Installing, Configuring and Using a Web Server

[This lab should take 45 – 60 minutes. Level: intermediate]

Introduction

Perhaps you have created a web page and uploaded it to a site either at your university or through a web hosting company such as Network Solutions or GoDaddy.com. As shown in the figure below, you might create the page on your computer and upload it to a server located somewhere else. The server would be operated by a web hosting company and most of the configuration necessary for the public to access your site is performed for you. You simply create the page and upload it to the web hosting service.

Have you ever wanted to host the site yourself? It's easier than you think—all you need is some software and a public domain name. You can get the software for free from the Internet, and you can purchase a domain name from an ICANN certified registrar for about the same cost your ISP will charge you to connect[1].

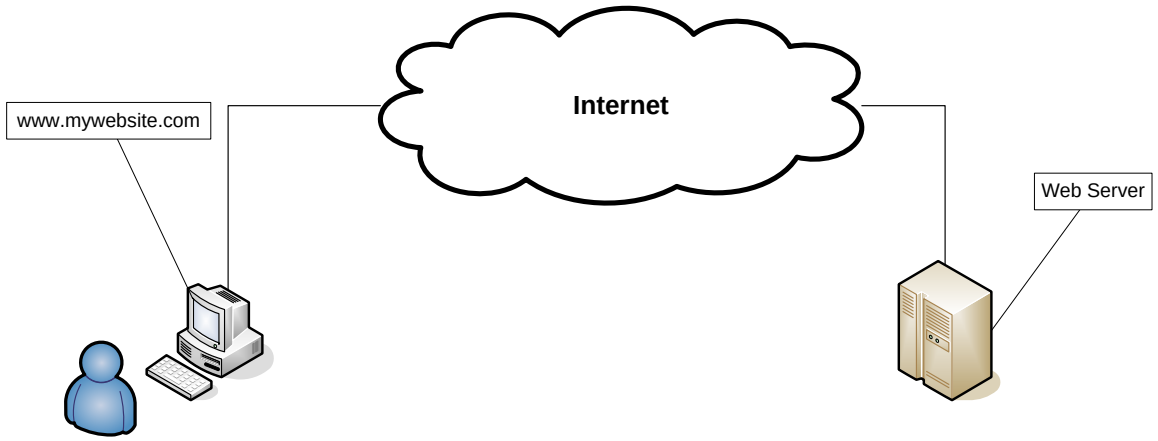

Figure 1: Accessing a Web Server

The Apache HTTP server is an open source web server that forms the basis for most of the commercial web sites in the world[2]. It is a commercial grade product that has enough features to handle the most demanding web sites. The Apache server is configured manually using a text file, discussed later. As you make changes in this file, you should remember that:

a) Syntax in this file is critical, just like any programming language;
b) Changes to the file are not applied until the file has been saved;
c) Once you make changes of any kind (whether to the configuration file or to the DNS server), you should restart the Apache web server in order for it to be able to correctly respond to requests;
d) Comment lines begin with "#" placed at the beginning of the line; and
e) These instructions apply to Windows operating systems.

[1] You must also have an open port (usually 80). Your ISP may charge you extra for this. It may also charge you extra for static IP addresses (see footnote 7).
[2] According to www.netcraft.com

Written by Michael A. Chilton, Ph.D., Associate Professor of Management at Kansas State University

Objectives

The purpose of this lab is to introduce you to web server installation and configuration. We want to be able to host a number of web pages for a set of domains, just as a web hosting company (like an ISP) might do. To accomplish this, we will download the Apache web server and run through the steps of installing it on a Windows machine.

We will first install the Apache software on your server and test it to ensure that it is working. Next we configure the web server to respond to requests for your own company's web page and the two customers' web pages that we previously loaded in the lab setup. We will simulate the adjustments to the DNS server and you should then be able to see all of the web sites using the browser on any client machine on your network.

This lab should be performed in a lab environment that uses a dedicated server so that you can use an unregistered domain name, test it using a work station, and assign a static IP address. You can accomplish all of this without interfering with other networks or the Internet.

Lab Preparation

The setup for this lab will simulate the network depicted in figure 1. These instructions apply to Windows machines only, although you can easily install a web server on a Linux, Unix or Apple machine by following similar steps. You will need two computers—one will act as the server and the other will act as the workstation that requests the web site. Your computers should be loaded with a version of the Windows operating system. Windows 8.1 is preferred and this lab is based on that OS. You will need to assign each computer a fixed IP address manually if you do not have a DHCP server available, and you should assign a fixed IP to the server. If your network is connected to the Internet, it is best if it is "hidden" behind a firewall so that it uses private IP addresses.

You will also need three web pages—one for your own company and one for each of two additional companies that you will host. These can be simple web pages—you need just enough to be able to identify each company's page. If you are unfamiliar with HTML, three sample web pages have been included in the appendix for you. Simply copy these pages to a text editor and save them in separate directories as described next.

Create a web root directory, C:\wwwroot, on your server. Create three sub-directories within this directory: C:\wwwroot\mydomain, C:\wwwroot\company1, and C:\wwwroot\company2. Place the web pages for each company into its respective directory and name each file index.html. This is the file that is initially served by the Apache web server. Although they all have the same file name, they are placed in different sub-directories so that the server can distinguish them.

Procedure

Step 1 Install the web server software

Download Apache HTTP Server, which is freely available from http://httpd.apache.org. The software is well documented, so if you need more detail about how to configure it or how it works than is provided here, you can find answers at the Apache web site. Look for the "Download" link for the most current release. It is usually at the bottom of the second section on the web page. This will lead you to a second page and a link for the stable release. On the next page, you can choose the version you want based on the operating system you have. Select the Win32 binary (*.msi file) without cryptography (SSL), and download it to the desktop on your server.

When your download is complete, you can begin installation by double clicking the file on the desktop. The installation wizard is straightforward and you should accept all defaults for a typical installation.

Figure 2: Installing Apache HTTP Server

When you are presented with the Server Information dialog box, you can enter your domain name (e.g., my-domain-1.com) and your server's fully qualified domain name (e.g., www.my-domain-1.com). You can optionally change the Administrator's email address from the default. This is shown in Figure 3 below.

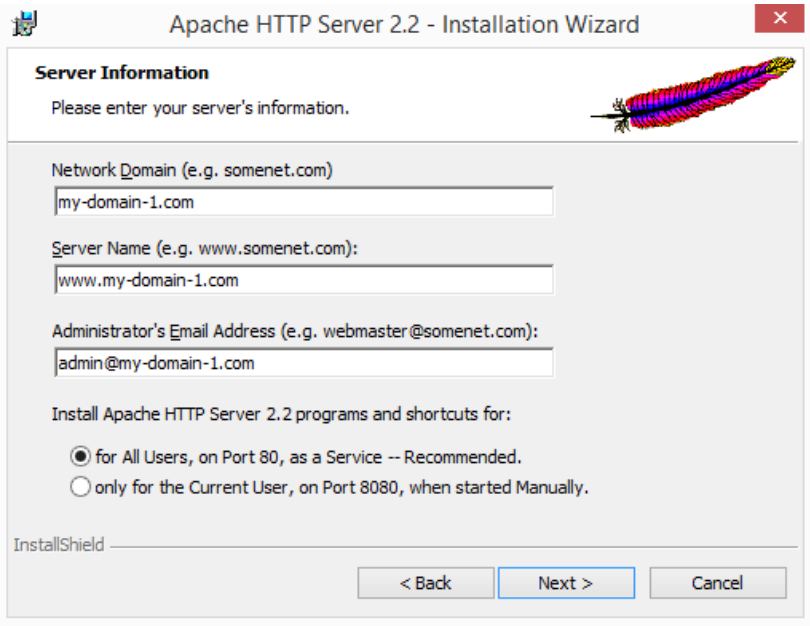

Figure 3: The Server Information Dialog Box

Normally, web servers utilize port 80 and this should only be changed under special circumstances. You can accept the default directory for installation or pick another one that may be more convenient for your setup.

Once you complete the installation, simply close the wizard and verify that the web server service has started. There are several indicators of this. The first is an Apache icon, which appears on your task bar (lower right corner) and it has a green arrow inside it. Figure 4 shows the icon in both the ON (green arrow) and OFF state (red square).

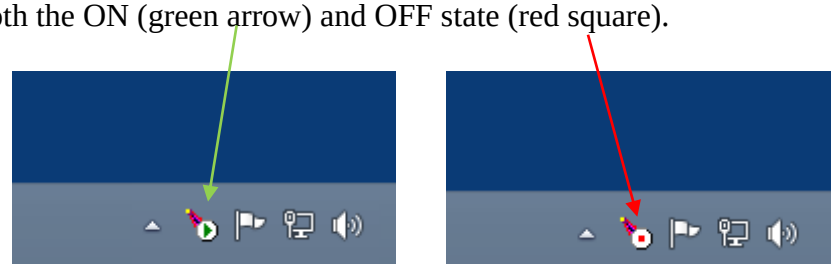

Figure 4: The Apache Icon in the Task Bar

Another method is to check the status of the service by looking at the services page (under Administrative Tools). Figure 5 below shows that the Apache2.2 service has started.

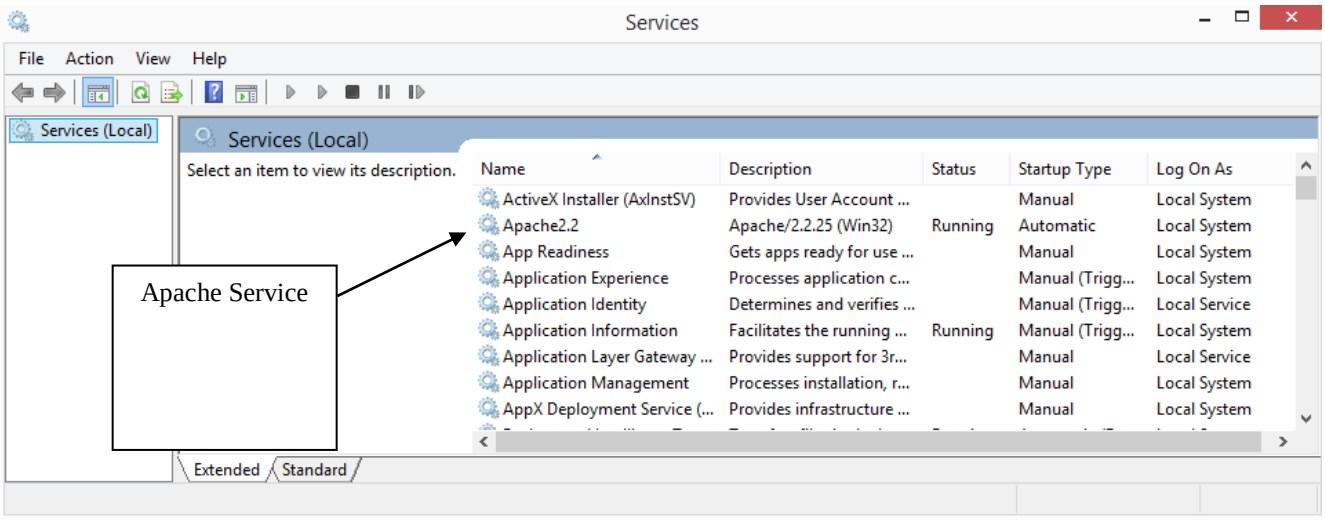

Figure 5: The Services Window

The third (often preferred) method is to call up a web browser and enter either "localhost" or "127.0.0.1" into the URL text box. The server should process the request and retrieve the default web page for you. If your server is running, it will be obvious to you. See Figure 6.

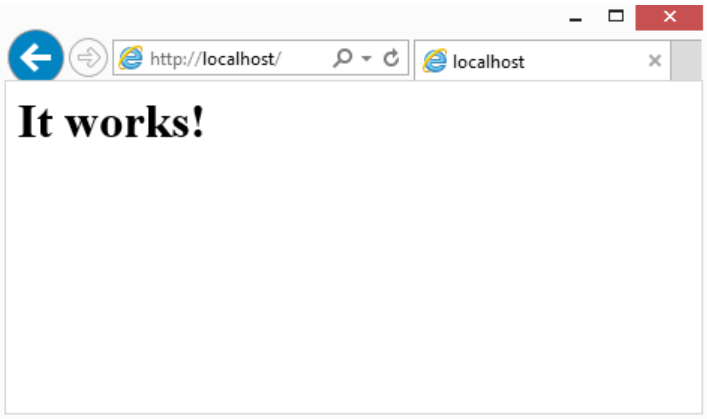

Figure 5: Local browser test: Apache Service running

To permit the Apache web server to communicate through the Windows Firewall, you will need to go to the Windows Control Panel. (Note: the screen captures we provide are for Windows 8.1). Click on “System and Security” followed by Windows Firewall (see Figure 6).

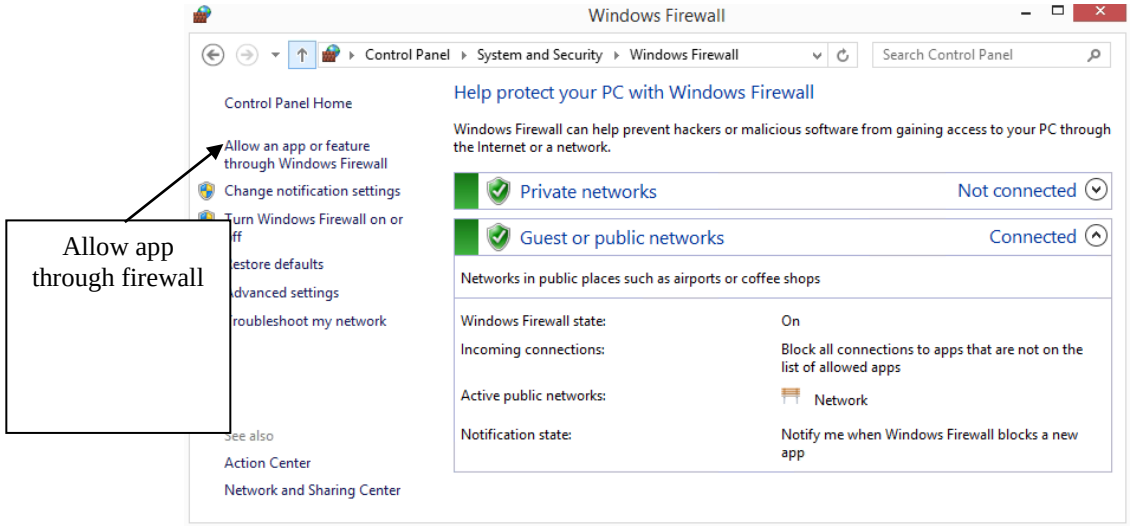

Figure 6: Control Panel ► System and Security ► Windows Firewall

Select “Allow an app or feature through Windows Firewall”. If the ability to allow an app is grayed out, choose the “Change Settings” button. Next pick, “Allow another app…”. If the Apache httpd executable does not appear in the “Add an app” list, choose the “Browse” button. Navigate to your Apache installation directory, and within that the “bin” sub-folder, and choose the “httpd” executable. Press “Open” (Browse window) followed by the “Add” button on the (Add an app window). See Figure 7.

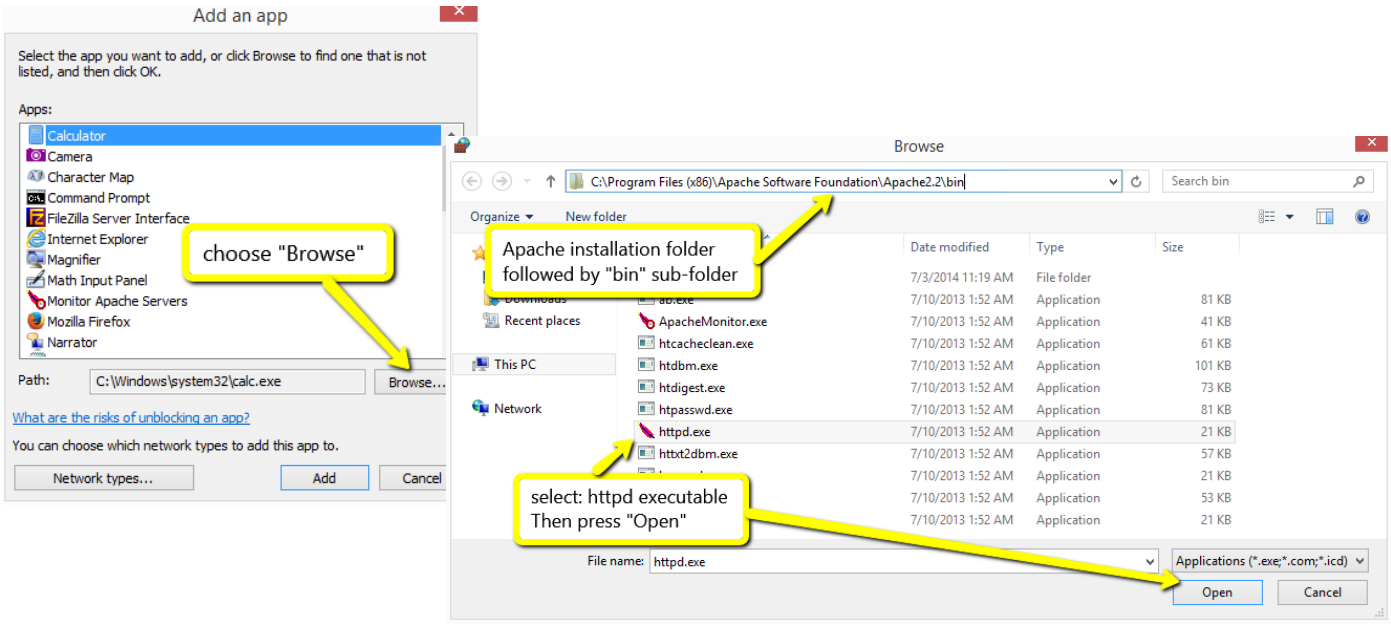

Figure 7: Permit the httpd executable to communicate through the Firewall

When you are done, your Firewall rule for the Apache HTTP Server should look similar to what you see in Figure 8. Press OK to return to the Control Panel.

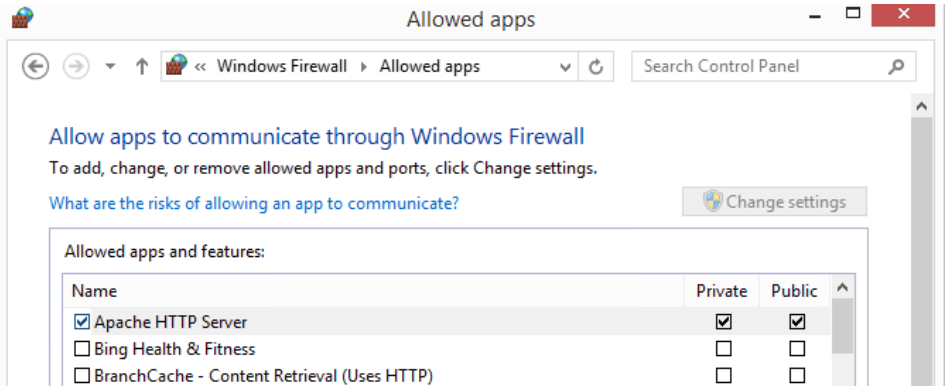

Figure 8: Apache HTTP Server Firewall Rule

Step 2 Configure the web server for your own domain

We would like to get our own web site up and running first, so we will reconfigure our web server to serve our web page and test it locally (on the server machine) to make sure it is working properly. If you completed the lab setup correctly by creating the web pages and saving them to the proper directories as "index.html" then you only need to redirect your Apache server so that it serves these web pages when requested.

In order to redirect the Apache server, you will edit the text file, "*httpd.conf*," which is found in the <<*Apache Home*>>*conf* directory. Before you begin, create a backup copy of this file that you can reopen in the event you make some mistakes. Next, open this file with a plain text editor such as Notepad (you may need to start Notepad with administrative privileges), and you will discover that it is a lengthy file consisting of many different entries and modules. Most lines contained in the file are commented out with the "#" sign[3]. Find the directive, DocumentRoot, and change it to the path that identifies the directory in which you placed your home page. You should have placed your file in c:\wwwroot\mydomain, so that the directive should look like this:

```
DocumentRoot "C:/wwwroot/mydomain"
```

Note that the slashes are forward slashes, not the reverse slashes that Windows uses. There is another section in this file that requires a similar change. It is located a few lines below the DocumentRoot directive and is preceded by the comment, "This should be changed to whatever you set DocumentRoot to." Change it to look like the following:

```
<Directory C:/wwwroot/mydomain>
```

Once again save the file and restart the web server in order for the changes to take effect. An easy way to restart the web server is to left click on its icon in the task bar, select Apache 2.2, then click on the "Restart" or "Stop" option in the popup menu. If you clicked "Stop," just repeat this process, but click on the "Start" option.

When the server indicates that it is running, open a web browser and type "localhost" in the address window. The server should display your company's web page. If it does not, make sure that you saved the configuration file and restarted the web server.

[3] Many of the commented lines are explanatory to help you understand the function of each section.

Step 3 Configure the additional domains

Now that we have our own web site functioning, we are ready to add some additional domains and host the web sites for our customers. We will set up sites for two of them. In order to accomplish this, you must create "virtual hosts." Use *www.new-company-1.com*[4] and *www.new-company-2.com* as the domain names for your two customers.

Apache supports two methods of virtual hosting. You can either assign a different IP address to each domain (IP-based virtual hosting), or you can assign all the domains you intend to host to a single IP address (name-based virtual hosting). We will use name-based virtual hosts and use your server's single IP address for all of the web sites we create.

In order to add your customers' domains, you must change the main configuration file and add some "virtual hosts." You can either modify the *httpd-vhosts.conf* file, which is located in <<*Apache Home*>>\conf\extra directory or simply add some lines to the main configuration file. We will do the latter. Simply add the following lines to the end of your *httpd.conf* file:

```
NameVirtualHost *:80

<VirtualHost *:80>
DocumentRoot "C:/wwwroot/mydomain"
ServerName www.my-domain-1.com
</VirtualHost>

<VirtualHost *:80>
DocumentRoot "C:/wwwroot/company1"
ServerName www.new-company-1.com
</VirtualHost>

<Directory C:/wwwroot/company1>
Allow from all
</Directory>

<VirtualHost *:80>
DocumentRoot "C:/wwwroot/company2"
ServerName www.new-company-2.com
</VirtualHost>

<Directory C:/wwwroot/company2>
Allow from all
</Directory>
```

Restart your Apache webserver.

Note that each web site that you intend to host must have a separate set of VirtualHost tags[5]. There are only two lines of code needed between these tags for our purposes: ServerName and DocumentRoot. There are other directives that could be included here as well, and if you're interested you can look these up in the Apache documentation.

[4] You may find that the URL www.new-company-1.com is already taken and your browser will direct you elsewhere. If this happens, try another name that is not in use on the Internet. You can check domain names at www.whois.net

[5] This includes the main host, because Virtual Hosts directives override the Main Server directive.

When your server receives a request for `www.new-company-1.com`, it will serve the *index.html* file located within the `c:\wwwroot\company1` directory. The <Directory> tags are needed to allow public access to the files contained within the directories that hold your web files.

Step 4 Test your server locally

You are now ready to test your server. You will first need to ensure that the web page you view each time you call up a web page is not one that is stored in cache, but is the most current. To do this, open Internet Explorer and click on the Tools icon (Alt-X) followed by "Internet Options". See Figure 9.

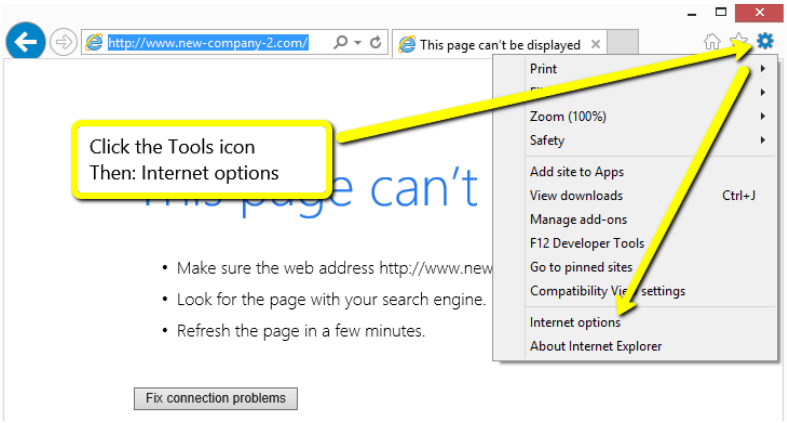

Figure 9: Internet Options

In the "General" tab under the "Browsing History" section click on "Settings". At the top of the new dialog box under the heading, "Check for newer versions of stored pages," select the "Every time I visit the webpage" radio button. See Figure 10. Click "OK" twice to exit.

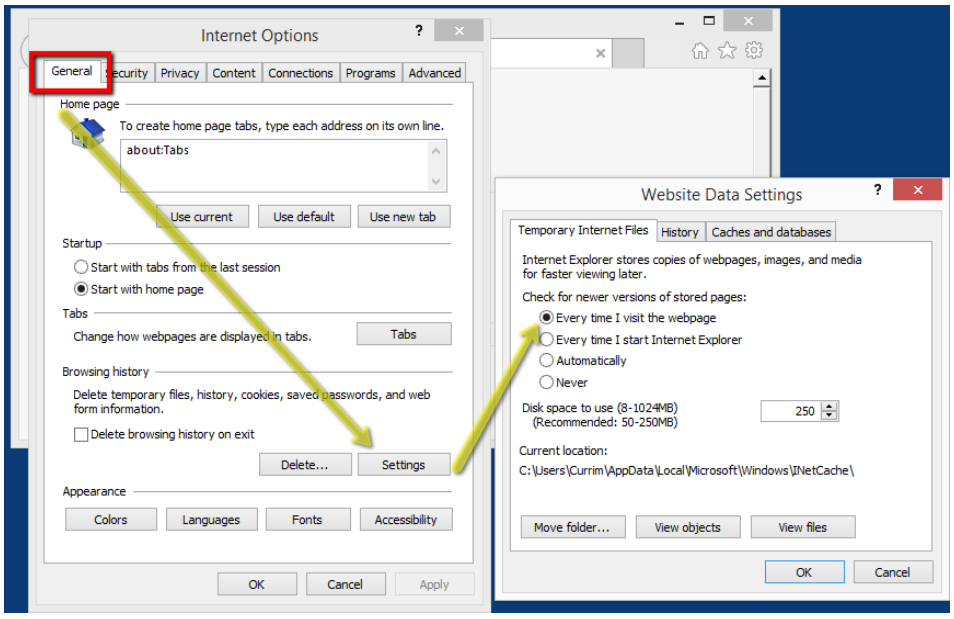

Figure 10: Check for newer versions of stored pages

By default, Internet Explorer uses Protected Mode for websites. We need to inform the browser that our hosted sites are Trusted Sites (or part of the Local intranet). To add our domains to the Trusted sites list, click on the Tools icon followed by "Internet options". Click on the "Security" tab and the icon for "Trusted sites". Press the "Sites" button to add new sites to the list. Ensure the "Require server verification (https:)…" option is not checked. See Figure 11.

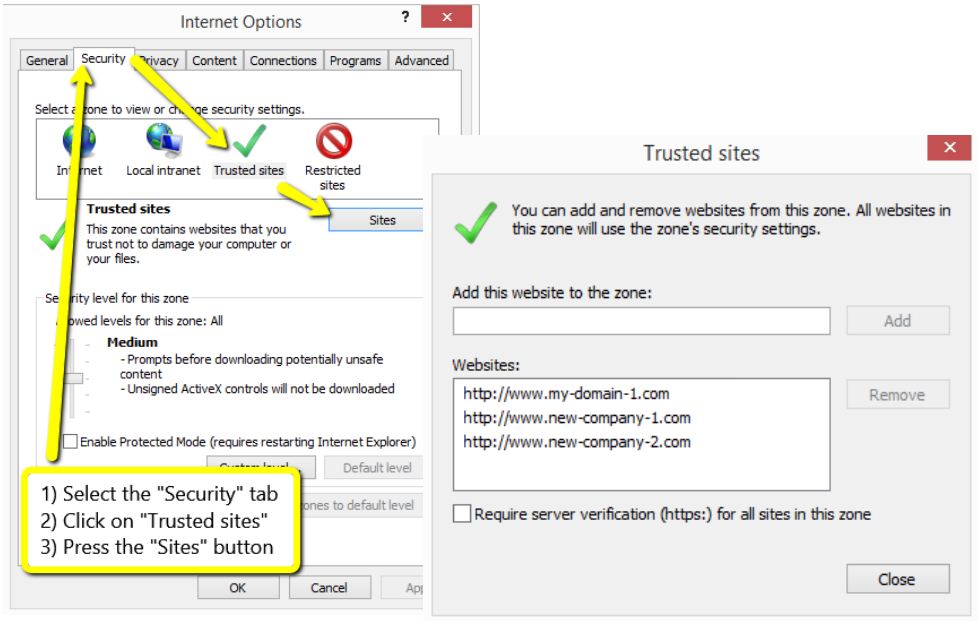

Figure 11: Trusted Sites

Press the "Close" button followed by the "OK" button to return to the main browser window.

Note: If the web browser times out or you have other errors, the chances are that your configuration file has been corrupted or is incorrect in some way; or, more likely, that you forgot to save it and/or restart the web server.

Step 5 DNS Redirection

If you have access to the DNS server for your network and can make changes to it, you will need to tell it that these URLs (domains) exist. This allows general access to your web sites from other networks. On your domain controller, go to the DNS console and add an Active Directory-integrated forward lookup zone for each domain you created. (Note: your own domain should already be there.) Name each new zone with the corresponding domain name: *www.new-company-1.com* and *www.new-company-2.com*. You would then add a host (www) to this zone by right clicking the zone name and selecting "New host…" from the popup menu. In the Name text box, type in "www" and enter the IP address for your server. You can leave the "Create a PTR record" option deselected.

If you do not have access to the DNS server, you can use the following to simulate the required DNS entry. You can modify the "hosts" file on the client computer (located in the c:\windows\system32\drivers\etc directory) by loading it into a text editor and adding the following lines to the end of the file (a sample hosts file is provided in the appendix):

```
192.168.0.1       www.my-domain-1.com
192.168.0.1       www.new-company-1.com
192.168.0.1       www.new-company-2.com
```

You should substitute the IP address of your server for `192.168.0.1`. These entries simply identify the IP address of your web server so that the web requests are sent to the correct computer. This method is especially helpful if you find that the domains you use are already taken and in use on the Internet, as it will supersede any DNS redirection.

Although we are working in a lab environment using private domain names, public web sites require that the IP address of your server be provided to your ICANN approved registrar. This step is necessary to allow people from the Internet access to your web site. It simply tells the registrar's server where your domain name is located and forwards all requests for your web site to that location.[6]

Step 6 Test your server remotely

Your web browsers should now display the web pages that you installed in each directory simply by typing in the domain name for each as a URL. Go to a different computer within your work group and try to connect to all of your sites by starting a web browser and typing in the URL for each: *www.my-domain-1.com*, *www.new-company-1.com*, and *www.new-company-2.com*. The web pages you saved should then be displayed in the web browser.

Note: If for testing purposes you want to permit your Windows 8.1 machine to receive pings, please see the Indiana University knowledge base article at: https://kb.iu.edu/d/aopy

Step 7 Troubleshooting

In case you have difficulty, there are three general areas that cause most of the problems. The first is to remember to save your *httpd.conf* file and restart the web server. Secondly, you should check the DocumentRoot directive to make sure that the web server is pointing to the correct directory; this includes providing public access to the directory within the <Directory> tags. Thirdly, check to make sure your VirtualHost tags are correct.

There are other areas that may cause problems, and these can be found at the Apache web site. Most problems occur from some sort of configuration error, but you can also check that your web pages render properly by opening them directly in a web browser (using the File ► Open menu).

Questions:

1. Discuss some of the precautions you should take before taking your web site "live" on the Internet. There is a web site called "Shields Up" that can help you answer this question. You can access it here: http://grc.com/default.htm.Find the Shields Up link on this page and follow it. Perform a port test on your server. Report your results.
2. This lab focused on name-based virtual hosts, but mention was made of IP-based virtual hosts. Discuss IP-based virtual hosts and how they differ from name-based virtual hosts. Is it possible to host more than one web site using different IP addresses if your server has only one network interface card (NIC)? (A good resource for this is the Apache documentation.)
3. What does URL stand for? Compare and contrast the following: URI, URL and URN. (Hint: check http://www.ietf.org/rfc/rfc2396.txt)

[6] For this reason, web servers need to have a static IP address or have it dynamically updated with the registrar; otherwise it becomes a moving target and unreachable. Your ISP will charge you for a static IP address.

Appendix

1. HTML Web pages

The following code is provided for your use. There are three files: the first is a web page for your own company, which is served at www.my-domain-1.com. The other two are provided for two additional companies, which are served at www.new-company-1.com and www.new-company-2.com. These files are basic and are intended only to provide you with easy recognition of each web page. Paste each of the code snippets into 3 instances of a text editor and save them as index.html into their respective directories.

a) Mydomain.com (save to C:\wwwroot\mydomain):

```
<html>
<body>
<h1>This is my company's web site</h1>
<h2>www.my-domain-1.com</h2>
</body>
</html>
```

b) Company1.com (save to C:\wwwroot\company1):

```
<html>
<body>
<h1>This is the first company's web site</h1>
<h2>www.new-company-1.com</h2>
</body>
</html>
```

c) Company2.com (save to C:\wwwroot\company2):

```
<html>
<body>
<h1>This is the second company's web site</h1>
<h2>www.new-company-2.com</h2>
</body>
</html>
```

2. Virtual Hosts Section of the Configuration File (httpd.conf)

You can either add the code below to the end of your httpd.conf file, or you can include it in the http-vhosts file, which is located at <<*Apache home*>>conf\extra\. If you use the http-vhosts file, you will need to uncomment the line, "Include conf/extra/httpd-vhosts.conf" immediately below the comment, "# Virtual hosts" in your httpd.conf file. If the first line "`NameVirtualHost *:80`" already exists in the http-vhosts.conf file, you do not need to repeat it.

```
NameVirtualHost *:80
<VirtualHost *:80>
      DocumentRoot "C:/wwwroot/mydomain"
      ServerName www.my-domain-1.com
</VirtualHost>
```

```
<VirtualHost *:80>
      DocumentRoot "C:/wwwroot/company1"
      ServerName www.new-company-1.com
</VirtualHost>
<Directory "C:/wwwroot/company1">
      Allow from All
</Directory>

<VirtualHost *:80>
      DocumentRoot "C:/wwwroot/company2"
      ServerName www.new-company-2.com
</VirtualHost>
<Directory "C:/wwwroot/company2">
      Allow from All
</Directory>
```

3. Entire "Hosts" file (C:\windows\system32\drivers\etc\hosts)

This file is for your client, not your server. Make sure you replace the IP address with the address of your server. You will only need to make this change if you don't have access to the DNS server.

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#  127.0.0.1       localhost
#  ::1             localhost
1

192.168.0.1     www.my-domain-1.com
192.168.0.1     www.new-company-1.com
192.168.0.1     www.new-company-2.com
```

Change these IP addresses to match your server. If you are making the change on the server itself (for testing), you can use 127.0.0.1 (localhost) for the IP addresses.