. SmartMailFront becomes classified on its startup. SmartMailBack does not become "classified" at all. It knows the sensitive rules, and still can access the web. But this is OK – because it is assumed to be trustworthy as it does not contain MalloryAds. EmailLogger will have to become "classified" and lose internet access in order to receive messages from SmartMailFront.

[3] Based on (good, but not necessarily completely correct) students' answers, all of them submitted more than once.