CS 161 Fall 2006 — Computer Security Joseph/Tygar — HW 3

Due Friday, December 1 at 3pm

Please include the following at the top of the first page of your homework solution:

Your full name
Your login name
The name of the homework assignment (e.g. hw3)
Your TA's name and section number/time
Names of students you worked with

Please explain all work clearly. We will be enforcing length limits strictly on this homework, taking off points for answers that exceed them (including run-on sentences). Please write concisely.

You will submit code for Question 3 electronically. You may choose whether to submit the writeup electronically or in hardcopy. If you submit the writeup electronically, please use plain text, HTML, or PDF. See Question 3 for submission details. If you submit the writeup in hardcopy, staple all pages together and put them in the CS 161/Fall 2006 slot of drop box #2 in 283 Soda as usual.

Both electronic and paper submissions must be turned in by 3pm on Friday, December 1. *No credit will be given after 3pm on the due date. If you have not finished, turn in what you have for partial credit.*

Homework exercises:

1. **(25 pts.) Digital Millennium Copyright Act**

   Read a summary of Digital Millennium Copyright Act here:

   `http://www.copyright.gov/legislation/dmca.pdf`

   Based on the DMCA summary, give a **brief** (1-2 sentences) argument to support or refute each of the following statements:

   (a) It is illegal to distribute software that disables the advertising code from an executable that uses advertising to generate revenue (for example, the free "sponsored" version of Eudora at www.eudora.com).

   (b) It is illegal to use ad-blocking web browser software, such as pop-up blockers or adblock (addons.mozilla.org/firefox/10/).

   (c) It is illegal to build a digital video recorder that automatically detects broadcast commercials and does not play them.

   (d) It is illegal to build a VCR with a fast forward button, because a viewer can use it to skip over commercials broadcast on television.

   (e) It is illegal to go to the bathroom if you are watching television while commercials are being displayed.

Here is an example of the sort of brief yet descriptive answer we are looking for:

(x) It is illegal to distribute software that automatically removes digital rights management protection from multimedia files.

This is true; it clearly is prohibited by the "anti-circumvention" rules of the DCMA.

2. (25 pts.) Worm Propagation

In lecture, we talked about ways of increasing the propagation rate of worms. In this problem, we'll examine the effects of decreasing the propagation rate of worms.

Recall that $a(t)$ is the proportion of machines in a network that are infected by a worm at time t, K is the initial compromise rate, and T is a constant of integration that fixes the time position of the incident. We'll use a Random Constant Spread (a.k.a Susceptible-Infected) model for worm propagation and assume a network of tens of millions of susceptible machines.

Please limit each answer to 1-3 sentences. You may also include graphs or tables if you like, though they are not necessary.

(a) If K is 1.8 and T is 12, at what time are 50 percent of the machines infected? At what time will all machines be infected? Hint: An easy way to work through this problem is to use Mathematica, Excel, or OpenOffice to generate or graph your results.

(b) A value of K of 1.8 means that initially each infected machine is able to find and infect 1.8 other machines. Subsequently, each machine is able to find and infect $K * (1 - a(t))$ machines. If we are able to reduce the initial infection rate to 0.9, what is the 50 percent infection time? What is the time for all machines to be infected?

(c) For the two infection rates, examine the infection proportions before and after the 50 percent points. What do you observe about the differences?

3. (50 pts.) Buffer overflow exploit

For this problem, you will write an exploit for a buffer overflow vulnerability. To get started, read over Aleph One's "Smashing the Stack for Fun and Profit".[1] (You don't have to read the section "Shell Code", since we provide you with shellcode, though you may find it interesting.)

The vulnerable program is `/home/ff/cs161/hw3/victim/victim` on the instructional machines. Copy the directory `/home/ff/cs161/hw3/exploit` to your working space; it contains skeleton code and a Makefile for your exploit program.

Your task is to edit `exploit.c` so that it exploits the buffer overflow vulnerability in `victim` to run a shell. We provide exploit code in `shellcode.h`; you just have to cause it to be executed in `victim`. If you are successful, you should see a '$' shell prompt:

```
bash-3.1$ ./exploit
$
```

The only file you should edit is `exploit.c`. Build it with `gmake`. The path to `victim` is hard-coded in `exploit.c`; please do not change it.

Because buffer overflow exploits are highly machine-dependent, you are restricted to working on **sphere.cs**, **rhombus.cs**, or **pentagon.cs**. Your exploit must work on one of those machines (they are Solaris x86 boxes).

[1] `http://reactor-core.org/stack-smashing.html`

To start with, we recommend that you use `gdb` to explore the stack and memory layout of `victim`. It will be different when called via `execve()`, so here is the best way to get set up (after running `gdb exploit`):

```
(gdb) run
Starting program: /home/ff/cs161/hw3/exploit/exploit

Program received signal SIGTRAP, Trace/breakpoint trap.
0xce7cd062 in ?? ()
(gdb) symbol-file /home/ff/cs161/hw3/victim/victim
Load new symbol table from "/home/ff/cs161/hw3/victim/victim"? (y or n) y
Reading symbols from /home/ff/cs161/hw3/victim/victim...done.
warning: rw_common (): unable to read at addr 0xce7ac660
warning: sol_thread_new_objfile: td_ta_new: Debugger service failed
(gdb) break main
Breakpoint 1 at 0x80507d7: file victim.c, line 12.
(gdb) continue
Continuing.

Breakpoint 1, main (argc=2, argv=0x8047f34) at victim.c:12
12          if (argc != 2) {
(gdb)
```

Running it this way makes it difficult to restart, however, so you may want to just run `gdb victim` to explore initially and then switch to the `execve()` version when it's time to find the actual addresses for your exploit.

You will want to become familiar with the following `gdb` commands (use the 'help' command): `break`, `where`, `disassemble`, `next`, `nexti`, `x`, and `info`. Be sure to explore the display options for the `x` command.

You should not follow Aleph One's directions too closely. You may or may not want to execute the shellcode on the stack, and you can use `gdb` to figure out the exact address to jump to, so you don't have to use anything like `get_sp()` or NOP padding.

You must submit your code electronically. Go to the directory where `exploit.c` resides and type `submit hw3`. The files that must be present are `shellcode.h` (unchanged), `Makefile` (probably unchanged), and `exploit.c` (changed). If you are submitting your writeup electronically as well, that should be here as `hw3.{txt|html|pdf}`. Be sure you say yes when asked if you want to include that file.

You must ensure that your code runs on one of the three servers listed above. We should be able to type `gmake` and then `./exploit` to run your exploit.

You should include in your homework writeup a brief description of how you tackled this problem, including how you determined which address to jump to. Please tell us which server you ran your code on (sphere, rhombus, or pentagon). The writeup for this question should be no more than 6 sentences.

Do not forget to list students you worked with for this homework. As always, you may discuss problems with other students but you may not share writing (including code).