

CS 487, Programming Assignment P2: Web Application Vulnerabilities

V.N. Venkatakrishnan

Introduction

This assignment has several objectives. First, it gives you an idea of how simple web applications are programmed. Secondly, it requires you to understand the structure of given web application. Third, it requires you to perform a security analysis of an application. Finally, it requires you to *retrofit* these applications to defend against all of these attacks.

Logistics

You will need to work individually for this assignment. Submission instructions are described in a later section. Any clarifications and revisions to the assignment will be posted on the course Web page.

The material needed for this homework is available on the Linux machine `thompson.sisl.rites.uic.edu`. Everything is installed on your account in this machine.

Start early!!

Start by changing to a (protected) directory in which you plan to do your work. (Note: It is your responsibility to protect your work; Since you are doing a practical course in security, you are expected to keep your work confidential. If you do not know how to set up permissions under Unix, please read Chap 5 of UNIX and Internet Security by Garfinkel et al.)

Description

You will be assigned a web application that has several security vulnerabilities.

The application is one of the following:

- *App 0: Bookstore* An online bookstore application.
- *App 1: Classifieds* A web application that allows posting classified advertisements.

- *App 2: Employee Directory.* A web listing of employees of an organization.
- *App 3: Events* An event scheduler.
- *App 4: Portal* An online portal of articles.

To look at the deployed version of these applications, visit <http://thompson.sisl.rites.uic.edu:443/35c74dc9> and click on the links.

There will be two parts to this homework. A written submission and changes to the sources. One difference is that you will not submit the source code, but will leave it in your account for us to access and test your submission on the same server.

Assignment of applications

You will be assigned an application among these five applications. Just log into your account on thompson.sisl.rites.uic.edu with the same username and password as the first homework. Look for a symbolic link named `www` in your home directory that will take you to the source code for the application.

These web applications are written in JSP (Java Server Pages). Applications in JSP are written in Java. However, they need not be compiled. Any changes you make to these applications are made available over the web right away. If you want some introduction to JSP, read the Wikipedia page on JSP. This is however, not required though, as you can easily follow the application logic by looking at the source code and also by examining the application through a browser.

To access your assignment over the web, use the following URL: <http://thompson.sisl.rites.uic.edu:443/COOKIE/APPLICATION/Default.jsp> where `COOKIE` is your cookie from HW1 and `APPLICATION` is one of `bookstore`, `classifieds`, `portal`, `empldir` or `events`.

The programs

The programs I have given as part of this homework are real world applications. These applications, though have some useful functionality, have not been hardened against any security threats. In addition, we have also introduced some weaknesses into them.

Now we describe the four parts to the homework. Each part has a written component, and make sure each part is answered separately.

Part I: Application Study (10 points)

Study the application carefully. Can you describe the design of the application in about 10 sentences? For this part, focus on architectural design rather on code level description.

Part II: Security Analysis (30 points)

From your knowledge of the application study above, can you identify two vulnerabilities in the system? Find an instance of a SQL injection vulnerability and one instance in which client-state is not validated.

Describe each vulnerability briefly (about 3 sentences). Your response should address the following questions:

- Under what conditions will the vulnerability be exploited?

Part III: Exploit Construction (30 points)

For this part, you will construct two *exploits* for each of the vulnerabilities you identified in the previous step. Recall that a vulnerability is a mere weakness, but an exploit is a concrete demonstration of a weakness.

Your exploits must be constructed in your submission in such a way that would be easy for us to *replicate* the attack.

The easiest way of providing the exploit is to provide a wget string in (with the appropriate options) that will directly launch the attack.

Your exploits must be repeatable, and must be described very clearly. In your written submission, explain the mechanism to launch these exploits clearly.

Ensure that the URLs you provide work and are escaped correctly.

Part IV: Retrofitting to defend attacks (30 points)

Address each one of these exploits that you found through suitable means. Remember that you do not have to fix **all** the vulnerabilities that you found in the application, but only those you described in Part II above. Specifically, your defense should address the exploits described in part III above, and any similar exploits. Describe the techniques you used to retrofit the applications in Part IV of your written submission.

Create a subdirectory under your original URL containing all the retrofitted files that demonstrate that your fixes work.

Some other (logistical) hints

1. You can implement any technique that you think will defend against attacks. For instance, you can have stronger input validation. Note that your technique should defend any similar attacks at the vulnerable location.
2. You should test any technique you implement with several exploits before being convinced about its strength.
3. Although all your application code copies are different, they all share the same underlying database. So do not use “dangerous” exploit strings (such as “SHUTDOWN;” or “DELETE * from” or “DROP

ALL”) on the database. Also, if you need to add any entries to the database, prefix them with a unique string (such as your name or cookie, so that someone else’s exploit will not delete them).

You can discuss the design of web applications, but not any specific vulnerabilities or exploits.

Submission Instructions

Submission

Your submission must only contain the wget files and the essay.

- `client-state-attack.wget` the wget file for launching the corresponding exploit.
- `client-state-defense.wget` the wget file for demonstrating that the attack does not work.
- `sql-inject.wget`
- `sql-inject-defense.wget`
- `Readme.txt` The essay describing your submission. You can also submit a PDF file for your essay, in that case, name your file `Readme.PDF`.

To submit, create a tarball of all the above files. Name the file `net-id-hw2.tar`. and email it to the course account (i487 at cs dot uic dot edu). Do NOT email them to me.

The following is a sample session :

```
$ mkdir submit
$ cp client-state-attack.wget client-state-defense.wget .... submit/
$ tar cvf venkat-hw2.tar submit
```

We will test your program on using your own exploits (and any of our similar exploits).

If you want to use rich text, only use PDF attachments. Do NOT submit MS-Word documents. Again, please do not email your essays to me, I do not keep track of these submissions.

Other points

You are allowed to discuss the homework, but not any solutions. Any cheating will be treated with the maximum possible penalty. If you help anyone in coding the homework, you will be considered guilty. In your essay, you should name all the students you discussed the homework with and any references that you consulted. Failure to list the above information completely is also considered cheating. It is “OK” to submit an unfinished homework, than to take these illegal short cuts to obtain some points in the homework.