PricipalPermission.Demand vs IsInRole

The Demand() call throws an exception, which would propogate up the call stack automatically if not caught.

IsInRole() call is a comparison test, which can be used to perform some either-or code.

Are they the same: No, one throws an exception, the other returns a true/false value.

When should I use which: Use Demand() if you want to force-terminate processing if you do not have the privilege, use IsInRole() to perform conditional processing.

**When to use Forms Authentication?**  
Forms authentication is used for internet web applications. The advantage of Forms authentication is that users do not have to be member of a domain-based network to have access to your application. Many internet web sites like Gmail.com, Amazon.com, facebook.com etc uses forms authentication. To access these applications we do not have to be member of their domain-based network like with Windows Authentication.

**How do we enable forms authentication?**

To enable forms authentication, set authentication element's mode attribute to forms in web.config file of the application.

**Advantages of using HTTPS**  
HTTP stands for Hyper Text Transfer Protocol. HTTPS, stands for Hyper Text Transfer Protocol Secure. As the name suggests, HTTPS is more secure than HTTP. When the web server and the client communicate, using HTTP, protocol, the messages that are exchanged over the internet are not encrypted. Any one can secretly listen and see the messages that are exchanged between the client and the web server. That's why, any sensitive information like passwords, financial transactions should never be done over HTTP protocol. Most of the banking applications use HTTPS protocol. Messages exchanged between the client and web server, using the HTTPS protocol are encrypted and are very secure. HTTP use port 80 and HTTPS use port 443.

**How to configure HTTPS instead of HTTP for asp.net web applications**

The configuration to use HTTPS, is usually done in IIS. The encryption and decryption of  messages exchanged between the client and the server is done by server certificates. These server certificates needs to be installed on the IIS server.

**What is Secure Socket Layer and how is it different from HTTPS**  
HTTPS is HTTP (HyperText Transfer Protocol) plus SSL (Secure Socket Layer). SSL standing for Secure Sockets Layer (SSL) is a standard security technology for establishing an encrypted link between a web server and a browser, so that the data sent over the Internet can’t be read by others. When a user requests a secure Web page, the server generates an encryption key for the user’s session and then encrypts the page’s data before sending a response. On the client side, the browser uses that same encryption key to decrypt the requested Web page and to encrypt new requests sent from that page. SSL uses server certificates for encryption and decryption. An SSL certificate contains a public key and certificate issuer. Not only can clients use the certificate to communicate with a server, clients can verify that the certificate was cryptographically signed by an official Certificate Authority.

HTTPS is a transport protocol that uses the SSL protocol to secure its messages.

**What are self signed certificates**  
A self-signed certificate is an identity certificate that is signed by its own creator. Certificates are signed by Certificate Authority. In general self signed certificates are fine for testing purposes and not for production use.

**What are the step for creating a self-signed certificate in IIS7?**

In IIS 7.5  
**1.** Click on the "Server Name"  
**2.** Double click "Server Certificates" feature  
**3.** Click on "Create Self Signed Certificate" link, under "Actions"  
**4.** Specify a friendly name for the certificate and click OK. The friendly name is not part of the certificate itself, but is used by the server administrator to easily distinguish the certificate.

**How do you Associate an asp.net web application with a specific certificate**  
Add HTTPS site binding, if it is not already present  
1. Open IIS  
2. Expand the "Server Name"  
3. Expand "Sites"  
4. Select "Default Web Site"  
5. Click "Binding" under "Edit Site" in "Actions" pane.  
6. In the "Site Bindings" window, Click "Add"  
7. Select Type = "https" and the SSL Certificate and click "OK"  
8. Click "Close" on "Site Bindings" window

**If you want to dis-allow, access over HTTP protocol there are 2 ways. What are they?**  
**First Way:** Remove HTTP binding at the IIS Server level. This option will prevent all the web applications, running on that server to use only HTTPS binding.   
  
**Second Way:** Let both the bindings be available at the server level and configure SSL settings at an application or web site level.   
1. Select your web application in IIS  
2. Double click "SSL Settings" from the features window  
3. Make sure "Require SSL" check box is checked.  
4. Click "Apply" under "Actions" pane