Assignment

Your first assignment is to

1. Identify and understand a vulnerability in an open source software project[1]
2. Write a brief report describing the vulnerability as if you found it during a code review

How to Identify a Vulnerability

There are two distinct ways to identify a vulnerability in the context of this homework.

1. Use a vulnerability database such as OSVDB[2], Secunia, CVE, NVD, or full-disclosure to identify an existing vulnerability.
   OR
2. Download and audit the source code of an application to find a new and unpublished vulnerability.

If you chose the first option, I would make sure you choose a software project that is written in a language that you're familiar with. Include the OSVDB (or other) ID# you used in your report.

Students who chose the second option and find a new and unpublished vulnerability will get double credit for this assignment. If you chose this option, I would suggest you chose a software project written in a language you are intimately familiar with as well as a project of small size and low complexity.

How to Understand a Vulnerability Class

I will expect you to rapidly become familiar with new and different classes of software vulnerabilities in this course. A deep understanding of a software vulnerability class will help you get the issue fixed and will also help you exploit it. Two resources I use to help me understand software vulnerabilities are the Common Weakness Enumeration (CWE) and the Fortify VulnCat[3].

How to Perform Code Review

Use the strategies that Vinnie Liu went over in his presentation, obviously focusing on his advice for performing manual source code analysis. If you looked up a vulnerability from a vulnerability database, look for reference URLs which may contain additional information such as original advisory e-mails, bug reports, or proof of concept exploits. These references may help you track down the exact component or function that is affected.

Note: You must find the source code to an affected version of the application to complete this assignment.

How to Write Your Report

[1] This can include any code that you wrote, past assignments, code from a friend, etc.
[2] http://letmegooglethatforyou.com/?q=OSVDB
[3] http://www.fortify.com/vulncat/

The audience for your report is a client that hired you to perform code review. Therefore, you must balance your report to be short enough for a manager to read yet technically detailed enough to aid a developer in fixing the security issue. A brief description of the vulnerability, a code snippet that demonstrates it, and a trace of user-controllable input required to reach it are items that will help you explain the issue. Screenshots, though not necessary for this assignment, are extremely useful for such reports in real life. An example from a report I authored is reproduced below:

> Areas within the APPLICATION employ string building to dynamically construct SQL queries and commands. The exploitable code instances include user-controllable application input parameters within the SQL query or commands that are not properly validated to verify dangerous characters are rejected, escaped, or removed before executing the query or command.
>
> In iBatis, elements bounded by $'s (for example, $orderBy$) become "simple dynamic SQL elements" whose values are replaced by properties in parameter objects. If the values of such properties in parameter objects are controllable by the user, this functionality can be abused to pass arbitrary SQL to the database, ie. a SQL injection vulnerability.
>
> The following code excerpt was taken from FILE in APPLICATION:

```
<isNotEmpty prepend="order by" property="orderBy">
  $orderBy$
</isNotEmpty>
```

Listing 1: Dynamic SQL String Building in FILE

> In the example above, the METHOD calls METHOD upon handling a request. The use of SQL string building results in an exploitable condition when application users manipulate parameters passed to that controller. The above vulnerable flow can be reached through the following actions by an ACCOUNT on APPLICATION:

1. Signup to trial ACCOUNT on APPLICATION
2. Verify email address and login to account
3. Browse to ...
4. Click ... and create a new ...
5. On the PAGE page, click ... to expose the ... parameter

> Impact: The use of string building for SQL queries and commands can lead to SQL Injection vulnerabilities when user-controllable input that is not properly validated is used to construct the string. This vulnerability results in access to back-end application databases with all privileges granted to that of the application database login account. This could include the ability to read and write to most application-related tables and to potentially export sensitive user data such as customer login IDs, customer contact data, etc. Using this vulnerability, business logic normally enforced by the application could potentially be circumvented since the attacker has direct access to back-end database tables.

Also note, you are required to calculate a CVSS score[4] for the vulnerability and include it in the report. Please submit your reports as PDFs in the E-Poly dropbox.

Assignment objectives

- Learn to use vulnerability databases
- Research software vulnerability classes
- Perform source code review
- Practice technical communication

This assignment is graded out of 10 points.

[4] http://nvd.nist.gov/cvss.cfm