

Shellshock Attack

Copyright © 2017 Wenliang Du, All rights reserved.

All problems are 2 points unless otherwise noted.

- 3.1. When a shell variable containing a shell function definition is passed down to a child process as an environment variable, what is going to happen to the function definition?
- 3.2. For the Shellshock vulnerability to be exploitable, two conditions need to be satisfied. What are these two conditions?
- 3.3. How do user inputs get into a remote a CGI program (written in Bash) in the form of environment variables?
- 3.4. There is another way to send inputs to a CGI program. That is to attach the input in the URL. See the following example.

```
http://www.example.com/myprog.cgi?name=value
```

Can we put our malicious function definition in the value field of the above URL, so when this value gets into the CGI program `myprog.cgi`, the Shellshock vulnerability can be exploited?

- 3.5. We run "`nc -l 7070`" on Machine 1 (IP address is 10.0.2.6), and we then type the following command on Machine 2. Describe what is going to happen?

```
$ /bin/cat < /dev/tcp/10.0.2.6/7070 >&0
```

- 3.6. Please describe how you would do the following: run the `/bin/cat` program on Machine 1; the program takes its input from Machine 2, and print out its output to Machine 3.
- 3.7. (6 points) Consider the following program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

extern char **environ;

int main()
{
    char *args[] =
    {
        "/bin/sh", "-c",
        "/bin/ls", NULL
    };
    pid_t pid = fork();

    if(pid == 0) {
        /* child */
```

```
    printf("child\n");
    execve(args[0], &args[0], NULL); ①
}
else if(pid > 0) {
    /* parent */
    printf("parent\n");
}
return 0;
}
```

The program forks a child process, and executes the `/bin/ls` program using `/bin/sh`, which is a symbolic link to `/bin/bash`. The program is executed as the following. Explain what the output of the program will be and why.

```
$ gcc prog.c -o prog
$ export foo='() { echo hello; }; echo world;'
$ ./prog
```

- 3.8. Let's make a change to the code in Problem 3.7.. We change the code in Line ① to the following. Please redo Problem 3.7. with this change made.

```
execve(args[0], &args[0], environ);
```