Project: Part 2

Goal (from part 1):

The goal of this project is to give you some experience in analyzing the security of a system. For this, you will play the role of a security analyst.

You work for a corporation that keeps its data on a system protected by a firewall. The system runs a web server that displays sanitized data. It should not display any sensitive data. Your bosses have asked you to determine the following two things:

1. Does the firewall correctly pass the traffic that it is supposed to pass, and block all other traffic?
2. Can outsiders (those without accounts on the system) get access to any data beyond that displayed on pages marked *authorized*?

The following outlines what you are to do to answer these questions

Third Step: WWW

We'll focus on the web server now. The first step in the analysis (the Flaw Hypothesis Methodology) is to learn about security problems in web servers, and in particular holes found in this web server.

Identify the web server

First, figure out the type of the web server. Do this by a telnet to port 80 on the target host, 169.237.7.61 (see the *telnet*(1) manual page) and note the greeting. Then hit return.

Due: Please turn in the output from this step.

Learn about vulnerabilities in web servers

This step asks you to gain insight into problems that web servers have had in the past (and undoubtedly still have).

First, look for vulnerability reports on web servers. A good way to do this is to go to the site *www.securityfocus.com*[1] Please go to this web site, and click on the Vulnerabilities entry in the left-hand menu. In the center, you will see a menu with 5 buttons, plus data below the row of buttons. This is the access point to their vulnerability database. Click on "by keyword." Now search for a string like "web server" to obtain a list of web server vulnerability reports.

Now click on the first title (when I did this it was "IBM HTTP Server Denial of Service Vulnerability"). Look at the two lines: "class" and "cve". These give 2 classifications of the vulnerability.[2] For more details, click on the "discussion" button. (You can look at the "exploit" part, too; sometimes that is very helpful.)

Another good search would be to look at CGI script holes. These programs are run by web servers, and should constrain what the remote browser (user) can do. Unfortunately, they often don't work too well.

Finally, see if you can find any reports of vulnerabilities in the specific type of server that the target is running.

Due. Please submit the string(s) you used to search the database. Then look at no fewer than 10 reports, and submit their names and the two classifications of each. What seems to be the most common problem, or set of problems? Finally, please submit the search string you use to look for vulnerability reports on the particular server, and the titles and classifications of any vulnerability reports. Do they apply to the version of the server being run?

Trying to get in

Based upon your results, try to gain access to the target. You can do this in a couple of ways:

---

1. I recommend you either allow or disallow cookies before you go there (don't have the browser ask if it should accept individual cookies). The cookies are harmless, but you get so many of them that being asked if you want to accept each one becomes very annoying.
2. "Class" refers to SecurityFocus' own classification scheme. "Cve" refers to the common vulnerabilities exposure classification; for more information, see *http://cve.mitre.org*.

1. Attack the server directly. Try to exploit the types of vulnerabilities you uncovered in the previous part.
2. Attack the CGI scripts provided for the server. (A list of these will be posted to the newsgroup. It is not guaranteed to be complete!)

Both parts have common elements. Buffer overflow attacks, strings with unusual characters as arguments, and so forth may produce surprises. So, you might proceed by looking back to the notes from the beginning of the term, in which common programming errors were discusses. Think about how that discussion correlates with the data you found in the previous step. This should give you a good idea of where to begin.

Due. Please submit the following:

1. Which of the 5 types of programming errors (see #3 on the notes for October 10, 2000) was most common in the set of reports you found? Where does this suggest you should start your probing?
2. Develop and run 3 tests on the web server or CGI scripts on the web server. For each test, say:
   a. what the test tries to do;
   b. how you are doing it;
   c. what you expect the result to be; and
   d. what the results are.

   You need *not* make the tests attacks. However, they should be designed to tell you something about the server's (or CGI script's) robustness and/or ability to confine the user to the web page.
3. Write a one-paragraph assessment of the security of the web server. In particular, do you think it prevents outsiders from gaining unauthorized access to the system? Justify your opinion in light of the results of the first two parts of this "due" section.

Turning In Results

Please submit them to the directory *proj2* using the *handin* program. This is due on December 8 at 11:59PM.