Codename Anders

22 April 2013

Background

This assignment is about making a security assessment of a real open source project, and trying to remedy some security issues.

An unnamed Faculty of SCIENCE is contemplating to replace a certain IT system used for teaching because there have been found numerous security issues in the current IT system. The successor system goes under the codename *Anders*. An unnamed Vice Dean is championing an open source system called jurpopage as starting point for Anders.

Your task is to make a security assessment of jurpopage and write up your findings.

What to Hand In

You should hand-in two things:

- A report (as a text file or PDF file) with findings.

  Your report should be no longer than four pages (excluding appendices and code examples) and should be composed for ease of grading. That is, structure your report so that the first four pages makes it clear why you should pass the assignment, and include minor finding in appendices.

- Source code you may have produced as part of your hand-in, in a zip archive.

Appendix A contains instructions on how to set up an evaluation environment. This step is only relevant if you are *not* using the PCS VM image (as jurpopage is already set up and running there).

Appendix B contains an example of how we would like you to structure your hand-in.

Task 1: Black Box Assessment

Even though you have access to the source code of jurpopage, you should (try to) complete this part of the assignment as a *Black Box* assessment. That is, without looking at the source code for jurpopage. And you should only access jurpopage through the web interface.

You shall focus on Access Control and Cross-Site Scripting (XSS). If you have trouble getting started, you may start with the challenge to determine if it possible to gain assess as the special user `admin`.

To get access as `admin` you might try:

- Brute force
- Cookie theft

Other things you might want to investigate:

- Phishing with XSS
- Stored XSS Attacks
- Cross Site Request Forgery (CSRF)
- Reflected XSS Attacks
- HTTPOnly Test
- Cross Site Tracing (XST) Attacks

Task 2: White Box Assessment

In this task you are allowed (but not required) to look at the source code for jurpopage. If you solve the task as a black box assessment, let us know and we'll give you extra credit.

You shall focus on SQL Injection. Your main tasks are to:

- See if you can get a list over registered users and their information (in particular email and password).
- See if you can register a user and give that user the same rights as the `admin` user.
- See if you change the password for `admin`.

Task 3: Fix It

Select an XSS issue and an SQL injection issue you have found, and fix them. That is, find (some of) the relevant places were HTML or SQL can be inputted by a user, and implement sanitizers for the input.

Demonstrate that your have fixed the issues by writing tests that demonstrates that the issues have been fixed. Your tests should be more than just string comparisons of bad input to good output, as it should not depend on specific implementation of the sanitizers. You can assume that non-dangerous HTML is left alone and that dangerous HTML is removed or replaced, but should not depend on the specific way that the transformation is performed.

Appendix A: Installing Evaluation Environment

Follow this guide *only* if you are not using the PCS VM image. If you *are* using the PCS VM jupopage is already running at http://localhost (inside the VM).

Rather than installing jurpo on you own machine, we (strongly) recommend that you install it in a virtual machine. We have good experience with VMware (VMware Player) and VirtualBox.

VirtualBox Notes

- VirtualBox reads VMware images.
- VirtualBox is available in most Linux repositories

  ```
  sudo apt-get install virtualbox-ose
  ```

  The current `virtualbox-4.0` package (at least) in Ubunt 10.10 might have some issues on computers that doesn't have hardware virtualization when the guest OS is compiled with PAE, which the TurnKey LAMP image is. The `virtualbox-ose` package has been tested and works on Ubuntu 10.10
- When the TurnKey LAMP image is imported, remember to enable PAE. This can be done in the settings dialogue of the imported image under System -> Processor -> check "Enable PAE/NX"

Turnkey Linux: LAMP

As jurpo is based on PHP and MySQL, the TurnKey LAMP Stack Appliance (VMware image, OVF image) is an excellent starting point. Download it, boot it, set the root passwords, and update it.

Now, to get your PHP installation jurpo-ready you should:

1. Install GD support

   ```
   root@lamp ~# apt-get install php5-gd
   ```

2. Edit `/etc/php5/apache2/php.ini`, find `register_long_arrays` on line 701 and change it to `On`

   ```
   register_long_arrays = On
   ```

IMPORTANT: This is required for jurpopage to function properly.

Installing Jurpo

1. Get jurpo from http://diku.dk/~kflarsen/jurpopage.zip and unzip it.

   ```
   root@lamp ~# wget http://diku.dk/~kflarsen/jurpopage.zip
   root@lamp ~# unzip jurpopage.zip
   root@lamp ~# rm jurpopage.zip
   ```

2. Create an MySQL database for jurpo

   ```
   root@lamp ~# mysql -h localhost -u root -p
   mysql> CREATE DATABASE jurpopage;
   mysql> quit;
   ```

3. Populate the jurpo database

   ```
   root@lamp ~# mysql -h localhost -u root -p jurpopage
   mysql> source jurpopage/jurpopage-datatest.sql
   mysql> quit
   ```

4. Edit the file `jurpopage/www/library/configuration.php` to make the following changes:

```
$website_url = "http://localhost/jurpo/"; /* Your Website URL */
$db_passwd = "your mysql root password"; /* change it to your database password */
```

5. Get jurpo in place:

```
root@lamp ~# mv jurpopage/www /var/www/jurpo
root@lamp ~# rm -rf jurpopage/
root@lamp ~# cd /var/www/jurpo
root@lamp ~# find . -type f -exec chmod 644 {} \;
root@lamp ~# find . -type d -exec chmod 755 {} \;
root@lamp ~# chmod 755 .
```

6. Restart apache:

```
root@lamp ~# /etc/init.d/apache2 restart
```

7. Enjoy jurpo at `http://<virt.machine.ip.addr>/jurpo`

Appendix B: Example Structure for hand-in

Following is an example solution (unfortunately it is rather misguided) for how we would like you to structure your hand-in.

```
# Problem: Unprotected Attributes

The attributes in the class ‘pagination‘ have been declared with the
keyword ‘var‘ which implicitly gives public visibility in PHP.  This
is a problem, as it allows unauthorised client code to modify the
attributes (as I understand the documentation on php.net), which in
turn can lead to security problems.

# Test/Example that demonstrate Unprotected Attributes Problem

Below is a client class that modify the variable ‘$q‘:

‘‘‘
    class RockStar {
      function roll() {
        $p = new pagenation();
```

```
        $p->q = "La, la, I can change all your variables";
      }
    }
```

Fixing the Unprotected Attributes Problem

Replace ‘var‘ with ‘public‘.