Chapter 6 Configuring a Web Server

In this chapter, you will:

- Understand how a Web server works
- Install the Internet Information Services (IIS) Manager and Apache Web servers
- Examine the IIS and Apache properties
- Host multiple Web sites
- Configure new Web sites in IIS and Apache
- Understand virtual directories
- Configure HTTPS in IIS

All Web servers are based on the Hypertext Transfer Protocol (HTTP), which governs the way Web servers communicate with browsers and other client software. When you install a Web server, it is configured to perform most common tasks by default, such as display simple Web pages. Although you can configure Microsoft's Internet Information Services (IIS) Manager Web server and the Apache Web server in a similar manner, IIS and Apache use different approaches to configuration. For IIS, you use a GUI, often with wizards, to configure the Web server. For Apache, we will alter text files in the /etc/apache2 directory.

Often a Web server is not what it appears to be. A Web site such as *www.microsoft.com* uses URLs that appear to have a structure of folders similar to those on a hard disk. In reality, that structure doesn't always reflect the organization of the folders on the Web server's hard disk. You can create virtual directories that are part of the Web site yet located outside of the Web site. Conversely, you may think that sites such as *www.MyFavoriteWidgets.com* and *www.WidgetSupplies.com* must represent different servers, or at least different IP addresses. However, you can use virtual servers to configure a Web server to host multiple sites.

Understanding How a Web Server Works

Although Web servers have evolved substantially since their introduction, the main purpose of a Web server is to send HTML documents to a browser. HTML is the formatting language that browsers use to display text and graphics. All Web servers support the **Hypertext Transfer Protocol (HTTP)**, which defines how information is passed between the browser and the Web server. Web servers and browsers must follow the same rules defined by HTTP. This consistency allows someone using a Firefox browser or an Internet Explorer browser, for example, to access the same pages on any server. Internet Explorer, Firefox, and other browsers may differ in terms of the HTML that the Web designers use to create Web pages, but the Web server attempts to render the same HTML to the browser.

The two most popular Web servers are Apache from the Apache Software Foundation and Internet Information Services (IIS) Manager from Microsoft. According to Netcraft (https://news.netcraft.com/archives/category/web-server-survey/), approximately 46% of the Web servers use Apache and 9% use IIS as of this writing. Both servers publish HTML pages and perform other tasks necessary for producing interactive Web pages. Chapter 7 explores the use of programming languages and databases on Web servers to produce interactive Web pages.

After you install a Web server, you can configure it to change the port number on which it listens for Web requests, the location from which the Web server retrieves HTML files (called the root of the server), and the settings that determine the performance of the computer depending on levels of traffic. You can expand the Web server to accept requests from multiple domains, thereby creating **virtual servers**. You can also store HTML documents that are not part of the root file structure by using **virtual directories**. This chapter explores these configurations in detail.

Like DNS servers, Web servers are services (often called daemons in Linux) that listen for requests at ports. Most HTTP traffic listens on port 80 for incoming requests, though they can use any port beyond 1023. Ports up to and including 1023 are reserved for other uses. Often, if a Web server is not running on port 80, it runs on port 8080 or 8000, but using these ports is simply convention. The HTTP Secure (HTTPS) default port is 443. Web server administrators sometimes take advantage of this technique when two Web servers are running on the same computer.
https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers#Well-known_ports

Each Web server has a **root** folder, which is where you store the HTML documents and subfolders for your site. If you install IIS on the C: drive, the root is c:\inetpub\wwwroot. For example, if you used this root for the *www.technowidgets.com* Web site, and you stored a file named hello.htm in c:\inetpub\wwwroot, you could display it from a browser by entering *http://www.technowidgets.com/hello.htm* as the URL.

Understanding HTTP

The current version of HTTP, version 1.1, describes how to format messages that are sent from the browser to the Web server and back. The messages contain specific commands that the server uses to retrieve certain Web pages. Because all popular Web servers and browsers have been supporting HTTP version 1.1 for years, this section focuses on this version.

HTTP is a **stateless** protocol, meaning that each Web page sent to the user is independent of every other Web page the server sends. As a consequence, you cannot use the protocol to keep track of users who are viewing Web pages on your site. For example, if you have an e-commerce site where people buy books, you need to use programming techniques instead of HTTP features to track visitors and the books they buy.

One of the most important characteristics of HTTP 1.1 is its support for **persistent connections**. This capability allows the browser to receive multiple files in one TCP connection. Without a persistent connection, each file sent would require an independent TCP connection, which takes extra processing on the Web server to set up and release. Such extra processing can decrease performance. This potential problem becomes even more significant when you realize that most Web pages contain multiple files. Besides the main file that contains text, the Web page may contain graphic images, audio and video files, scripts, etc.. Each image must be sent to your browser separately.

The following procedure shows the communication between a browser and a Web server that displays a simple page containing the text “Hello,World.” The procedure assumes that a host called *www.technowidgets.com* resolves to an IP address of 192.168.0.100.

1. You type http://www.technowidgets.com/hello.htm in the Web browser.
2. The Web browser contacts the DNS server to find the IP address for *www.technowidgets.com*.The DNS server returns 192.168.0.100.
3. The browser composes the following message and sends it to port 80 on 192.168.0.100:

```
GET /hello.htm HTTP/1.1
Host: www.technowidgets.com
```

4. In this example, the Web server is a Microsoft Web server and responds to the browser with the following message:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Fri, 17 May 2005 18:47:30 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Fri, 17 May 2005 18:21:25 GMT
ETag: "90cbb2a7cffdc11:b50"
Content-Length: 43
<html><body>
Hello, World
</body></html>
```

5. The browser retrieves the message and reads its **header**, which contains information about the page. Each header starts with the header name, followed by a colon. The data associated with the header follow the colon. At the bottom of the message, you see that the page displays the text "Hello,World."

Step 3 is an important one for configuring servers. Notice that when the message is sent to IP address 192.168.0.100, the host name is *www.technowidgets.com*.The host name is separate from the IP address, meaning that the Web server at 192.168.0.100 can look at the host name and then display pages on a different Web site. You will see how to apply this technique later in the chapter.

In Step 5, the headers contain the following information for the browser: the Web server's use of IIS, the current date and time on the server, and the last time the file was updated. The other important information in one header is that the content type is text/html, which defines it as a typical Web page.

Understanding Features in Apache Web Server

Apache's philosophy is to start with minimal features and then expand the server as necessary. For example, programming languages must be added in Apache setup before they are available. Sometimes this is done with a single line in a configuration file. When you install Apache, a directory is set up for an online manual and icons used on Web pages. The only sample HTML document is a single sample test page. This modularity allows you to optimize Apache for the required tasks without wasting processing time and memory on unneeded tasks. You can even use Apache to help secure your Web environment by configuring it as a **proxy server**, which isolates your real Web server from the Internet. A proxy server takes requests for pages from the Internet and transfers them to the real Web server inside your network.

The following are some major features of Apache: http://en.wikipedia.org/wiki/Apache_HTTP_Server

- *Support for Windows*— improved reliability.
- *Support for IPv6, the future version of IP addressing*—Internet Protocol version 4 (IPv4) addressing uses 32 bits which does not provide enough addresses to meet demand as the Internet grows. IPv6 uses 128 bits which greatly expands the number of available addresses.
- *Secure Sockets Layer* - cryptographic protocols designed to provide secure communications over the Internet.
- *Virtual hosting* - allows one Apache installation to serve many different websites.
- *Unicode support in Windows*—Apache supports Unicode, which allows you to use foreign-language character sets in Web pages.

Understanding Features in Internet Information Services Manager

For many Web sites, any version of IIS provides the needed functionality of publishing Web pages. Windows Server 2003 uses IIS 6.*, Windows Server 2008 uses IIS 7.* and Windows 2012 uses IIS 8.*. Older versions of IIS support **Active Server Pages (ASP)** used to create dynamic pages. ASP.NET was first introduced with Windows 2000. Microsoft's current development environment is ASP.NET Version 4.5. You work with ASP and ASP.NET in Chapter 7.

Some of the key features in IIS 5.0 include:

- *Web Distributed Authoring and Versioning (WebDAV)*—WebDAV allows the server to share Web-based files.
- *Named virtual hosting*—If you are running more than one physical Web server, virtual hosting lets them all use single IP address.
- *Per Web site bandwidth throttling*—This feature allows you to control the amount of bandwidth that each site on your server consumes.
- *Kerberos*—The Kerberos authentication protocol is integrated with the Web server, which allows you to implement more secure authentication.
- *Secure Sockets Layer 3.0*—This technology supports encrypted communication between a Web server and a browser, increasing the security of a Web site.

Some of the key features in IIS 6.0 include:

- *Increased security*—The default installation for IIS 6.0 is the "locked down" mode, which permits only HTML files to be used. IIS 5.0, by default, supports ASP and other methods of creating dynamic pages. Allowing only HTML files significantly reduces attackers' ability to extract important data.
- *Expanded language support*—**XML** and **SOAP** are important components used in application development. You will learn about both in Chapter 7.
- *Support for IPv6, the future version of IP addressing*—As mentioned in the "Understanding Features in Apache Web Server" section, the current method of IP addressing uses 32 bits, which does not provide enough addresses to meet demand as the Internet grows. Internet protocol version 6 (IPv6) uses 128 bits, which greatly expands the number of available addresses.
- *Increased dependability*—IIS 6.0 increases dependability through **kernel-mode** HTTP service and a self-healing mechanism. A kernel-mode service is protected from being corrupted by another program.

Some of the features new to IIS 7.0 and 7.5 include the following:

- *Increased security*— Reduced surface attack and Transport Security Layer
- *Forms-based Management, Improved WebDAV, FTP, and Powershell support*

Some of the features new to IIS 8.0 and 8.5 include the following:

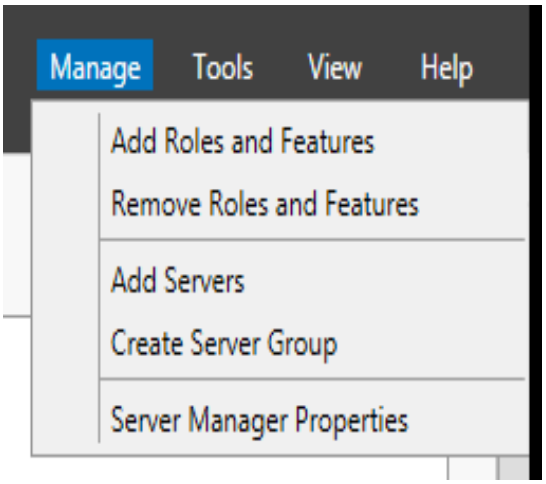
- *SSL Certificate Support, Dynamic Site Activation and Enhanced Logging*

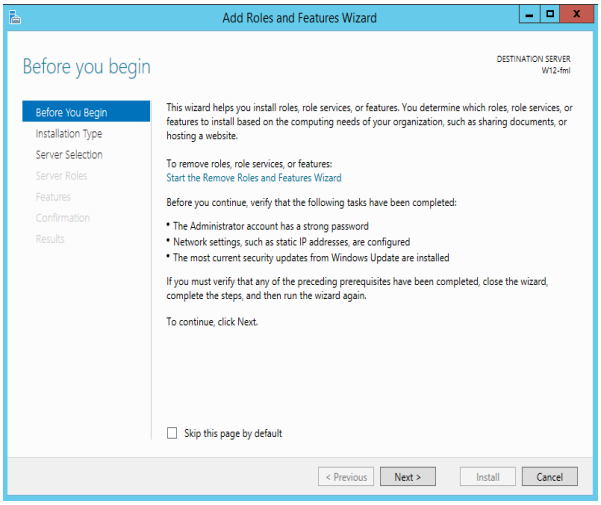
INSTALLING WEB SERVERS

Each version of Windows or Linux provides a Web server that is easy to install and runs without any extra configuration. However, they use very different installation procedures.

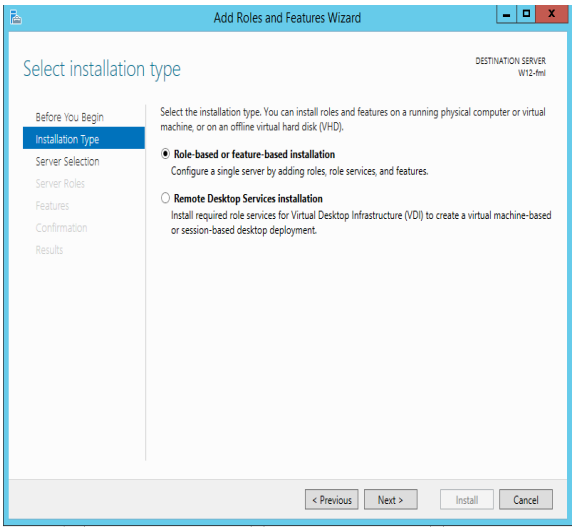
Activity 6-1: Installing IIS in Windows Server 2012R2.

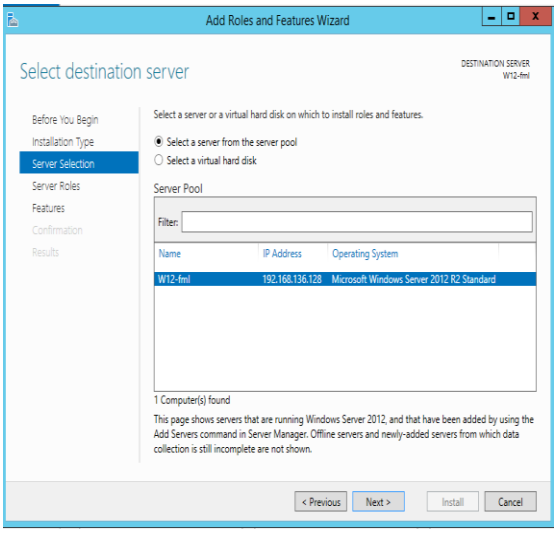
1. From Server Manager, choose **Manage > Add Roles and Features > Next**



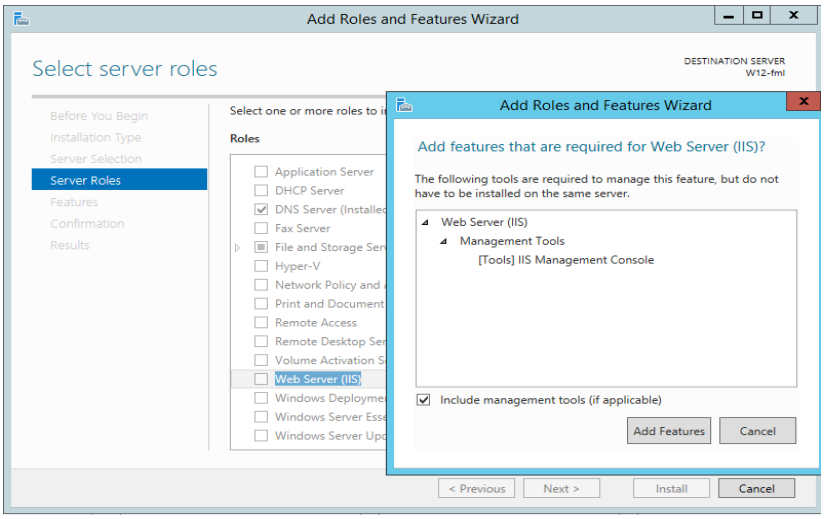


2. The Installation Type is Role-based. Click **Next.** Your server should be selected by default. Click **Next**.

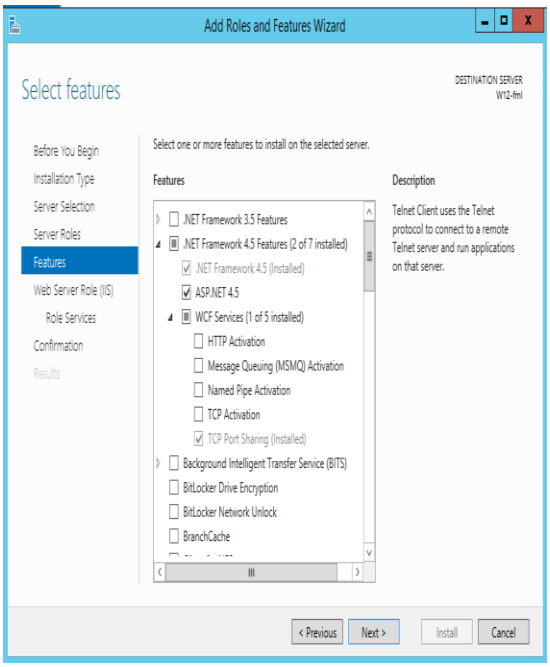


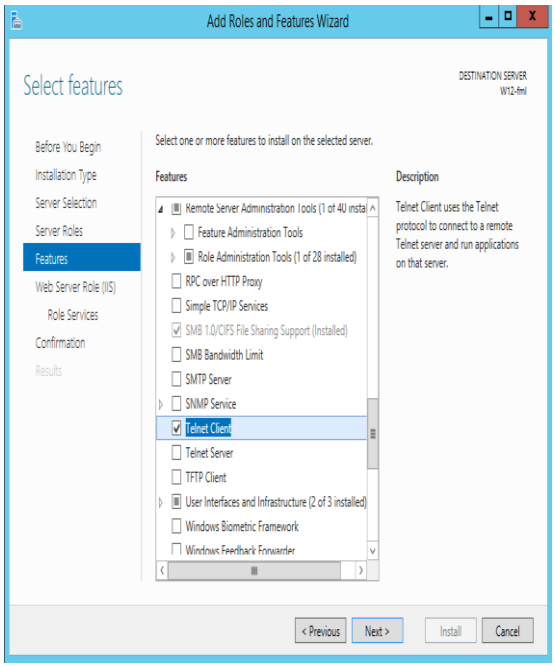


3. Select the Web Server (IIS) Role and click **Next**.

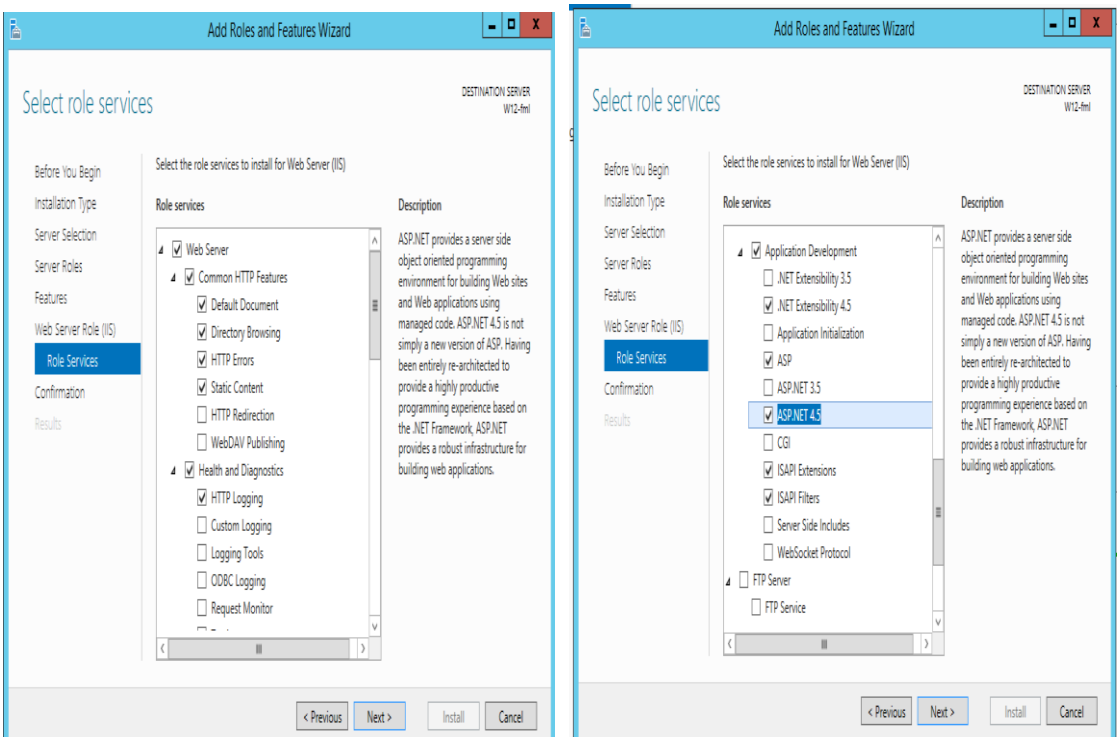


4. Expand **.NET Framework 4.5 Features** and select the features **ASP.NET 4.5** and **Telnet Client**. Click **Next**.

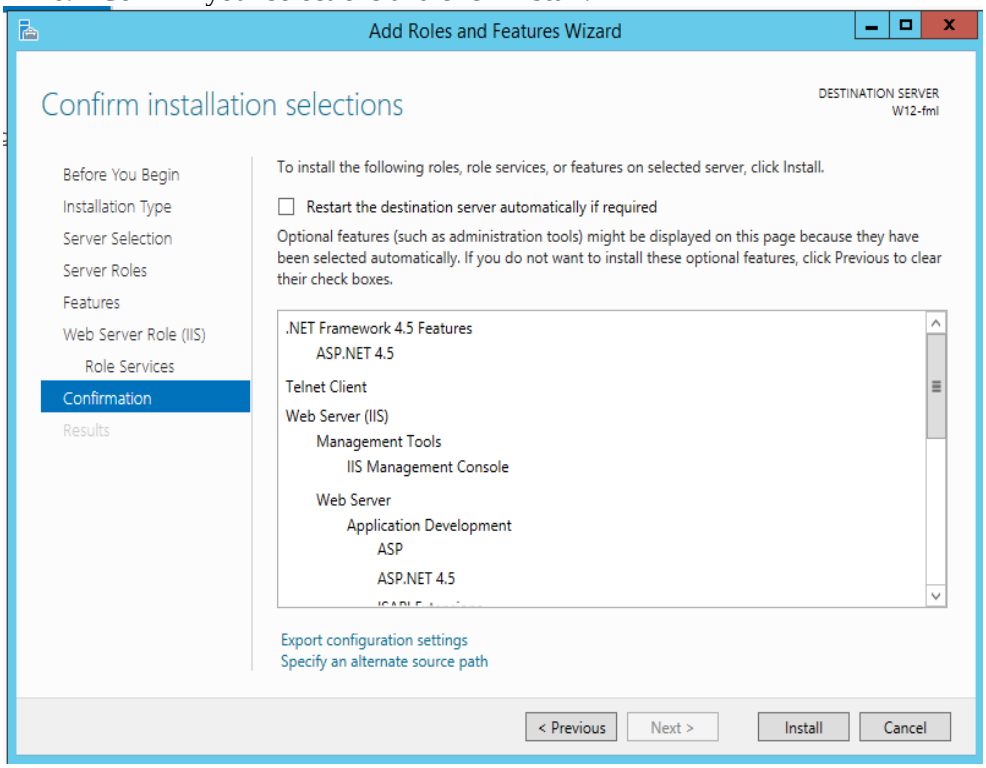


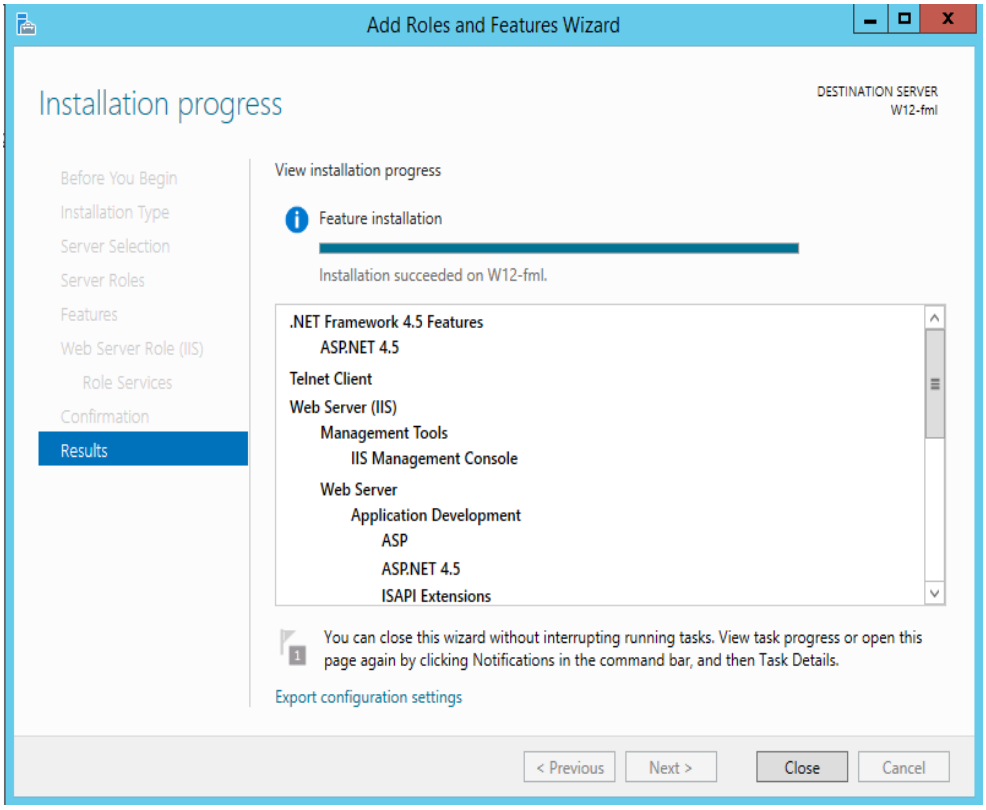


5. Choose **Next** for Web Server Role (IIS) and select the following Application Development Role Services (carefully select all of these!): **.Net Extensibility 4.5, ASP, ASP.NET 4.5** and click **Add Features** to add ISAPI Extensions & ISAPI Filters. Click **Next**.



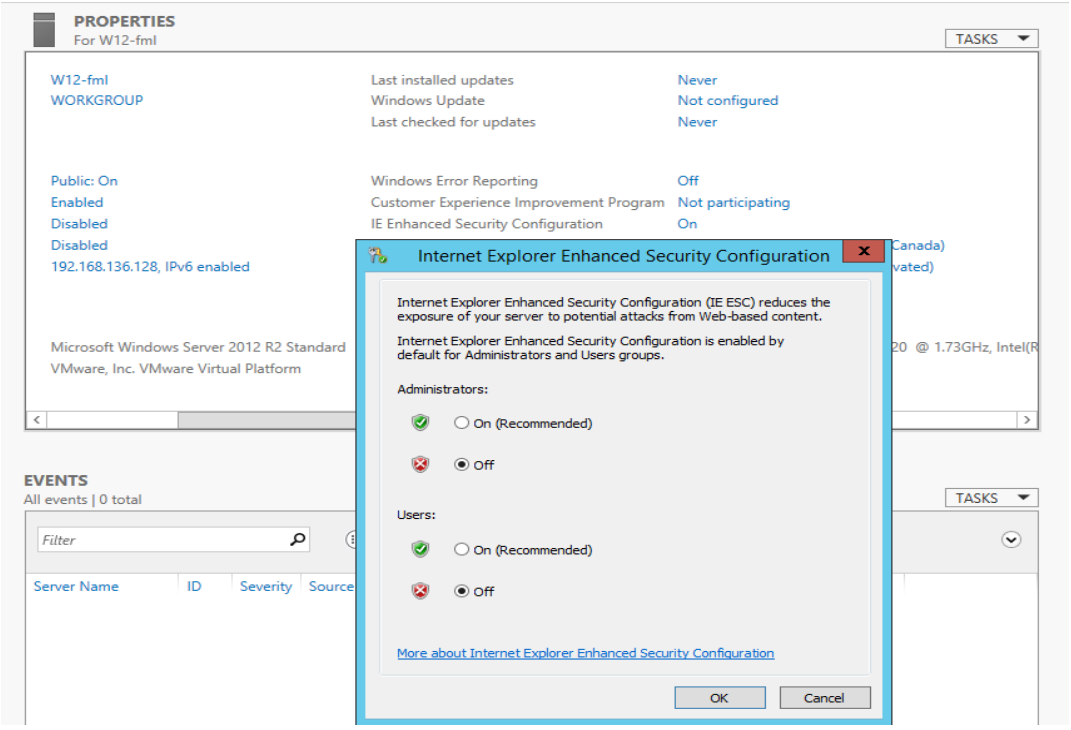
6. Confirm your selections and click **Install**.



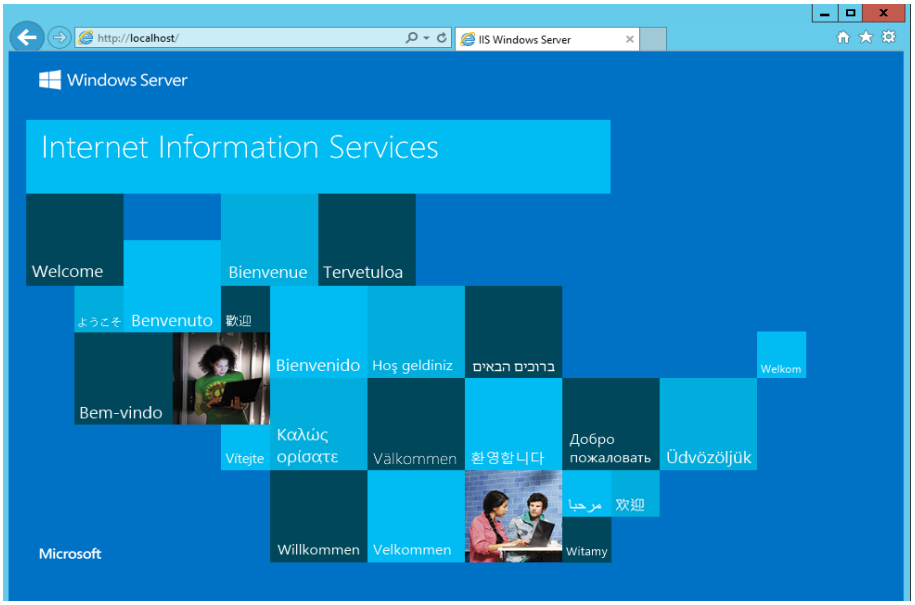


Click **Close** when the installation is complete. Double check to be sure you have added all the features.

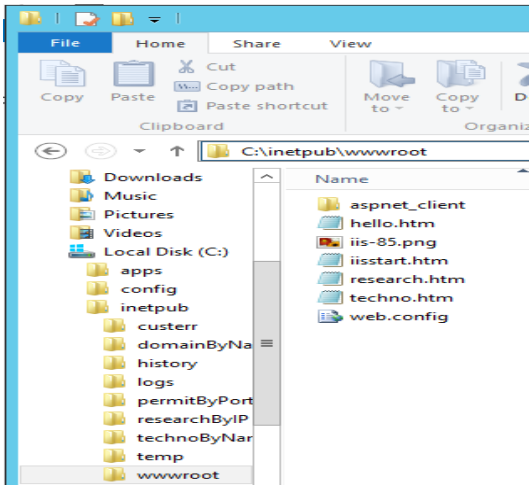
7. Also go Server Manager > Local Server and click **On** next to I.E. Enhanced Security and turn it **Off** for both Adminstrators and Users.



8. The proof is in the pudding, or rather the localhost. Open Internet Explorer and type **localhost** in the address text box. **Submit a screenshot** of the default IIS webpage. Be sure to put your name in the screenshot. Note: You should also be able to visit **http://www.technowidgets.com** in your browser. If Internet Explorer doesn't correctly open this page, try downloading Google Chrome and use that browser. No screenshot required, but realize that this must work for future exercises in this chapter.

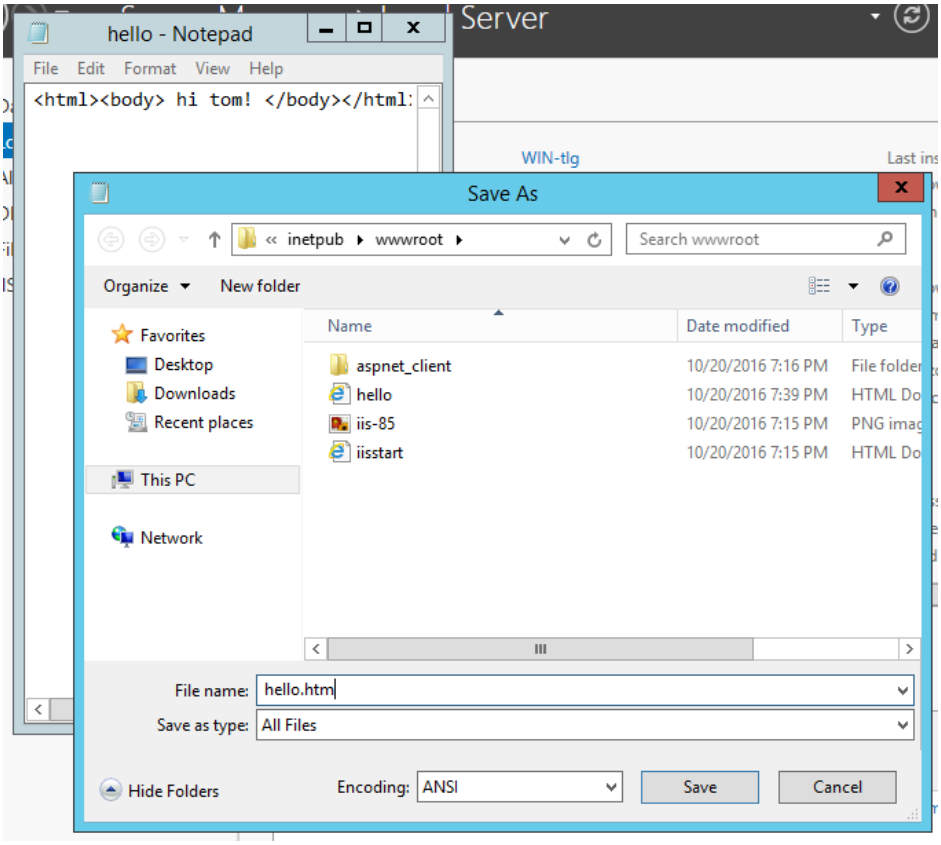


After installing IIS in Windows, there is a menu item in **Server Manager > Tools > Internet Information Services Manager**. There is also the folder c:\inetpub\wwwroot\ where web pages are stored by default.

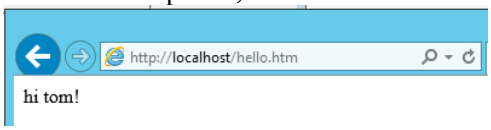


9. Now we'll test a sample web page using the localhost domain. From the Start Screen, type **Notepad**, right-click the **Notepad** icon, and choose **Run as Administrator**. Click **Yes**.

10. Enter the simple html script (use **your name**) in the small Notepad window below and save it (select **Save as Type > All Files**) as **\inetpub\wwwroot\hello.htm**. Click **Save**.



11. In Internet Explorer, test **localhost/hello.htm** and **submit a screenshot**.



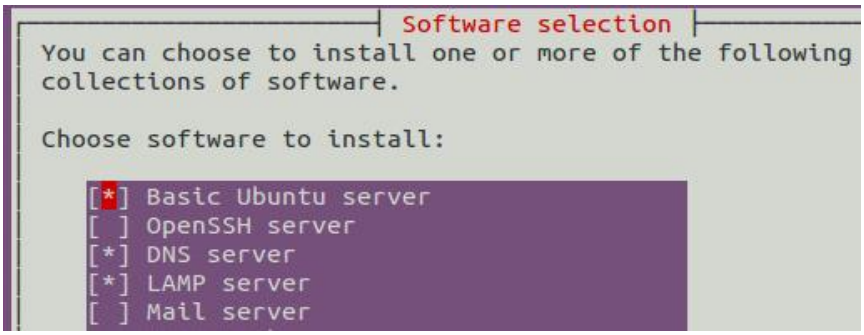
Note: As with the previous page, you should also be able to visit **http://www.technowidgets.com/hello.htm**. Be sure this works in Internet Explorer or Chrome.

Activity 6-2: Installing Apache in Ubuntu 14.

When you install LAMP Server, you are installing Apache, MySQL and PHP.

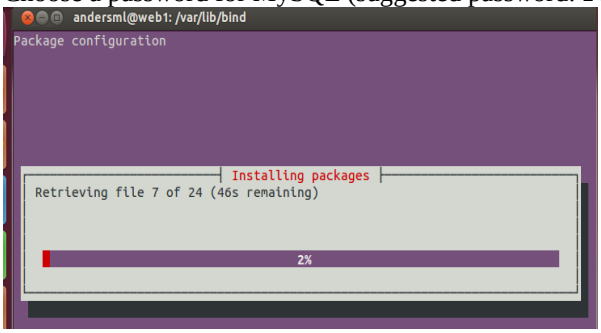
1. From the Terminal App, Install tasksel (task select)
   sudo apt-get install tasksel
2. Type **sudo tasksel**, press **ENTER**, and use the **down arrow** key to highlight LAMP Server. Press the **space bar** to select LAMP Server and press **ENTER** to do the install. (Notice that you've already installed Ubuntu Server & DNS).

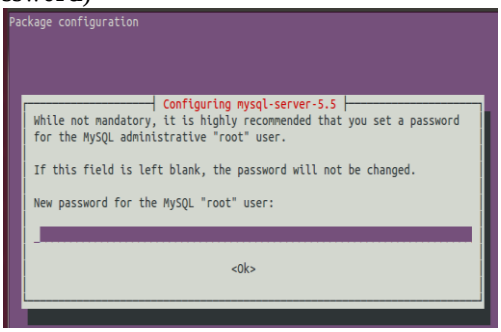
   Note: If this fails with an "aptitude failed" message, run these two commands to install LAMP:
   sudo apt-get update
   sudo tasksel install lamp-server



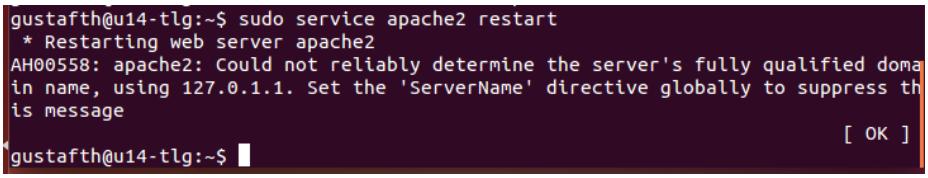
Note: the use of tasksel in not recommended for uninstalling LAMP.

3. Choose a password for MySQL (suggested password: **P@ssw0rd**)





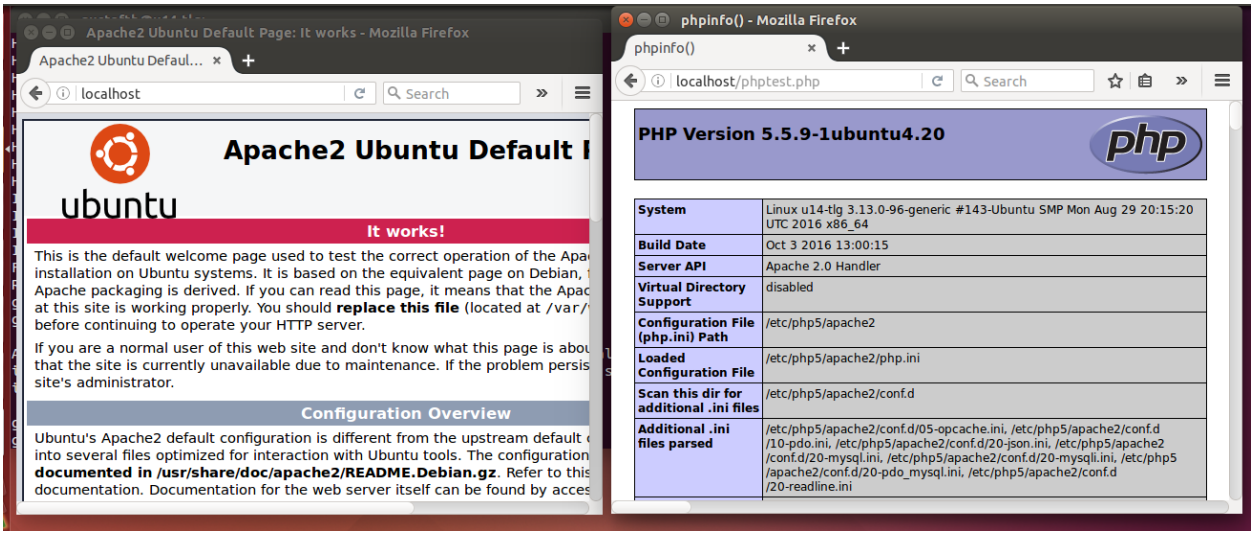
4. Next make sure the service will restart (other parameters for the apache2 service: start, stop, reload) **sudo service apache2 restart**



5. Create this php script using **nano** and save it as /var/www/html/phptest.php

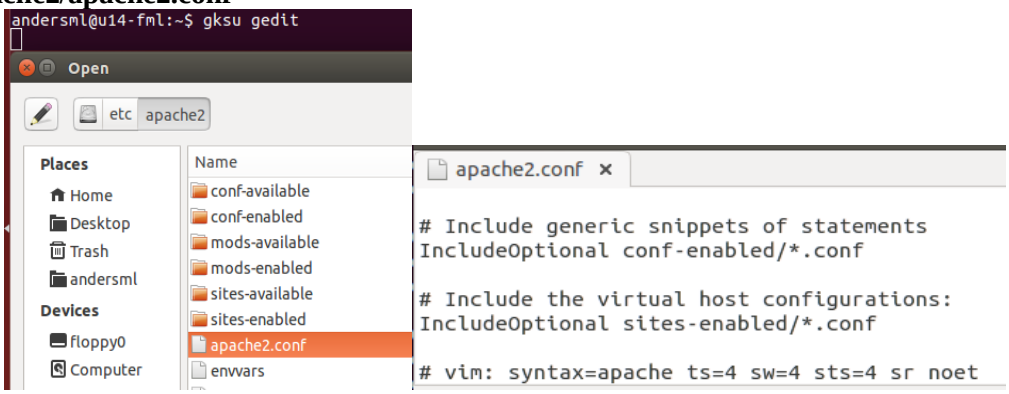
```
<?php   phpinfo();  ?>
```

6. The proof is in the pudding or rather the localhost. **Submit screenshots** of the default Apache webpage (type **localhost** in the Firefox address text box) and **phptest.php**. Hints: to open a new Firefox window, click the button in the upper right and choose **New Window**. To resize the windows and display them side by side, double click the title bar and then you can resize**.** Be sure to put **your name** in all screenshots!

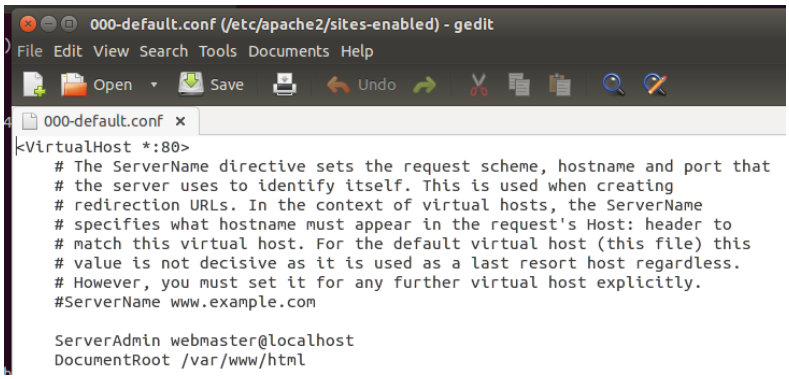


Next let's examine Apache2 configure files we just installed and make a sample web page.

7. Run gedit from the terminal window as a superuser (or use nano if you prefer):
   gksu gedit
8. Choose: **File > Open** and under Devices > **Computer**, open Apache's configuration file **/etc/apache2/apache2.conf**



9. Notice at the end of the file that sites-enabled/*.conf files (in /etc/apache2/) are included. This is where we'll setup site-specific configurations. Next, open **/etc/apache2/sites-enabled/000-default.conf**.



10. Notice that all port 80 traffic directed to localhost has a DocumentRoot of /var/www/html.

To create a hello.htm Web page for localhost in this director

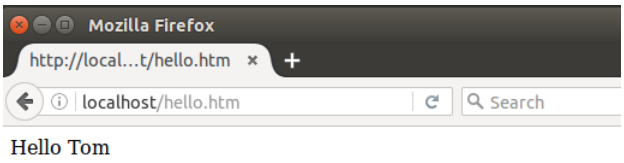
11. Choose: File > New and type the following HTML code (use your name):

```
<html><body>
Hello Tom!
</body></html>
```

12. Click **File > Save As: > Computer > var > www > html** and type **hello.htm** in the Name text box and click the **Save** button.

To display the Web page in Linux:

13. Run Firefox and key in **localhost/hello.htm** and press **ENTER**.
14. Submit a **screenshot of the browser**.

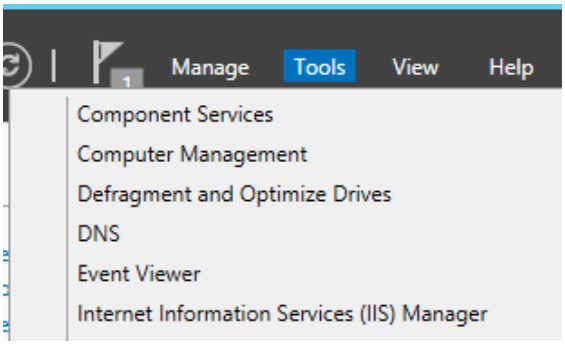


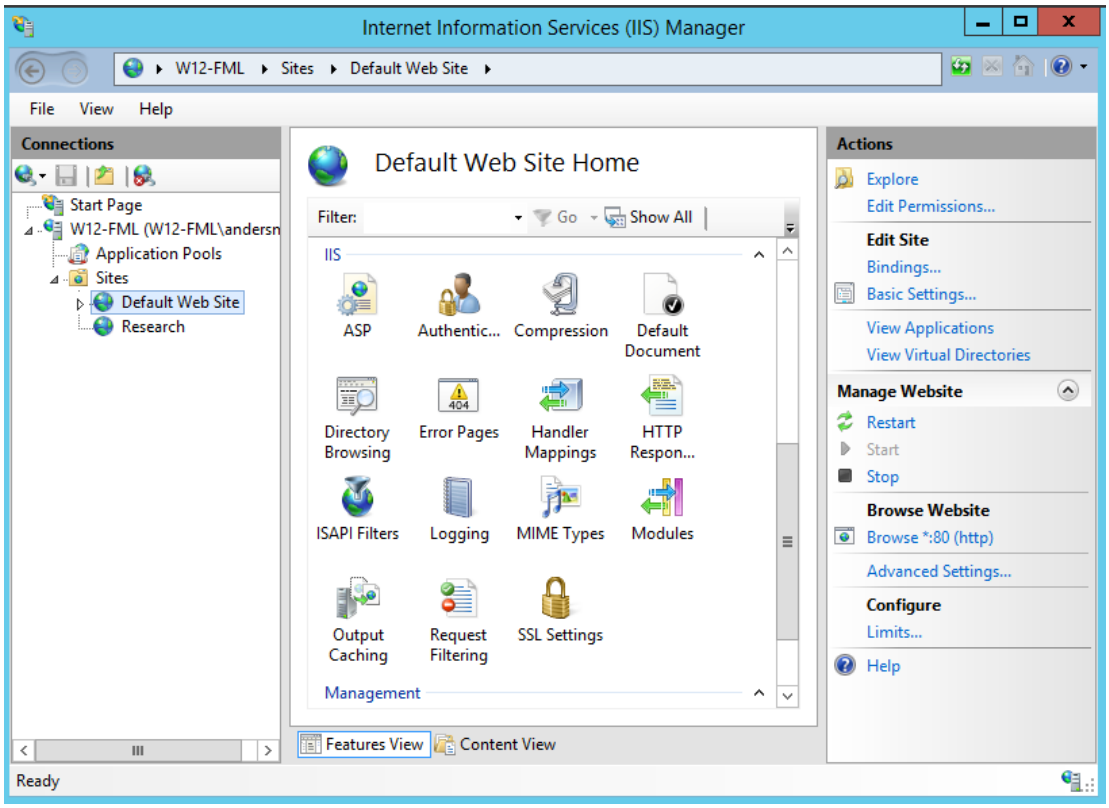
EXAMINING INTERNET INFORMATION SERVICES MANAGER PROPERTIES

The Windows Setup program configures settings in IIS that meet the demands of most Web sites. In many cases, it is not necessary to make any changes. We've configured IIS to support both ASP and ASP.NET. However, as your Web site grows and changes, it will probably be necessary to adjust the settings and other properties of IIS. This section defines those properties. You will learn why and how to change these properties later in this chapter and in subsequent chapters.

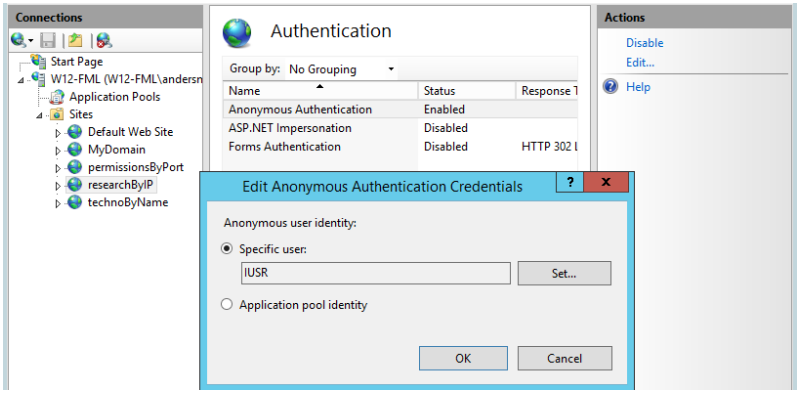
To display the properties of the default Web site on IIS:

1. Choose the Tools drop down menu and click **Internet Information Services (IIS) Manager**. The Internet Information Services (IIS) Manager window opens.



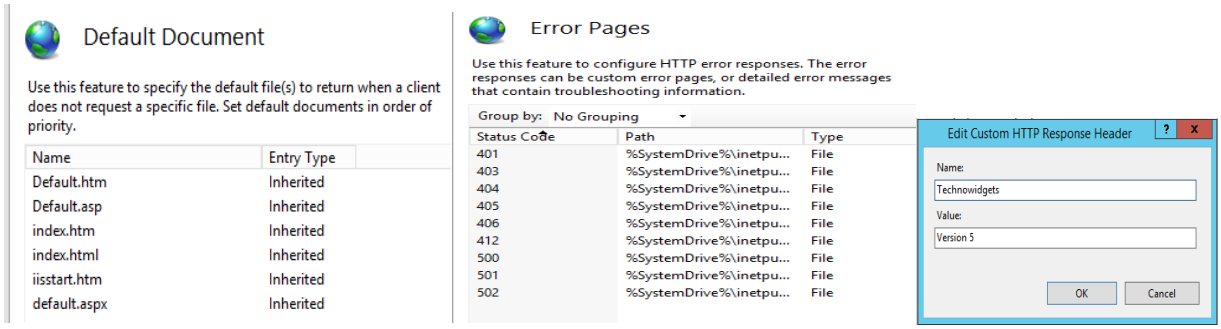


2. The left pane called **Connections** is used to navigate the IIS Server. Expand the entry with your server name and then the **Sites** entry to navigate the Server's Web Sites. Choose **Default Web Site** before exploring the middle and right panes.
3. The middle pane has two tabs at the bottom.
a. Please look at these features in the 'Features View' tab to prepare for Chapter 6 Projects:
- Note that you can find any of these icons by beginning to type its name. Double-click **Authentication** - controls anonymous access. When you installed IIS, a user was created called IUSR. This guest account only has access to your Web site. When people request a Web page, they are logged on to the server using this account. Click on **Default Web Site** to return to the home page.

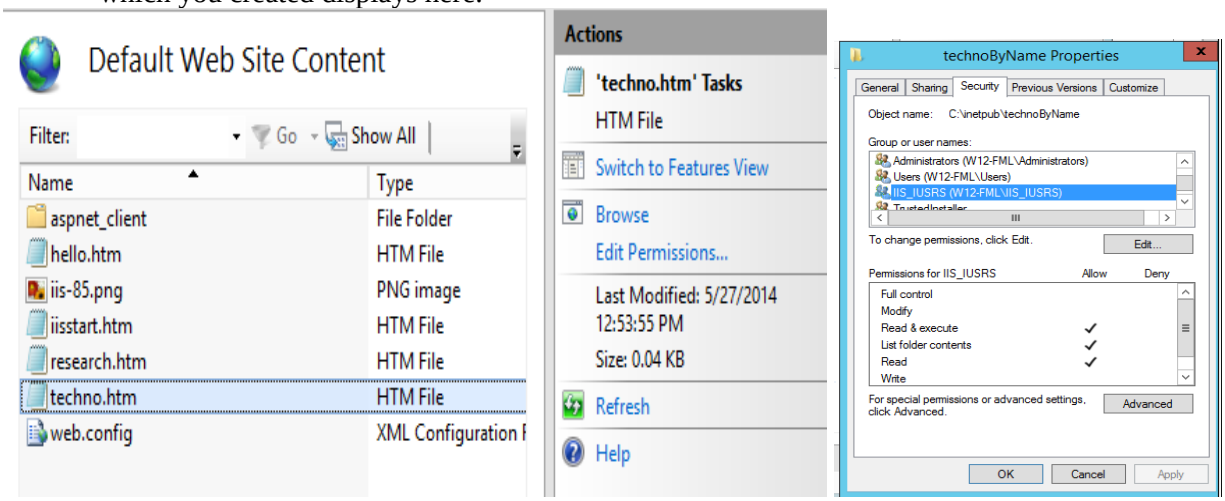


- Double-click **Default Document** – an ordered list of default web pages for this site including iistart.htm which you viewed in Activity 6-1 step 8, even though you didn't include this name. Click on **Default Web Site** to return to the home page.

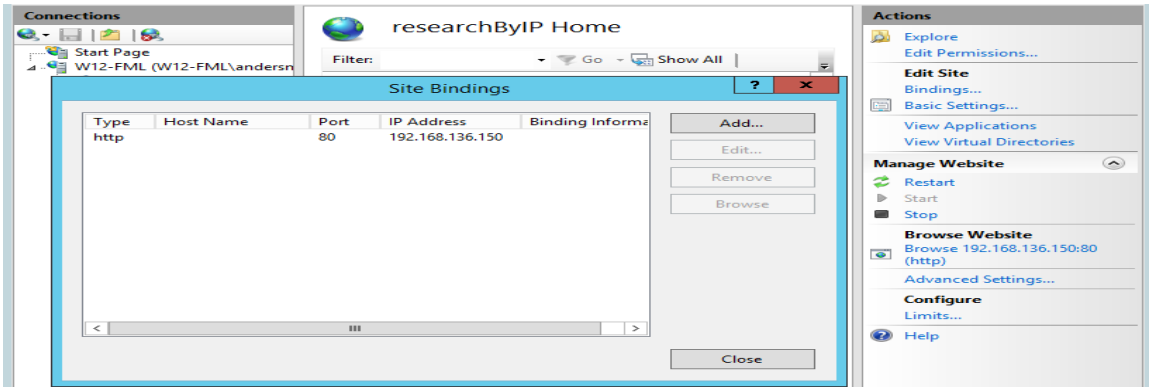
- Double-click **Error Pages** – allows customization of standard HTTP errors. Click on **Default Web Site** to return to the home page.
- Double-click **HTTP Response Headers** – allows customization of HTTP Headers. Click on **Default Web Site** to return to the home page.



b. The 'Content View' tab lists all pages and files within a website. Note that the **hello.htm** file which you created displays here.

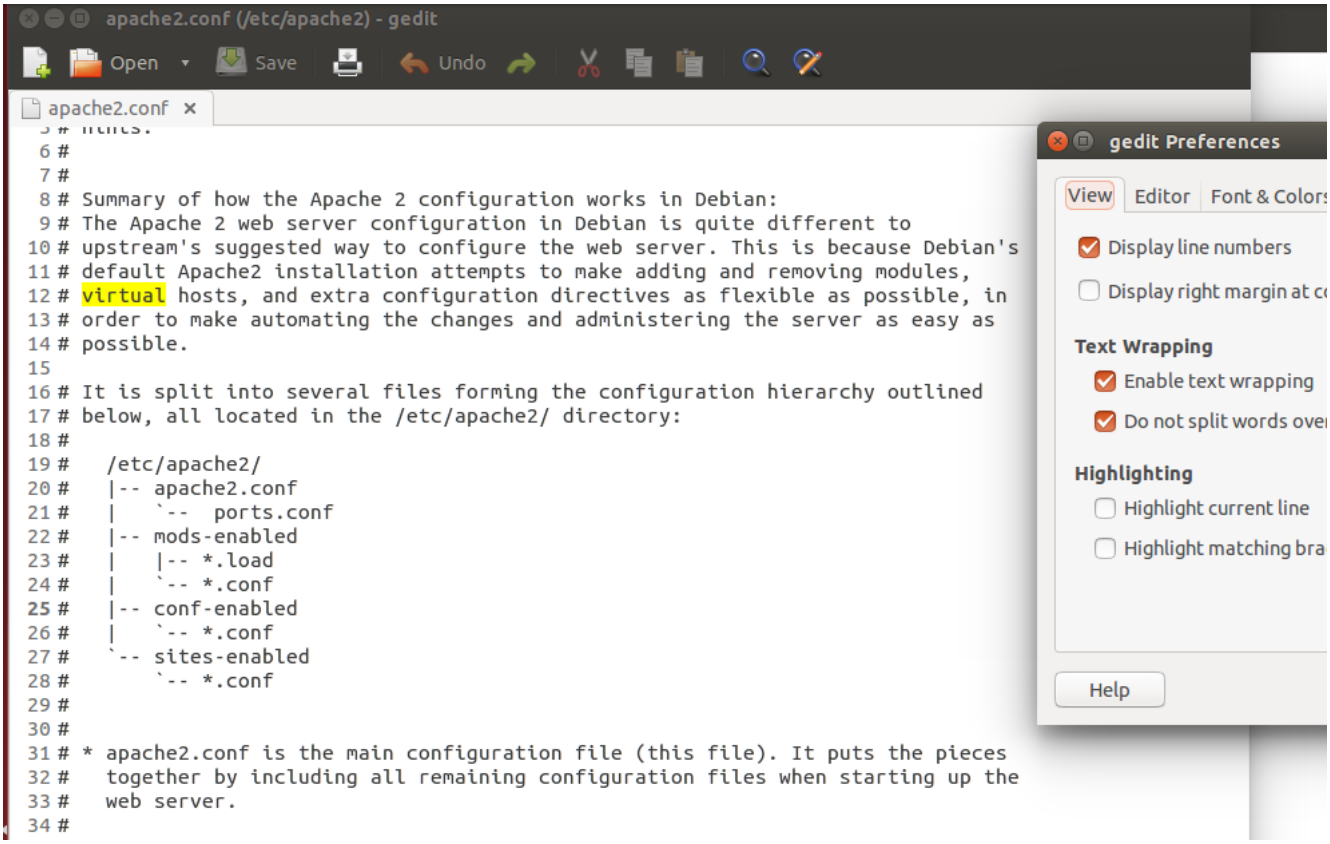


4. The right pane **Actions** is context sensitive.
a. Above is a screen shot of **Actions** when **Content View** and a web page is selected. **Browse** launches the page in Internet Explorer. **Edit Permissions** let us change web page permissions (see figure above).
b. When a web site, such as **Default Web Site** in the **Connections** pane is selected, **Actions** includes **Explore**, **Edit Permissions**, **View Virtual Directories**, **Restart** and **Browse**. **Bindings** reveals the HostName, Port Number and IP address. This allows IIS to identify one of multiple web sites by its HostName, Port Number or IP address.



EXAMINING APACHE PROPERTIES

In Activity 6-2 we introduced the main Apache configuration file /etc/apache2/apache2.conf and the site-specific configuration files /etc/apache2/sites-enabled/*.conf. Apache2.conf only has 250 lines of code. More than 60 percent of the lines in the file are comments (lines that begin with #) that help explain changes or provide examples of changes. Most of the site specific configuration occurs in **include** files. Any file with the extension **.conf** will be included as a site specific configuration file. Let's view the apache2.conf file in gedit (**gksu gedit**). To see line numbers in gedit, select **Edit** > **Preferences** > **Display line numbers**.



Apache Global Environment Settings

You will rarely modify the global settings in the Apache configuration file. Some of these settings focus on physical locations of directories during setup or standard settings common to most servers, such as the port number.

Notice the default **KeepAlive On** in line 92. Recall that the KeepAlive property gives users of your site a persistent connection over a given number of seconds. It allows browsers to download a Web page with images without having to create a new connection for each image.

```
55 # Global configuration
56 #
57
58 #
59 # ServerRoot: The top of the directory tree under which the server's
60 # configuration, error, and log files are kept.
61 #
62 # NOTE!  If you intend to place this on an NFS (or otherwise network)
63 # mounted filesystem then please read the Mutex documentation (available
64 # at <URL:http://httpd.apache.org/docs/2.4/mod/core.html#mutex>);
65 # you will save yourself a lot of trouble.
66 #
67 # Do NOT add a slash at the end of the directory path.
68 #
69 #ServerRoot "/etc/apache2"
70
71 #
72 # The accept serialization lock file MUST BE STORED ON A LOCAL DISK.
73 #
74 Mutex file:${APACHE_LOCK_DIR} default
75
76 #
77 # PidFile: The file in which the server should record its process
78 # identification number when it starts.
79 # This needs to be set in /etc/apache2/envvars
80 #
81 PidFile ${APACHE_PID_FILE}
82
83 #
84 # Timeout: The number of seconds before receives and sends time out.
85 #
86 Timeout 300
87
88 #
89 # KeepAlive: Whether or not to allow persistent connections (more than
90 # one request per connection). Set to "Off" to deactivate.
91 #
92 KeepAlive On
93
94 #
```

```
95 # MaxKeepAliveRequests: The maximum number of requests to allow
96 # during a persistent connection. Set to 0 to allow an unlimited amount.
97 # We recommend you leave this number high, for maximum performance.
98 #
99 MaxKeepAliveRequests 100
100
101 #
102 # KeepAliveTimeout: Number of seconds to wait for the next request from the
103 # same client on the same connection.
104 #
105 KeepAliveTimeout 5
106
107
108 # These need to be set in /etc/apache2/envvars
109 User ${APACHE_RUN_USER}
110 Group ${APACHE_RUN_GROUP}
111
112 #
113 # HostnameLookups: Log the names of clients or just their IP addresses
114 # e.g., www.apache.org (on) or 204.62.129.132 (off).
115 # The default is off because it'd be overall better for the net if people
116 # had to knowingly turn this feature on, since enabling it means that
117 # each client request will result in AT LEAST one lookup request to the
118 # nameserver.
119 #
120 HostnameLookups Off
121
122 # ErrorLog: The location of the error log file.
123 # If you do not specify an ErrorLog directive within a <VirtualHost>
124 # container, error messages relating to that virtual host will be
125 # logged here.  If you *do* define an error logfile for a <VirtualHost>
126 # container, that host's errors will be logged there and not here.
127 #
128 ErrorLog ${APACHE_LOG_DIR}/error.log
129
130 #
131 # LogLevel: Control the severity of messages logged to the error_log.
132 # Available values: trace8, ..., trace1, debug, info, notice, warn,
133 # error, crit, alert, emerg.
134 # It is also possible to configure the log level for particular modules, e.g.
135 # "LogLevel info ssl:warn"
136 #
```

HOSTING MULTIPLE WEB SITES

You can create multiple Web sites on a single server for a variety of purposes. These are known as **virtual hosts**. For multiple Web sites to exist on the same computer, you have to make each site distinct, which you can do by hosting Web sites by port number, by IP address, or based on host name. The following sections describe each method.

Hosting Web Sites by Port Number

One way to host multiple Web sites is to have each Web site listen at a different port. The first Web server you install listens at port 80, so any additional Web site using this feature must listen at a different port. In Chapter 5, you learned that the well-known ports are numbered from 1 to 1023. Because they are reserved, you should not use these port numbers for your additional Web sites. Recall that common choices are 8000 and 8080, because they are unreserved and easy to remember, but you could pick any port above 1023.

When you configure a Web site to use a port other than 80, you must reference the alternate port number in the URL associated with the site because the browser otherwise assumes that a Web site uses port 80. For example, if the host *www.technowidgets.com* is listening at port 8080, and you want to display the document prod.htm, you would type *http://www.technowidgets.com:8080/prod.htm*.

Because entering a URL that specifies a port number is awkward and difficult to remember, creating Web sites by using an alternate port number is the least popular method. When you are creating your Web pages, your links have to reference the alternate port number, too.

The other two methods do not force the person requesting a Web page to alter the format of the URL.

Hosting Web Sites by IP Address

You access a service such as a Web server by using a combination of an IP address and a port number. For example, when someone types *www.technowidgets.com* in a browser, the host name is translated into an IP address. The user does not know or care whether that IP address is one of many on the server or is the only IP address on the server.

As a Web server administrator, you configure two types of IP addresses on a computer. The first type of IP address corresponds to a physical NIC in the computer. The second type is a **virtual IP address**, which is added to the IP address for a NIC.

You use multiple NICs when you are connected to multiple networks. For example, one NIC may be connected through a router to the Internet. Another NIC may be connected to an internal LAN. In this case, you could set up a Web site that corresponds to the IP address connected to the Internet for potential customers of your products. You could also set up a Web site that corresponds to the IP address connected to your LAN for employee information such as department memos and information on benefits. From the Web server's perspective, the Web site for internal use by your organization exists on the intranet, which is a private network.

Virtual IP addresses are often used when you have multiple domain names. For example, when you set up the NIC on your Web server, you gave it an address of 192.168.*.100 (* represents your specific network address). You could then add virtual IP addresses to the same NIC of 192.168.*.150 and 192.168.*.200. When you configure DNS for the new domain, the host IP address for each Web server would be unique, based on each of the virtual IP addresses. However, to support this scheme, you need multiple IP addresses from your ISP. In the past, it was fairly easy to get a block of 254 addresses, but now it is more difficult because the Internet is more popular and fewer IP addresses are available. Virtual IP addresses can be used to reference e-mail servers and FTP servers that exist on the same server with your Web site when your Internet presence is minimal. When the site becomes so popular that you want to split it across three separate physical servers, the IP addresses are already set up and the ensuing conversion is relatively easy.

Hosting Web Sites Based on Host Name

The **easiest and most common way to host Web sites is by host name**. Early in the chapter, you learned that when the browser sends a request for a Web page to a server at a particular IP address, it also sends the host name, such as *www.technowidgets.com*, to the Web server. What if you also wanted another Web site at the same IP address called *www.widgetsofthefuture.com*? Given the way the browser requests a Web page, the Web server can read the host name and redirect the request to the appropriate Web site.

If you configure Web sites by host name, you need only one IP address from your ISP, making it the most economical method. When you configure the DNS, you have to remember just a single address. As

opposed to changing the port number, which also requires only a single IP address, you do not have to change the URL for the Web site.

CONFIGURING NEW WEB SITES IN IIS

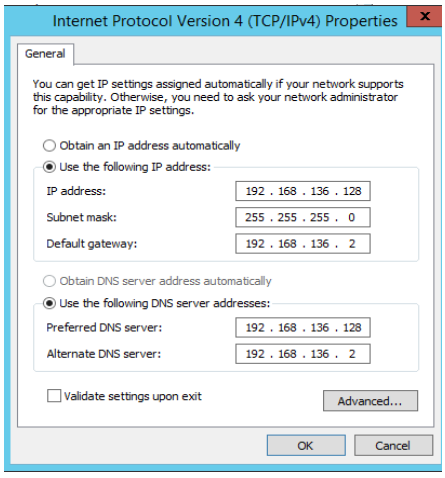
In this section, you will learn how to configure the Web server so that you will have one Web site based on an IP address and a different site based on a host name. You will use the DNS configuration from Chapter 4 to test it. In Chapter 4, you created a DNS configuration based on a domain name such as *technowidgets.com*. In the example, *www.technowidgets.com* referred to 192.168.*.100 and *research.technowidgets.com* referred to 192.168.*.150. One of the new Web sites you will create here is for research. The DNS configuration also has ***your-server-name****.technowidgets.com* at 192.168.0.100, which is the same as *www.technowidgets.com*. The site for ***your-server-name.****technowidgets.com* will be distinguished by your host name. Refer to the **actual IP addresses** that you used in Chapter 4 along with your actual domain name.

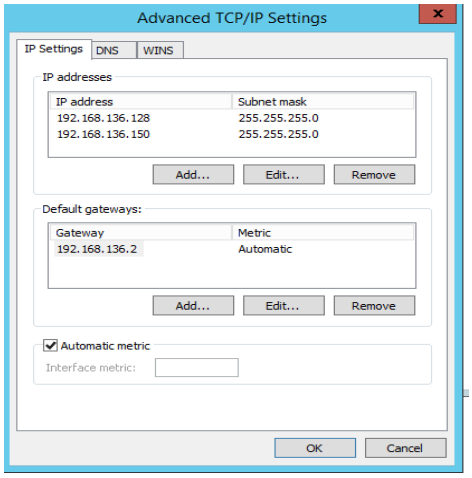
Configuring a Web Site Using an IP Address

You need to complete two procedures to create a Web site using an IP address. First, you must add another IP address to your configuration. Second, you must add the new Web site based on the IP address.

Activity 6.3: Adding IIS Web Sites by IP Address in Windows Server 2012R2

To add an IP address to your NIC in Windows:
Go to your static IP address: **Server Manager > Local Server > Ethernet 0 > right-click Properties > TCP/IPv4 > Properties** and click **Advanced >** under IP Address, click **Add** and enter the *.150 address, click on the **Subnet Mask** field and it will be automatically filled in. Click **Add. Screenshot** the advanced TCP/IP Setting with the additional address. Click **OK > OK > Close** and close the **Network Connections** window to exit.





Before you create the new Web site based on the new IP address for *research.technowidgets.com*, it would be a good idea to test your configuration by pinging the host name at your domain. Run PowerShell and *ping research.technowidgets.com.* You should receive four messages that begin "Reply from 192.168.*.150:". If you do not receive these messages, check your DNS for the exact name and corresponding IP address for research. Then return to the Internet Protocol (TCP/IP) Properties dialog box

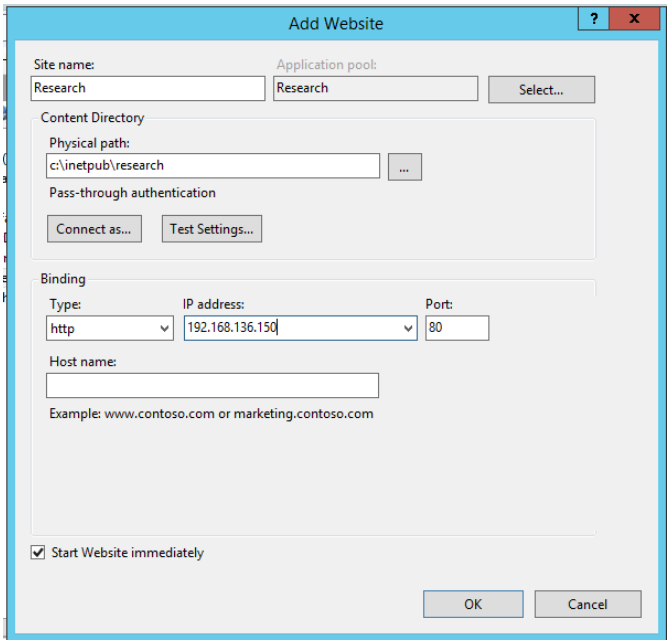
to make sure that the IP address matches the one you configured in DNS.

```
PS C:\Users\andersml> ping research.technowidgets.com

Pinging research.technowidgets.com [192.168.136.150] with 32 bytes of data:
Reply from 192.168.136.150: bytes=32 time<1ms TTL=128
Reply from 192.168.136.150: bytes=32 time<1ms TTL=128
Reply from 192.168.136.150: bytes=32 time<1ms TTL=128
Reply from 192.168.136.150: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.136.150:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
PS C:\Users\andersml> nslookup research.technowidgets.com
Server:  w12-fml.technowidgets.com
Address:  192.168.136.128

Name:    research.technowidgets.com
Address:  192.168.136.150
```

When you are certain the research IP address works properly close PowerShell, and you can create a new web site. To start the Web Site Creation Wizard:

1. In **Server Manager > Tools**, click **Internet Information Services (IIS) Manager**.

Expand your server name, right-click **Sites,** and choose **Add Website** and to open the Add Website dialog box:



2. Enter **Research** in the **Site name:** field and click the ellipsis (…) by the **Physical path:** text box. Browse to **This PC > C: > inetpub** and then click **Make New Folder** and type **research** in the folder name. Click **OK**.
3. To configure IIS to route by IP address, under **Binding** choose your *.150 IP address. Let the **Port** default to 80 and leave the **Host Name** blank. Click **OK** when you are done.
4. Right click the **Powershell** icon on your taskbar and choose **Run as Administrator**.
5. Type **Notepad** and press **ENTER**, then enter the simple html script below (use your name) and click **File > Save As**. Type the filename **\inetpub\research\default.htm** and choose **Save as Type > All Files**. Click **Save** to save the file.

```
<html><body>
This is Tom's research
site </body></html>
```

Important Note: In all exercises below, when you load a web page, you need to **make sure you are not loading a cached page**. You can refresh the page, restart the browser, or remove browsing history to be sure you are loading the latest modified page.

5. Start Internet Explorer and type **research.technowidgets.com** in the Address text box. The default Web page with the "This is <your name>'s research site" text should appear. You could also type the IP address of the new Web site in the browser and open the same page. **Submit a screenshot** of Internet Explorer with the research page displayed.

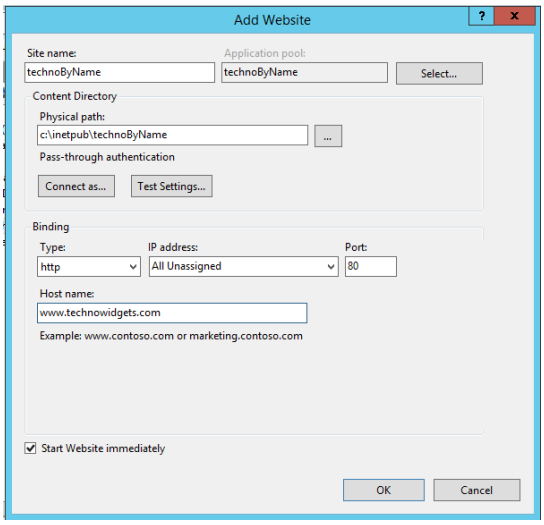
Configuring a Web Site Using a Host Name

Now that you have configured a Web site using an IP address, configuring one with a different host name is even easier. To test the functionality of the new Web site, DNS must be running on your Web server. You need to be able to ping *www.technowidgets.com* (the Web site you set up in Chapter 4) to make sure it is working properly. The important difference between using a host name and using an IP address is that you enter a **host name** instead of an **IP address** in the Web Site Creation Wizard.

Activity 6.4: Adding IIS Web Sites by HostName in Windows Server 2012R2

To start the Web Site Creation Wizard:

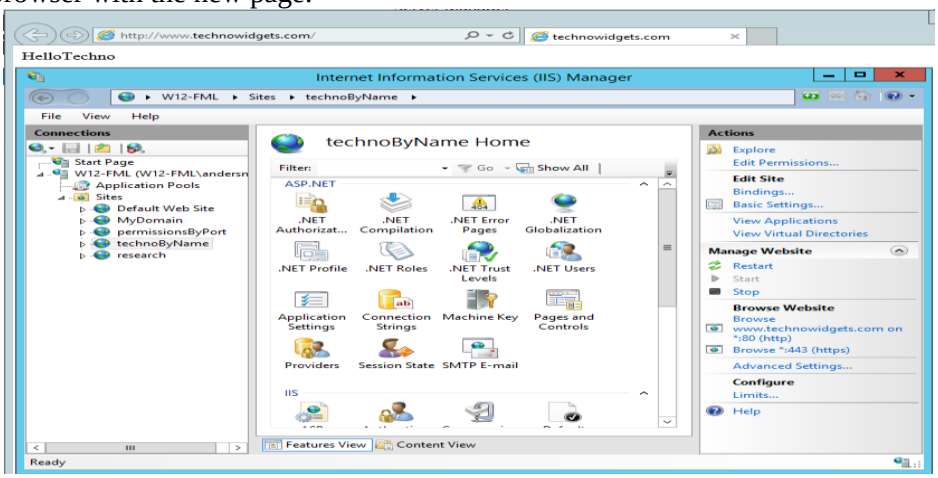
1. In Tools dropdown, click **Internet Information Services Manager**.
2. In the **Connections** pane, right-click **Sites** and choose **Add Website** and to open the Add Website dialog box



3. In the **Site Name:** text box, enter **technoByName**, and click the ellipsis (…) by the **Physical path:** text box. Browse to **This PC > C: > inetpub** and then click **Make New Folder** and type **technoByName** in the folder name. Click **OK**.
4. To configure IIS to route by name, under **Binding**, leave the **IP address** unassigned and let the **Port** default to 80. In the **Host name:** text box, type **www.technowidgets.com**, and click **OK**.
5. Right click the **Powershell** icon on your taskbar and **Run as Administrator**, and click **Yes**.
6. Type **Notepad**, enter the simple html script below (put your name in it) and save it as **\inetpub\technoByName\default.htm** (be sure to use **Save as Type > All Files**). Close Notepad and PowerShell.

```
<html><body>
HelloTechno Tom!
</body></html>
```

7. Open the browser of your choice and type **www.technowidgets.com** in the Address text box. Because we are now associating a different web page with the name of our web site, the default Web page "HelloTechno <your name>" should appear. **Submit a screenshot** of your browser with the new page.



CONFIGURING NEW WEB SITES (VIRTUAL HOSTS) IN APACHE

When you create a new Web site in Apache, you are creating a **virtual host**. You need to create a directory for each new Web site. To make sure that each Web site works correctly, you will create and save a Web page in each Apache directory for testing. Based on information from the DNS server you created in Chapter 4, you will create virtual hosts based on IP addresses and host names.

The Apache configuration file uses tags, which are related to the tags in HTML. For example, in the sample Web page created for the new Web sites in IIS, an HTML tag called <body> tells the browser that what follows is the body of the page. The </body> HTML tag tells the browser that the end of the body has been reached.

In this section, you will create virtual hosts with a tag that looks something like <VirtualHost 192.168.*.150> and end the description of the virtual host with </VirtualHost>. One important concept to remember is that nearly all text is case-sensitive in Apache configuration files. If you accidentally type </Virtualhost>, Apache will not recognize the new Web site because the "H" in the <VirtualHost> tag is not capitalized. Apache does not display a message for capitalization errors when you restart the Web server, so you must enter tags in the configuration file carefully.

Between the opening and closing tags, you can specify many configuration options. In fact, most of the configuration options for the default server can be put between the opening and closing tags. The items you will enter in this section have already been specified for the default site elsewhere in the configuration file. You will begin with the minimum needed to make the Web sites functional. In later chapters, you will add and refine these options.

The first item Apache needs to know is the **host name** of the server. You need to add the key word **ServerName**, followed by a space, and then the host name of the Web site. In the case of the host name **research.technowidgets.com**, you would add the following line:

```
ServerName research.technowidgets.com
```

The other configuration line that you need names the directory where you will store the Web pages. You need to add the key word **DocumentRoot**, followed by a space, and then the location of the Web site. For example, if this directory is **/var/www/research**, you would add the following line:

```
DocumentRoot /var/www/research
```

Creating Directories and Web Pages for Virtual Hosts

Before you can create a virtual host, you must create a directory where you will store the site. You can place Web pages for virtual hosts anywhere in the Linux system. If you have multiple disk drives, you can put the Web pages for each virtual host on separate disk drives. In the following steps, you will create directories in the same directory as the existing Web site for ease of maintenance. In Apache2, the default Web site is found in the **/var/www/html** directory. Another logical place to create the new Web sites is beneath the **/var/www** directory, but you could put them on other places on the server as well. The following steps show how to create a directory called **research** for the IP address-based virtual host, and a directory called **web1** for the name-based virtual host. Recall that the name of the file used for the default Web page in Apache is **index.html**, which is different from the default name used in IIS.

Create the directories for the virtual hosts in Linux:

1. In a terminal window, change to the directory where you are creating the new subdirectories by typing **cd /var/www** and then press **Enter**. Remember to insert a space after **cd**.
2. Type **sudo mkdir research** to create a directory for the research Web site.
3. Type **sudo mkdir technowidgets** to create the directory for the technowidgets Web site.

Create Web pages for the new Web sites:

4. Run gedit from the terminal window as a superuser (or use nano if you prefer)
   gksu gedit
5. Type the following to identify the research site (use your name):
   <html><body>
   Hello Research Tom
   </body></html>
6. Click **File** and then click **Save As** to open the Save File As dialog box.
7. Navigate to **/var/www/research** and save the page as **index.html.**
8. Make an **index.html** page for technowidgets using steps 5 and 6 with web page text **HelloTechno Tom** (use your name) and the folder **/var/www/technowidgets.**
9. Close gedit.

Activity 6.5: Adding Apache Web Sites by IP Address in Ubuntu

Before you can create a virtual host based on another IP address, you must add iface eth0:1 with the other IP address in your **network interfaces** file.

Create another IP address. This example uses the **192.168.136** network. Substitute your own third octet.

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
auto eth0:1
#iface eth0 inet dhcp
iface eth0 inet static
address                 192.168.136.129
netmask                 255.255.255.0
network                 192.168.136.0
broadcast               192.168.136.255
gateway                 192.168.136.2
dns-nameservers         192.168.136.129  8.8.8.8
iface eth0:1 inet static
address                 192.168.136.150
netmask                 255.255.255.0
network                 192.168.136.0
broadcast               192.168.136.255
```

Note that **ifdown eth0**, **ifdown eth0:1** and **ifup eth0**, **ifup eth0:1** can be used to restart the interfaces.

1. Use nano to update interfaces from the terminal window.

sudo nano /etc/network/interfaces

2. The lines you must add are in bold in the above sample. You can add the new lines to your existing file manually or coy and paste the example above into your interfaces file. If you choose to copy and paste, first copy the lines into Notepad and then change the third octet to match yours. The copy from Notepad and paste into your Linux editor. You can use the Paste menu option, or try SHIFT+INS to paste. Save your interfaces file with **CTRL+X > Y > ENTER**.

3. To tell Ubunto to recognize your new interface, type **sudo ifup eth0:1**. Ping your new IP address to prove that it's functional. If necessary, restart networking or reboot Ubuntu. **Submit a screenshot** of your successful ping of the *.150 address.

Create a virtual host configuration by IP address in an include.conf file in /etc/apache2/sites-enabled/

4. Use the editor of your choice to make the file: **/etc/apache2/sites-enabled/technowidgets.conf** with the conten in the example below. Be sure to copy to Notepad and change the third octet before pasting. Note that you could name this file anything as long as it has the .conf extension.

```
# by IP
<VirtualHost 192.168.136.150>
    ServerName    research.technowidgets.com
    DocumentRoot /var/www/research
</VirtualHost>
```

5. # is used for comment lines. The first line is a comment. These **VirtualHost** tags instruct Apache to route by IP address. **ServerName** is the host name for the site and **DocumentRoot** is the directory where web pages are stored.

6. We have already created a research default web page named **/var/www/research/index.html**, so we are ready to restart Apache:
sudo service apache2 restart
8. Open the Firefox web browser, type **research.technowidgets.com** for the URL, and then press **Enter**. You should see the Research web page you created. **Submit a screenshot**.

Activity 6.6: Adding Apache Web Sites by HostName in Ubuntu

Create a virtual host configuration by Hostname in **/etc/apache2/sites-enabled/technowidgets.conf**:
1. Use the editor of your choice to add the bold lines to **technowidgets.conf** as follows:

```
# by IP
<VirtualHost 192.168.136.150>
      ServerName    research.technowidgets.com
      DocumentRoot /var/www/research
</VirtualHost>

# by Name
<VirtualHost *:80>
      ServerName    www.technowidgets.com
      DocumentRoot /var/www/technoByName
</VirtualHost>
```

2. The second VirtualHost tag instructs Apache how to route any **port 80** traffic with hostname **www.technowidgets.com** by IP address. Notice the line **DocumentRoot /var/www/technoByName** specifies the location of the web page to load.
3. Create the technoByName directory with **sudo mkdir /var/www/technoByName**
3. Create a Technowidgets default web page **/var/www/technoByName/index.html** in the document root with this content (use your name)

```
<html><body>
Hello technoByName Tom!
</body></html>
```

4. Restart Apache by by restarting or reloading
sudo service apache2 restart
5. Open the Firefox web browser, type **www.technowidgets.com** for the URL, and then press **Enter**. You should see the technoByName web page you created. **Submit a screenshot** of the page.

UNDERSTANDING VIRTUAL DIRECTORIES

A **virtual directory** is a Web directory that is not physically located beneath the Web root. For example, a URL of *www.technowidgets.com/prod* does not necessarily mean that a directory called prod is below the default document root **C:\inetpub\wwwroot** in Windows or **/var/www/html** in Ubuntu. .

Instead, the directory prod could be one of the following locations:

- Another directory on the computer such as \categories\products
- A directory located on another computer
- A URL on another Web server

Allowing the Web directory to refer to another physical directory on the computer could make it easier to organize the Web site based on directory permissions. For example, the marketing manager could be in charge of updating product information. In Linux, that manager could log on and have the directory that corresponds to the prod virtual directory be her default home directory, which would facilitate updating the pages. The physical directory for the root of the Web server would also be less cluttered with directories. On the other hand, creating a virtual directory in this case would make it more difficult to keep track of where the Web pages for the site actually reside.

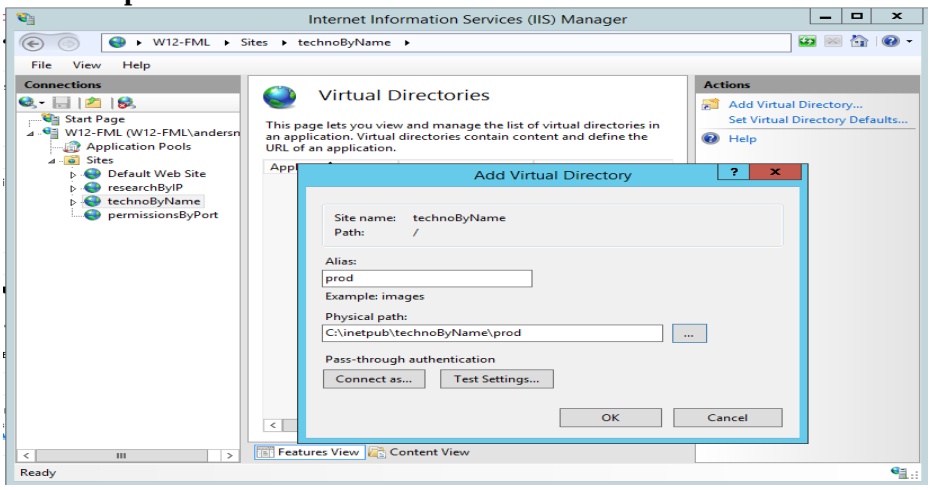
When you use a directory that is physically stored on another computer, you are typically connected by a LAN. The marketing manager updating the product Web pages may have access to another computer on the LAN. The Web server can use the directory on the other computer through a share in the Windows environment or NFS in a Linux environment.

The ability to have a directory actually be a URL on another Web server is a useful option. It allows you to balance the load of the Web server by letting other Web servers process certain Web pages. Suppose you used the prod directory to hold pages containing thousands of images. By having the pages reside on another Web server, you could dedicate a whole Web server to retrieving large image files. Virtual directories can also be used to organize corporate Web sites. For example, your company may have Web sites in various regions around the world. You can make them appear to exist on a single server by creating directories that correspond to your regional Web servers.

Activity 6.7: Creating an IIS Virtual Directory In Windows Server 2012R2

To create a virtual directory in IIS:

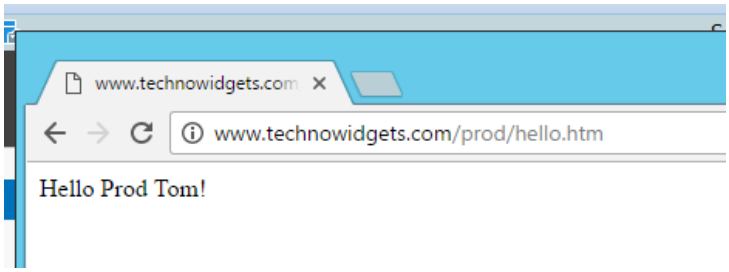
1. Open **Internet Information Services (IIS) Manager**.
2. In the **Connections** pane expand your server and **Sites**, and select the **technoByName site**.
3. Right-click and choose **Add Virtual Directory**.
4. Complete the Add Virtual Directory dialog box below. You must click the ellipsis (…) and browse to **C: > inetpub > technoByName** folder, then click the **Make New Folder** button and create the **prod** folder.



5. The Alias **prod** is used in the URL to access the Virtual Directory (VD) as follows: www.technowidgets.com/prod/ The Physical path or location for your VD web pages is **C:\inetpub\technoByName\prod**.
6. Build a test web site in **C:\inetpub\technoByName\prod\hello.htm** using the editor of your choice with the text below (use your name).

```
<html><body>
Hello Prod Tom!
</body></html>
```

7. Open Chrome or Internet Explorer and type **www.technowidgets.com/prod/hello.htm** in the URL. You should see the Virtual Directory web page. **Submit a screenshot** of the page.



Configuring Virtual Directories in Apache

As you have already seen, you need to modify a **/etc/apache2/sites-enabled/technowidgets.conf** file to configure this Web site that you are administering on an Apache Web server. As in IIS, a virtual directory in Apache has two components. First you associate the physical location where the Web pages will reside with the name of the directory as the browser would reference it. Second you would configure the directives for the virtual directory. Directives are used to secure the directory. For example, if the sales department stored their web pages in the sales directory under the /var/www/ and you wanted to address this virtual directory *http://www.technowidgets.com/sales/*, you would first need an Alias command:

```
Alias /sales/  "/var/www/sales/"
```

This code states that when /sales/ is detected in a URL, the request should be sent to the /var/www/sales directory. Second you define the virtual directory's directives.

```
<Directory "/var/www/sales/">
        Require all granted
</Directory>
```

- The **Require** directive specifies which users can access a resource. In the example above, **Require all granted** allows everyone access to the directory.
- The **Allow** directive was used for this purpose prior to Apache 2.4. Ex: **Allow from all** and the **Order** directive was used to control the order in which permissions were applied. Ex: **Order allow,deny**

See http://httpd.apache.org/docs/2.2/mod/core.html#directory for more information on this directive.

The .htaccess files provide a way to make configuration changes on a per-directory basis. They are used when content providers do not have root access. The use of .htaccess files can be enabled or disabled with the **Allowoverride** directive: **Allowoverride All** or **Allowoverride None**

See http://httpd.apache.org/docs/2.2/howto/htaccess.html for more information on htaccess files.

Virtual directories in IIS are different from virtual directories in Apache. In IIS, a virtual directory is always associated with an individual Web site. For example, earlier in this chapter, you created a virtual

directory called prod that you associated with the *www.technowidgets.com* Web site. That virtual directory can be used only with *www.technowidgets.com*, but no other Web site. In Apache, the sales virtual directory can be used by all of the virtual hosts, because it was defined in the main server configuration area and not within any of the virtual hosts. As a contrast, you will create the prod virtual directory so it can be used only from the *www.technowidgets.com* Web site in Apache. The physical location of the prod virtual directory will be /var/www/prod. Also note that you can the define properties for a virtual host in the same fashion that you defined properties for a virtual directory.

Activity 6.8: Creating an Apache Virtual Directory in Ubuntu 14

To create a virtual directory called prod on the Apache Web server:

1. Open a terminal window and change to the directory where you are creating new subdirectories. Type **cd /var/www** and then press **Enter**.
2. Type **sudo mkdir prod** and then press **Enter** to create the sub directory for the prod virtual directory.

To create Web pages for the new Web sites:

3. Using the editor of your choice, create **/var/www/prod/index.html** as follows:

```
<html><body>
Hello Prod Tom!
</body></html>
```

To add the virtual directory to the virtual host www.technowidgets.com in /etc/apache2/sites-enabled/technowidgets.conf:

4. Using the editor of your choice open /etc/apache2/sites-enabled/technowidgets.conf.
5. Add the virtual directory information, which is shown in bold in the following text. Take care to insert the lines **before** the closing </VirtualHost> tag.

```
# by Name
<VirtualHost *:80>
        ServerName      www.technowidgets.com
        DocumentRoot /var/www/technowidgets

        Alias /prod/ "/var/www/prod/"
            <Directory   "/var/www/prod/">
              Require all granted
            </Directory>

</VirtualHost>
```

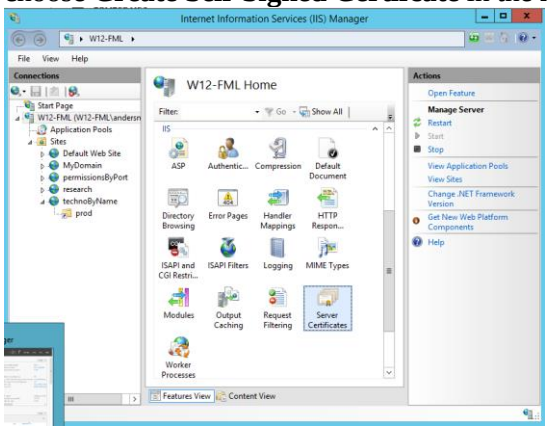
6. Restart your Apache server with **sudo service apache2** restart.
7. Now you can test your virtual directory by opening the Firefox browser and typing the URL **www.technowidgets.com/prod/**. Clear your browser history to be sure you get the latest page! Because you did not type the name of the Web page, Apache sends the default Web page of **index.html**, which you created earlier with the text of "Hi Prod <your name>" **Submit a screenshot** of this page.

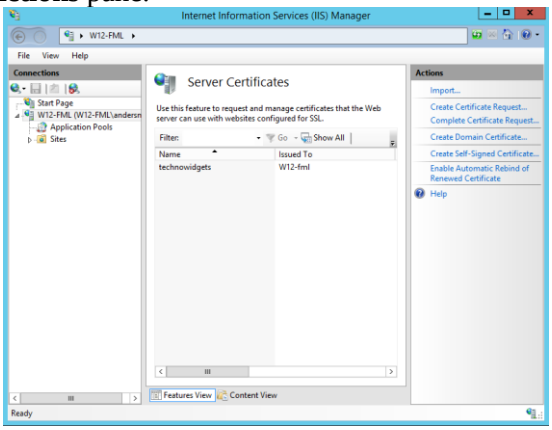
Activity 6-9: HTTPS - Adding an SSL Certificate in IIS for Server 2012R2

The **Hypertext Transfer Protocol Secure** (**HTTPS**) is a communications protocol for secure communication over the Internet. It layers the Hypertext Transfer Protocol (HTTP) on top of the SSL/TLS protocol. The security of HTTPS is TLS which uses long-term public and secret keys to exchange a short term session key to encrypt the data flow between client and server. For more information, see: https://en.wikipedia.org/wiki/HTTP_Secure

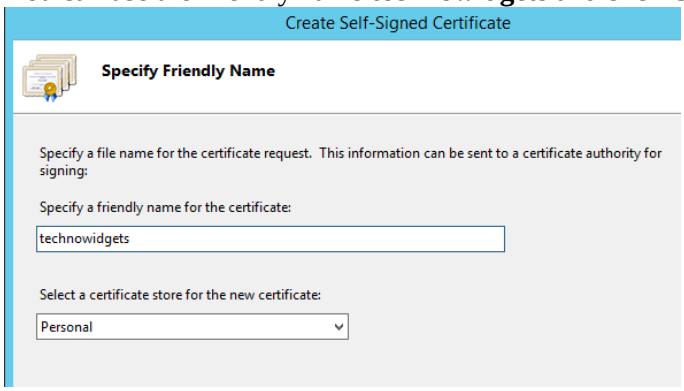
IIS lets you create temporary certificates known as self-signed certificates to temporarily setup HTTPS to encrypt communication between your server site and a browser. A certificate from a Certificate Authority is needed for a permanent setup.

1. From **Server Manager > Tools > IIS Manager**, click on your server name and click the **Server Certificates** feature in the midle pane. Click **Open Feature** in the **Actions** pane and choose **Create Self-Signed Certifcate** in the **Actions** pane.

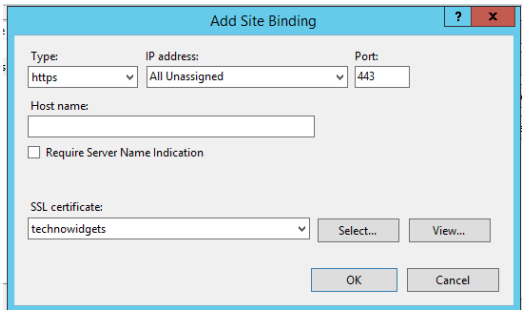




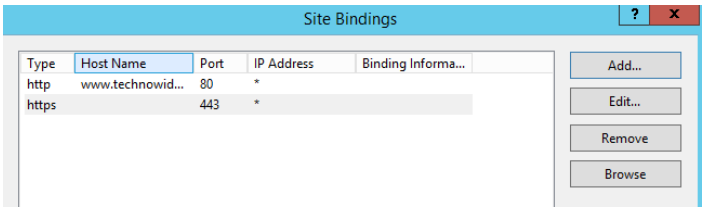
2. You can use the friendly name **technowidgets** and click **OK**.



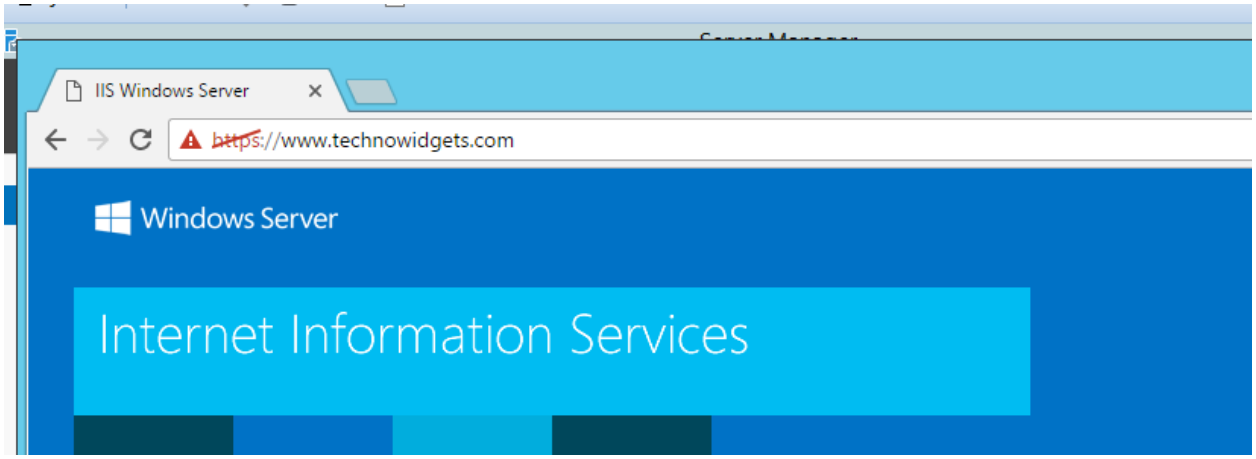
3. Next click on your **technoByName** Web site in the left pane, click on **Bindings** in the **Action** pane, and click **Add** to create an **HTTPS** binding.
4. Make these changes to the page: **Type: https** and **SSL Certificate: technowidgets**. Leave the other options at the default, as in the image below. Click **OK**.



5. When you're finished, your bindings should look like this. Click **Close**.



6. Try **HTTPS://**www.technowidgets.com in I.E. A browser expects a current certificate from a valid Certificate Authority. To acknowledge that this is a self-signed certificates, clic **Advanced**, and then **proceed to www.technowidgets.com**.



7. **Submit a screenshot** of your HTTPS page. Be sure to include your name in the screenshot.

CHAPTER SUMMARY

- The primary purpose of a Web server is to produce pages of HTML. The protocol used to communicate between the Web server and the browser is HTTP. The two most popular Web servers are Apache from the Apache Software Foundation and Internet Information Services (IIS) from Microsoft. Web servers typically listen at port 80, but you can modify this setting to have them listen at any port higher than 1024.
- IIS Web Server is installed as a Server Role in Windows. Apache is installed as part of the LAMP Server which also includes MySQL and PHP.

❒ IIS properties can be configured globally so that all new sites will share the new configuration. You can also change the properties for individual sites. Most IIS properties are related to the location of files, characteristics of files, and security settings. You can view and change them using a GUI.
❒ To configure an Apache Web server with local settings, you create file(s) in the /etc/apache2/sites-enabled/*.conf folder. The main configuration file apache2.conf includes this folder reference to local settings from Apache's global settings.
❒ You can create more than one Web site on a single server. For multiple Web sites to exist on the same computer, you have to make each one distinct. You can create a distinct Web site by configuring it with a different port number, a different IP address, or a different host name.
❒ In IIS, you use a wizard to create a Web site. You need to configure the folder where you will create the root of the Web site. One wizard dialog box allows you to make the Web site distinct from the other Web sites that you have configured previously. You also have to determine the basic access permissions and file types that you will allow on the site.
❒ New Web sites in Apache are called virtual hosts. You configure the virtual hosts in /etc/apache2/sites-enabled/*.conf using tags similar to those found in XML. When defining a virtual host, you may configure the IP address, the host name and/or the port and include the location of the documentroot.
❒ Virtual directories are directories that appear to be located beneath the root of the Web server, yet are actually located elsewhere. A virtual directory can correspond to another directory on the same computer as the Web server, a directory located on another computer, or a URL on another Web server.
❒ HTTPS together with an SSL Certificate is used to encrypt communication between a Web Browser and Web Server

- Go to D2L **Discussions** to respond to Ch 6 Web Server Configuration and **Quizzes** to complete the Ch 6 Review Questions. You have three attempts at the review questions and your score is the average of all three. Submit your completed worksheet to the Chapter 6 **Assignments** folder.