

# COMS 252 HOMEWORK 10: WEB SERVER

## 1 Objectives

For this assignment, you will configure network settings and build a web server, in Linux. For more information, consult

- Chapter 17 of the textbook.
- Chapter 26 of the textbook.
- The Apache manual, which should be available at <http://httpd.apache.org> (you will be running Apache version 2.4).
- The Apache version 2.4 tutorials, which should be available at <http://httpd.apache.org/docs/2.4/howto/>.
- A text file named

`/usr/share/doc/iptables-*/sysconfig.txt`

where “\*” is replaced by a version number. This is a large file, but there is a section that discusses `/etc/sysconfig/network-scripts`, in particular the files in this directory and the contents of the files.

## 2 Download

Download the virtual machine `Hw10.ova`. Accounts are `root` and `user`, with passwords `rootpw` and `userpw`, as usual. You should not need to install any packages for this assignment.

Currently, the virtual machine has one virtual network adapter enabled, with type “NAT”. This is used to connect to the Internet, and is required for submitting your work. Do not change this. You will add and configure a second network adapter for a virtual network between your host machine (i.e., whatever machine you are using to run VirtualBox) and the virtual machine. This way, you can test the web server on the VM by running a browser on your *host* machine<sup>1</sup>.

## 3 Configuring the VM

### 3.1 Build a virtual network

In the VirtualBox menu, select “File” and then “Host Network Manager”. Add a “host-only network” here (or use one that already exists), and make a note of its name. This is a virtual private network that VMs and your host machine can connect to. Next, you need to configure the network. Make a note of the IPv4 address and network mask. The IPv4 address will be the address of your *host* machine on this network; this will be referred to later as `www.xx.yy.h`. Also, make sure the “DHCP server” is *disabled* for this network. You will set up the VM to use a static IP address on this network.

### 3.2 Connect the VM to the network

Go to the settings for the VM for this assignment, and under “Network”, enable “Network Adapter 2”. The network type should be “Host-only adapter”, and be sure that the network name matches the one you just set up. Make a note of the MAC addresses for both network adapters.

---

<sup>1</sup>How cool is this?

## 4 Network configuration in Linux

The following will require you to use a text editor to edit the network interface configuration files in directory `/etc/sysconfig/network-scripts`.

### 4.1 Network adapter 1

The first network adapter should already be configured, on interface `enp0s3`. However, it was configured for a machine with only one network interface. Edit the configuration file for interface `enp0s3`, and specify the MAC address for the adapter (this is necessary for machines with more than one network interface), by adding a line of the form

```
HWADDR=xx:xx:xx:xx:xx:xx
```

where “`xx:xx:xx:xx:xx:xx`” is replaced by the MAC address for Network Adapter 1.

### 4.2 Network adapter 2

The second network adapter is interface `enp0s8`. Create a configuration file for this interface, with the following entries.

```
DEVICE="enp0s8"
ONBOOT="yes"
BOOTPROTO="none"
GATEWAY=0.0.0.0
IPADDR=www.xx.yy.z
NETMASK=nnn.nnn.nnn.nnn
HWADDR=yy:yy:yy:yy:yy:yy
```

Note that `www.xx.yy.h` is the *host machine* IP address on the host-only network, and you must use a different address `www.xx.yy.z`, but one on the same subnet, for the VM. Replace “`yy:yy:yy:yy:yy:yy`” with the MAC address for Network Adapter 2.

### 4.3 Testing

Reboot the virtual machine. Run `ifconfig -a` to check that all network interfaces are up. In particular, check for `inet` addresses for interfaces `enp0s3` and `enp0s8`. Interface `enp0s8` should have the static IP address `www.xx.yy.z` that you specified in the configuration file.

Verify that `enp0s3` is still working correctly. Since `enp0s3` is the interface that connects the VM to the Internet, you can do this with `ping google.com`. If packets can reach `google.com`, then `enp0s3` is configured correctly. You need this to work correctly for submitting your work.

On your host machine, start a shell<sup>2</sup> and try `ping www.xxx.yy.z`. If packets can get through, then you are ready to configure the web server.

## 5 Basic Web server

First, you will configure a basic web server on the VM, and see how to access it from a browser on the host machine.

1. Start the `httpd` service, and set `httpd` to start at boot time using `systemctl`.
2. You can test the server locally (on the VM) with `curl` or `lynx` and using the URL `http://localhost`. This will fail with a message like “unable to connect to remote host” if the server is not working; otherwise, you should see a test page that says that the server is working but has not been configured yet. (In `lynx`, the test page appears a few seconds after a “403 Forbidden” alert, so be patient!)

---

<sup>2</sup>This works in Windows, Linux, and Mac.

3. Configure the firewall so that HTTP packets are accepted (in the default “zone”). This change should be permanent (i.e., it should persist when the VM is rebooted). You may disable the firewall if necessary for testing, but in the end you should have the firewall running and allowing HTTP packets through.
4. You should now be able to open a browser on the host machine, and using URL `http://www.xx.yy.z`, obtain the same test page that you saw on the virtual machine.
5. Access logs are kept under `/var/log/httpd` on the virtual machine; these can be *extremely* helpful for server debugging.

From this point on, all testing of the server may be done with a browser in the host machine.

## 6 Static content

Create some simple static content. Save an HTML file in the appropriate location so that it is visible at URL `http://www.xx.yy.z/`. The “appropriate location” is made up of the “DocumentRoot” directory and the “DirectoryIndex” file, as specified in the Apache configuration file `/etc/httpd/conf/httpd.conf`. Note: **DO NOT** change these values; the Turnin script expects HTML content to be in the default location. Some example HTML:

```
<html>
<title>Foomatic Industries, Inc.</title>
<body>
<h1>Foomatic Industries</h1>
<p>
Foomatic Industries is a wholly-owned subsidiary of Foocorp,
the world's leader in the production of foo.
</p> </body> </html>
```

## 7 Dynamic content

Web pages can be dynamically created by the server. One way to do this is to write a `bash`<sup>3</sup> script (or other executable) whose output becomes the web page sent by the server. These are called “CGI scripts”.

1. Put a simple “hello world” script in the appropriate subdirectory (usually `cgi-bin`):

```
#!/bin/bash
#
echo Content-type: text/plain
echo
echo Hello, world!
```

Save the file as `hello.cgi` and turn on execute permission. You should then be able to view it at URL `http://www.xx.yy.z/cgi-bin/hello.cgi` which should display the output of the script. Of course, you can also execute the script on the server to check its output.

2. You can also use an HTML-version of the script:

```
#!/bin/bash
#
echo Content-type: text/html
echo
echo "<h2>Hello, world!</h2>"
```

---

<sup>3</sup>Bash is probably not the best choice, but we cover it in class. Perl, Python, or PHP is often a better choice.

where the script output should be legal HTML code.

3. Create a CGI script named `time.cgi` that prints the current date and time. Verify that the time changes in a browser when the page is reloaded.
4. Create a CGI script named `procs.cgi` that displays all running processes. Be sure to include user information in this output. Check the script in a browser.
5. Create a CGI script named `env.cgi` that displays the environment variables (using `env`) passed to the script.
6. Create a CGI script named `counter.cgi` that displays the total number of times the script has been executed. You will need to use an auxiliary file for this. The actual count is unimportant, as long as the count increments each time the script runs in a browser on the host machine.

## 8 Submitting your work

Run the Turnin script, as `root`, to submit your work.

## 9 Questions (submit in Canvas)

1. When the server executes one of the CGI scripts you wrote, which user is executing the script?