Seguridad HTTP y servicios Web de ASP.NET

Introducción

Uno de los temas que parece estar volviendo locos a los desarrolladores de servicios Web es el de averiguar el modo en que interactúan los servicios Web IIS y ASP.NET para ofrecer seguridad. Hoy por hoy la seguridad es una cuestión que gestiona IIS y que aprovecha ASP.NET. ASP.NET puede obtener la información de identidad que proporciona IIS y utilizarla para averiguar quién realizó la llamada o para hacer uso de la seguridad de acceso al código para determinadas operaciones en el servicio Web. La parte más difícil para muchos es habilitar la aplicación .NET para aprovechar las ventajas de las características de seguridad integradas en IIS. En un futuro no muy lejano, WS-Security será una opción aún mejor. Mientras tanto, la seguridad a nivel HTTP será la opción que muchos de nosotros adoptemos para garantizar la seguridad en mensajería.

Para ejecutar un método Web de forma segura, se deben tener en cuenta los siguientes factores:

- **Confidencialidad:** los datos se vuelven opacos para aquellas entidades que escuchen la conversación.
- **Integridad:** permite a los receptores detectar cambios en el mensaje SOAP.
- **Autenticación:** responde a la pregunta "¿quién realiza la llamada?"
- **Autorización:** responde a la pregunta "¿tiene el llamador derechos de acceso a este método Web?"
- **No repudio:** demuestra que se produjo una acción en particular para evitar que el cliente reniegue de manera fraudulenta de una transacción.

Muchas de estas funciones de seguridad están interconectadas. La autenticación permite que tengan lugar la autorización y el no repudio. Las medidas de confidencialidad que ofrece SSL incluyen también mecanismos para la integridad y la autenticación. En este artículo se presupone que el lector ya está familiarizado con el modo en que se utiliza SSL con IIS. De no estarlo, consulte los recursos que se ofrecen al final de este artículo. Me gustaría igualmente recomendar al lector que localice un servidor Microsoft® Windows® Server que tenga instalado Certificate Server (o que instale Certificate Server en un equipo disponible con Windows Server). Será de gran ayuda para seguir las secciones orientadas a SSL de este artículo.

Cifrado y firma con SSL

Siempre que necesite mantener la confidencialidad de la información en un mensaje de SOAP basado en HTTP, deberá ejecutar el servicio a través de SSL para que los datos del servicio Web se mantengan ocultos para cualquier entidad que intente tener acceso a la transmisión de datos durante la conexión.

Para abrir la consola de administración de Servicios de Internet Information Server (IIS)

1. Haga clic en **Ejecutar** en el menú **Inicio**.
2. Escriba **inetmgr** en el cuadro de edición **Abrir**.
3. Haga clic en **Aceptar**.

Se abrirá la consola de administración de IIS.

Solicitar SSL para el directorio virtual o para un archivo en particular dependerá de la selección de las opciones correctas en IIS. Para seleccionar las "opciones correctas", explore el directorio virtual en la consola de administración de IIS. Si desea solicitar SSL para todos los servicios Web accesibles a través de un directorio virtual dado, haga clic con el botón secundario del mouse en el directorio virtual, después haga clic en **Propiedades** y, a continuación, haga clic en la ficha **Seguridad de directorios**.

Para proteger únicamente un servicio Web específico, haga clic en el archivo .asmx asociado a dicho servicio Web con el botón secundario del mouse, después haga clic en **Propiedades** y, a continuación, haga clic en la ficha **Seguridad de archivo**. Para ambos procedimientos, verá un cuadro de diálogo similar al de la figura 1. En **Comunicaciones seguras**, haga clic en **Modificar**. Se abrirá el cuadro de diálogo **Secure Communications** tal y como se muestra en la figura 2.

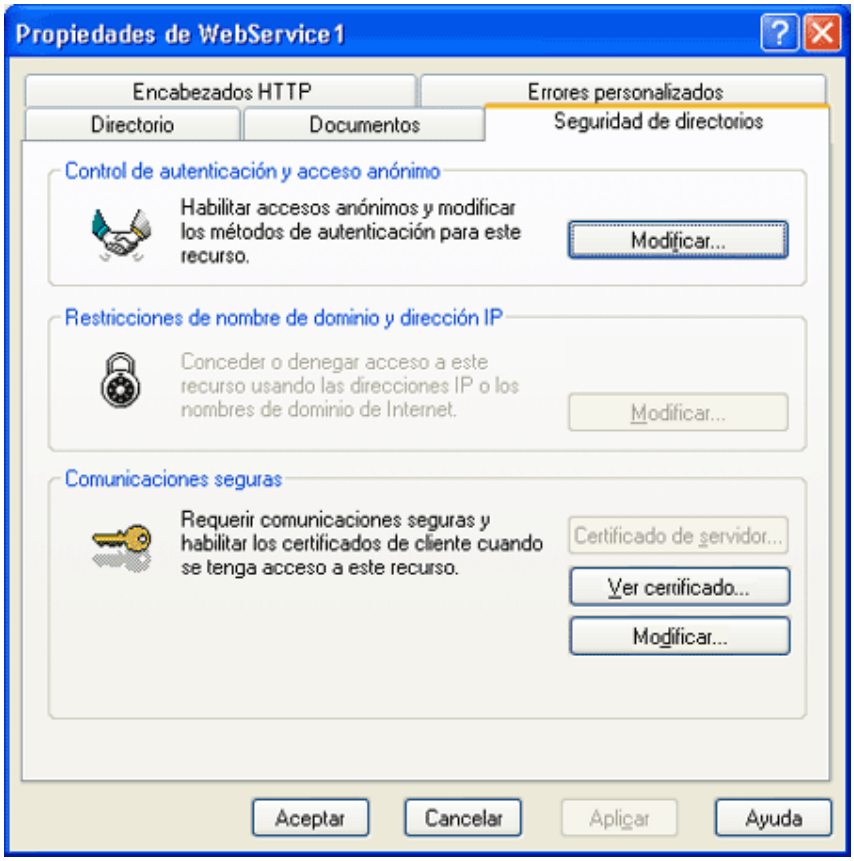

Figura 1. La ficha Seguridad de la consola de administración de IIS

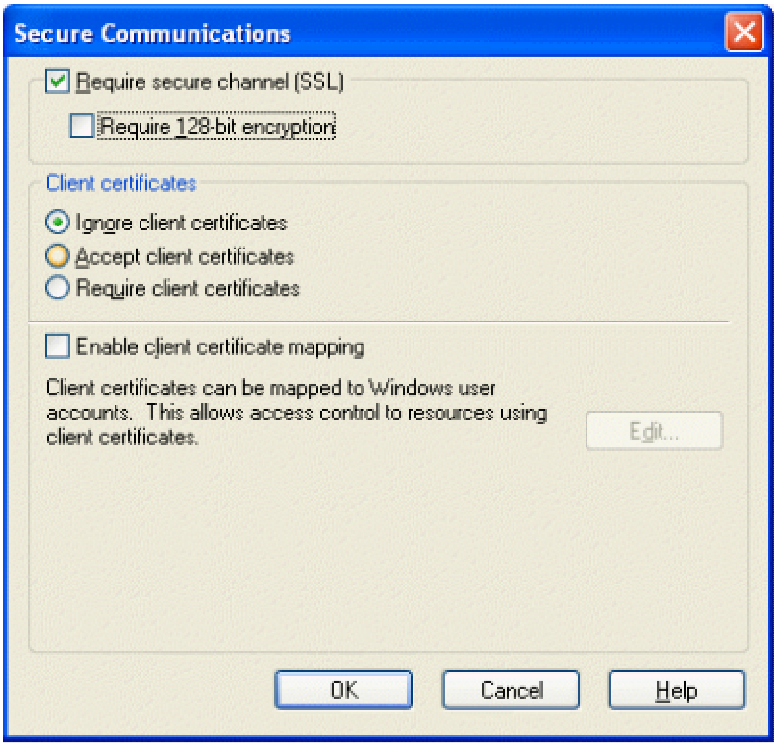

Figura 2. Cuadro de diálogo Secure Communications

De forma predeterminada, la casilla de verificación **Require secure channel (SSL)** no está seleccionada; selecciónela para requerir SSL. SSL admite tanto cifrado de 40 bits como de 128 bits. Cuantos más bits utilice el cifrado, más difícil será romperlo y adivinar después los bits que había originalmente. Esto es todo lo que necesita hacer para activar SSL para un archivo .asmx determinado o para todo un servicio Web. Con ello, la comunicación entre cualquier cliente de servicios Web y el servicio Web en sí estará protegida siempre y cuando no se descubra el certificado del servidor Web. SSL utiliza certificados X.509 que contienen una clave pública y que pueden también contener una clave privada. Si otras personas descubren la clave privada, las comunicaciones cifradas que utilizan la clave pública ya no serán seguras si dichas personas las inspeccionan.

Una vez que tenga los recursos configurados para requerir SSL para comunicaciones, los mensajes entre el emisor y el receptor se cifrarán y firmarán, lo que implica que el contenido de los mensajes no podrá ser leído por terceros. Si estas terceras personas modifican los bytes del mensaje, el receptor del mensaje podrá detectarlo.

Autenticación

Para hacer uso de toda autenticación que IIS haga por usted, necesitará editar el archivo Web.config asociado con su servicio Web. Para poder disponer de la identidad del usuario en el **HttpContext**, será necesario definir el atributo /configuration/system.web/authentication/@mode en **Windows**. El atributo de modo debe

definirse cuando IIS utilice alguna de las opciones de autenticación siguientes: básica, implícita, autenticación de Windows integrada (NTLM/Kerberos) o certificados X.509. Las credenciales de usuario que presenten cualquiera de las anteriores opciones de autenticación deberán disponer de un usuario asignado que exista bien en el equipo local o en Active Directory.

La combinación de IIS con los parámetros correctos del archivo Web.config permitirá que el servicio Web averigüe la identidad del llamador. Y una ventaja de esto es que el contexto de solicitud asumirá la identidad del llamador. Si decide utilizar la autenticación de Windows, el archivo Web.config tendrá que presentar este aspecto:

```
<configuration>
    <system.web>
        <authentication mode="Windows" />
        <!-- aquí van otros elementos -->
    </system.web>
</configuration>
```

Para la gestión de la autorización, auditoria y no repudio, es de suma importancia que se active la autenticación de Windows. Con ello se permite que el método Web se ejecute como llamador. Todo inicio de sesión, comprobación de acceso, etc., se efectuará en función de los permisos del usuario.

Para forzar que IIS proporcione la identidad del llamador, necesitará indicarle que desactive el acceso anónimo. Y eso es todo a decir verdad. Para ello, vuelva y abra **inetmgr** (haga clic en **Iniciar**, después en **Ejecutar** y, por último, escriba **inetmgr**). Desplácese hasta el directorio virtual de su interés. Dependiendo de si desea exigir la identidad de todos los archivos del directorio virtual o de un servicio Web solamente, haga clic con el botón secundario del mouse en el directorio o archivo .asmx y, a continuación, haga clic en **Propiedades**. Haga clic en la ficha **Seguridad de directorios** como se muestra en la figura 1. En **Control de autenticación y acceso anónimo**, haga clic en **Modificar**. Se abrirá el cuadro de diálogo **Métodos de autenticación** tal y como se muestra en la figura 3.

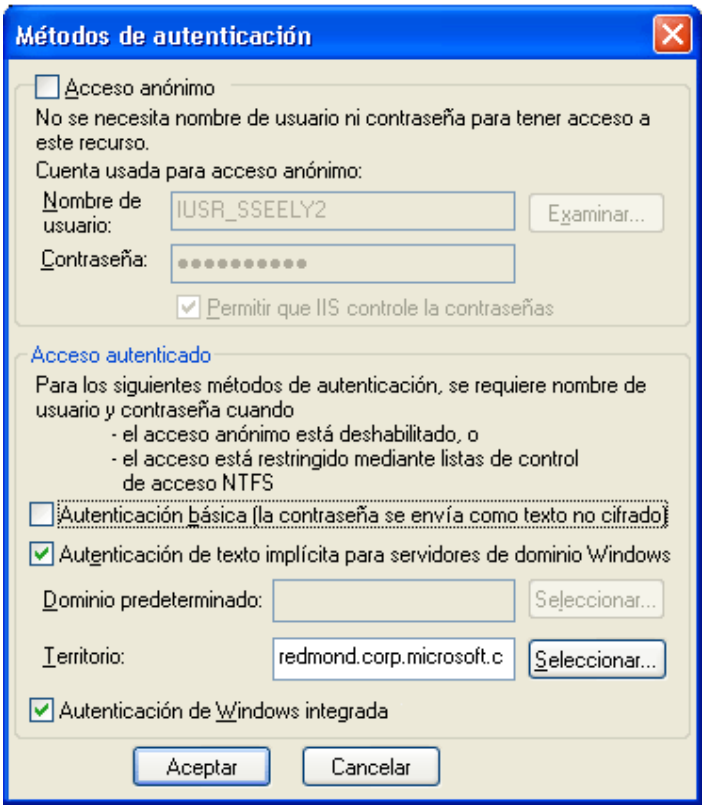

Figura 3. Cuadro de diálogo Métodos de autenticación con Acceso anónimo desactivado

El cuadro de diálogo **Métodos de autenticación** permite configurar el modo de acceso de los usuarios al archivo o directorio virtual. Para pasar las credenciales del usuario por medio de los encabezados HTTP, utilice la autenticación básica o implícita. Ni la autenticación básica ni la implícita ofrecen mecanismos para proteger el mensaje. El mecanismo para pasar las credenciales de usuario se define en RFC 2617: HTTP Authentication: Basic and Digest Access Authentication (en inglés). Básicamente, para pasar el nombre y la contraseña se utiliza un encabezado HTTP denominado **Authorization**. En el caso de autenticación básica la combinación nombre-contraseña se envía en texto sin cifrar. Bueno, no exactamente. El nombre y la contraseña se envían en realidad utilizando la codificación de Base64, que es simplemente texto sin cifrar. Para los que no estén familiarizados con la codificación de Base64, es un modo de tomar datos binarios y presentarlos como texto. Al codificar los datos no hacen falta más secretos ni claves. Si decide utilizar la autenticación básica sólo debe aceptar esas credenciales a través de SSL. Con ello se protegerá al servicio Web y al llamador de cualquier entidad que merodee por el canal con intención de capturar un conjunto de credenciales válidas.

Otra opción sería usar la autenticación implícita. Si se decide por esta opción, es importante que tenga en cuenta que muchos kits de herramientas de SOAP no son compatibles con la autenticación implícita. Por consiguiente, está limitando el número de kits de herramientas que puede utilizar el servicio Web. Utilice la autenticación implícita cuando desee saber la identidad del llamador, los kits de herramientas SOAP ofrezcan compatibilidad con autenticación implícita y el contenido de los mensajes SOAP no sean de especial

importancia. La autenticación implícita cifra las credenciales del llamador por medio de un secreto compartido denominado nonce.

Tanto la autenticación implícita como la básica utilizan mecanismos de desafío-respuesta. Por ello, antes de que se produzca una invocación de métodos Web, se intercambiarán varias solicitudes y respuestas entre el cliente y el receptor. En la autenticación básica, el desafío y la respuesta pueden producirse con bastante rapidez. De hecho, un cliente puede proporcionar credenciales básicas con antelación si sabe que se requiere autenticación básica. Esta aceleración puede ser mínima en conexiones basadas en SSL en las que se necesita verificar el certificado del servidor y establecer una clave de sesión. En la autenticación implícita, deberá intercambiarse el secreto compartido nonce antes de que se cifren las credenciales. De nuevo será necesario que se intercambien algunos protocolos de enlace antes de que se ejecute cualquier código del servicio Web.

Para forzar estos elementos de modo que se activen para el servicio Web, seleccione los cuadros apropiados del diálogo **Métodos de autenticación**. Si desea asegurarse de que únicamente obtiene usuarios autenticados, asegúrese de que la casilla de verificación **Acceso anónimo** no esté seleccionada. Una vez realizado, podrá llevar a cabo los siguientes pasos en el servidor:

- Averiguar quién es el llamador.
- Utilizar seguridad de acceso al código para limitar quién llama a qué métodos.

El siguiente servicio Web devuelve la información del llamador actual:

```
[WebMethod]
public string WhoAmI() {
    return "Ejecutándose como usuario: " +
        Thread.CurrentPrincipal.Identity.Name;
}
```

Modificaremos una sencilla aplicación de la consola que llama a este servicio Web. Para comenzar, el cliente presentará el aspecto siguiente:

```
static void Main(string[] args) {
    localhost.Sample svc = new localhost.Sample();
    try {
        Console.WriteLine( svc.WhoAmI() );
    } catch ( Exception ex ) {
        Console.WriteLine( ex.ToString() );
```

```
    } finally {

        svc.Dispose();

    }

}
```

Cuando no se aplica seguridad al servicio o aplicación Web, la función Main imprime la información siguiente:

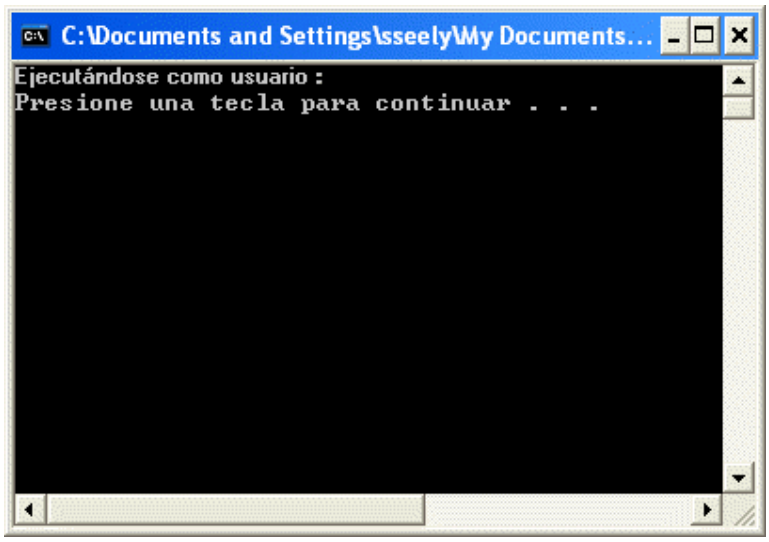

Figura 4. Ejecución sin seguridad, lo que implica que no existe identidad

Si desactiva el acceso anónimo por medio del cuadro de diálogo que se muestra en la figura 3, el cliente no podrá tener acceso al servicio Web y, en su lugar, se obtendrá el siguiente mensaje de error:

```
System.Net.WebException: Error de la solicitud con el código de estado

HTTP 401: Acceso denegado.
```

¿Por qué pasa esto? De forma predeterminada, un proxy de servicio Web no tiene ninguna información sobre el llamador ni sobre qué credenciales se deben pasar. Puesto que no puede autenticarse a sí mismo, falla el intento de llamar al método Web y se inicia una excepción. Si desea pasar las credenciales correctas del usuario actual, lo más fácil será pasar las credenciales predeterminadas del usuario actual. Se debe modificar el bloque **try** del cliente para que quede del siguiente modo:

```
svc.Credentials =

    System.Net.CredentialCache.DefaultCredentials;

Console.WriteLine( svc.WhoAmI() );
```

Esto permitirá que el proxy tenga acceso al método Web ya que puede tomar las credenciales del usuario actual y presentarlas al método Web cuando se produzca el desafío. El servicio Web devolverá el resultado

siguiente:

```
Ejecutándose como usuario: REDMOND\sseely
```

Esto funcionará tanto con la autenticación básica como con la implícita. La información de autenticación sólo es válida para una llamada de servicio Web. Dicho de otro modo, el código del servicio Web no puede llamar a otros servicios Web y hacerse pasar por el llamador utilizando estos mecanismos. Recuerde que si opta por requerir la autenticación básica, deberá también solicitar una conexión SSL para ese archivo, de forma que la identidad del usuario no quede expuesta a ninguna entidad que pueda estar controlando la conexión. En algunas ocasiones puede que desee acceder al servicio Web con una identidad diferente a la del usuario actual. ¿Cómo se puede hacer esto? Definirá las credenciales "manualmente".

Yo tengo un usuario **sseely2** en mi servidor Web local, al que he llamado **Example** y cuya contraseña es **Test$123**. Para establecer las credenciales manualmente se creará un **CredentialCache**. El código que utilice **CredentialCache** deberá llenar la caché con objetos **NetworkCredential**. Al agregar una **NetworkCredential** a la caché, el código tendrá que especificar qué combinación de URL-tipo de autenticación se utilizará para devolver las credenciales especificadas. Se puede llenar la caché con información de identidades para un cierto número de sitios y hacer que la caché devuelva de manera inteligente las credenciales correctas para cada sitio y tipo de autenticación. Utilice el código siguiente para configurar la caché de manera que envíe las credenciales correctas para un desafío de autenticación básica del servicio Web:

```
localhost.Sample svc = new localhost.Sample();
try {
    CredentialCache credCache = new CredentialCache();
    NetworkCredential netCred =
        new NetworkCredential( "Example", "Test$123", "sseely2" );
    credCache.Add( new Uri(svc.Url), "Basic", netCred );
    svc.Credentials = credCache;
    Console.WriteLine( svc.WhoAmI() );
```

Al pasar la URL que se utilizará en la línea que contiene **credCache.Add**, observará que la URL procede del servicio Web en lugar de estar codificado o proceder de otra fuente. A mí me gusta codificar llamadas al método **Add** de esta manera porque requiere un menor esfuerzo comprobar que el extremo del servicio Web y el extremo utilizado por la llamada a **Add** son idénticos.

Si decide utilizar las mismas credenciales para la autenticación implícita, la línea para agregar la información a la caché de credenciales será:

```
credCache.Add( new Uri(svc.Url), "Digest", netCred );
```

La autenticación básica funcionará en el caso de usuarios registrados en el equipo local o en el caso de un usuario registrado en el directorio. La autenticación implícita sólo acepta usuarios registrados en un dominio Windows de confianza.

Otro modo de autenticar llamadores de servicios Web es por medio de autenticación mutua a través de SSL. El emisor y el receptor del mensaje SOAP pueden intercambiar certificados y validarse el uno al otro. El servidor tendrá un certificado si es compatible con SSL. El cliente tendrá un certificado si se ha emitido alguno para el cliente de alguna forma. Si tiene a mano un Certificate Server, necesitará emitirse a sí mismo un certificado y, a continuación, asignárselo a su cuenta de usuario por medio del cuadro de diálogo que se muestra en la figura 2. Para obtener más información, consulte: Mapping Client Certificates to User Accounts (en inglés).

Si tiene certificados disponibles, se puede acceder a ellos a través del subprograma Opciones de Internet del Panel de control. La forma más fácil de tener acceso a este subprograma es a través de Microsoft® Internet Explorer. Si no tiene instalado ningún certificado, es importante que se haga con uno ahora. Sólo debe abrir Internet Explorer y desplazarse hasta un equipo Windows Server que tenga instalado Certificate Server. La URL que necesita es http://*nombre_máquina*/certsrv. Siga las instrucciones en pantalla para solicitar e instalar un certificado de cliente. A continuación, en el menú **Herramientas** de Internet Explorer, haga clic en **Opciones de Internet**, después haga clic en la ficha **Contenido** y, por último, haga clic en **Certificados**. Aparecerá un cuadro de diálogo parecido al que se muestra en la figura 5.

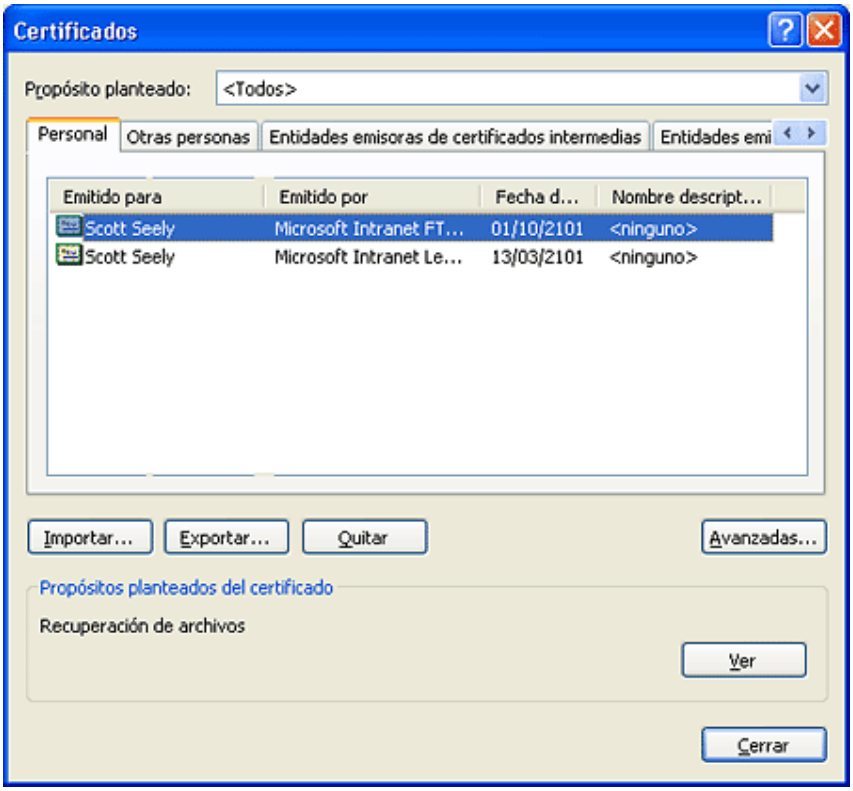

Figura 5. Cuadro de diálogo Certificados

Tendrá que exportar un certificado para que pueda utilizarse para la autenticación del proxy de servicios Web. Para exportar un certificado, haga clic en **Exportar** para abrir el Asistente para exportación de certificados. En el asistente, haga clic en **Siguiente** para aceptar todos los valores predeterminados y, a continuación, seleccione un nombre de archivo en el que se escribirá el certificado. En mi caso he guardado el certificado en c:\temp\secSample.cer. Haga clic en **Siguiente** y, a continuación, en **Finalizar**. Ahora tenemos que asociar ese certificado con un usuario en particular.

1. Repita el procedimiento que utilizó para solicitar que SSL protegiera uno o todos los servicios Web. (Consulte la sección Cifrado y firma con SSL empezando por el procedimiento sobre la apertura de la consola de administración de IIS).
2. Seleccione la casilla de verificación **Enable client certificate mapping** y haga clic en **Edit**.
3. En la ficha de asignación de **1 a 1**, haga clic en **Agregar**.
4. Seleccione c:\temp\secSample.cer
5. En el cuadro de diálogo **Asignar una cuenta**, defina los elementos siguientes:

   - **Asignación**: asignación de ejemplo de HTTP.
   - **Cuenta**: seleccione una cuenta de usuario. En mi caso, seleccioné **sseely2\Example**.
   - **Contraseña**: asigne la contraseña de la cuenta. En mi caso escribí **Test$123**.

No hay ningún problema si no coincide la identidad del certificado con la identidad vinculada con el

certificado. Al hacer corresponder un certificado con una identidad, el servidor simplemente busca otro certificado almacenado que coincida con el recibido. ¿Por qué? Porque puede ser que alguien tenga un certificado de cliente emitido por una autoridad de certificación pública (CA). Al utilizar autenticación de clientes SSL, el servidor puede asignar ese certificado a una identidad en el equipo host sin que tenga que estar asociarlo en modo alguno con el emisor de dicho certificado.

6. Haga clic para confirmar la contraseña y haga clic en **Aceptar** tres veces más para cerrar el cuadro de diálogo.

Ahora necesitará configurar las demás opciones en IIS. Lo primero que tiene que hacer es anular la selección de todos los métodos de autenticación disponibles para que el recurso protegido (archivo .asmx o directorio virtual) tenga los permisos establecidos tal y como se muestra en la figura 6.

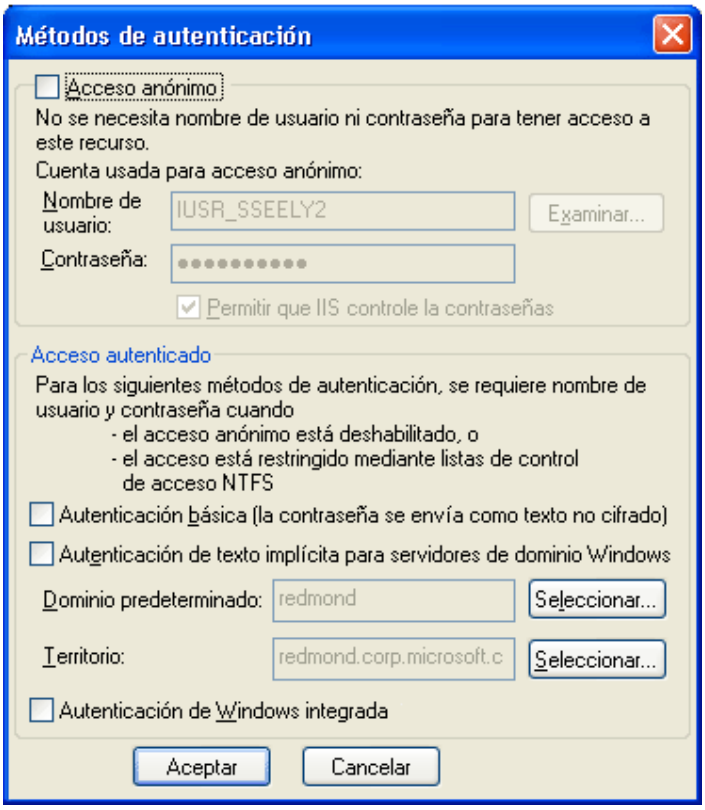

Figura 6. Todos los métodos de autenticación sin seleccionar

A continuación, solicite los certificados de clientes como se muestra en la figura 7.

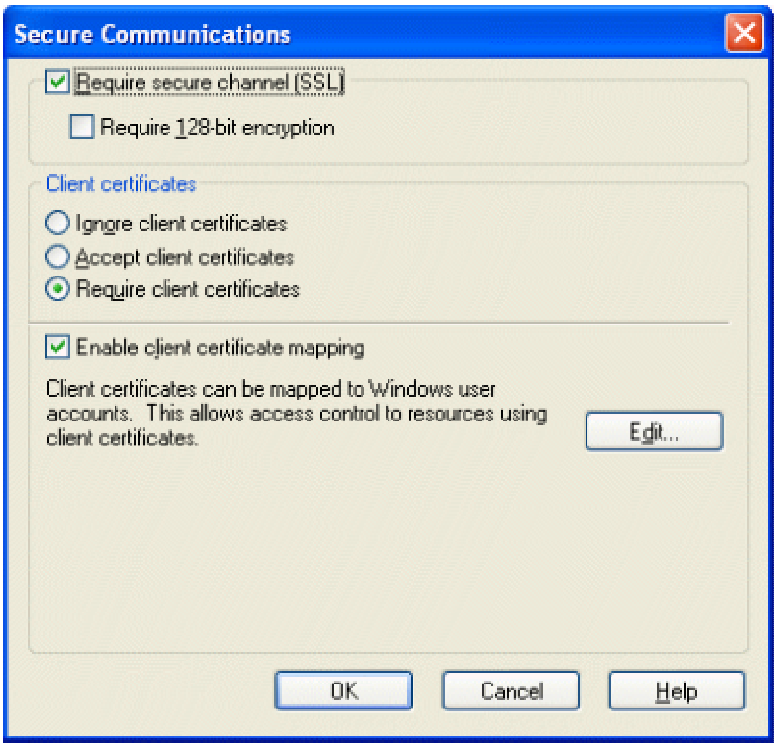

Figura 7. Solicitud de certificados de clientes y SSL

Por último, se debe configurar el cliente para cargar el certificado desde un archivo y presentarlo con el servicio Web. La clase **System.Security.Cryptography.X509Certificates.X509Certificate** sabe cómo leer un certificado X.509. Para cargar el certificado y ponerlo a disposición del servicio Web, lea el certificado y agréguelo a la colección de certificados de cliente del proxy.

```
static void Main(string[] args) {
    localhost.Sample svc = new localhost.Sample();
    try {
        X509Certificate x509 = X509Certificate.CreateFromCertFile(
            @"c:\temp\secSample.cer");
        svc.ClientCertificates.Add( x509 );
        Console.WriteLine( svc.WhoAmI() );
    } catch ( Exception ex ) {
        Console.WriteLine( ex.ToString() );
    } finally {
        svc.Dispose();
```

```
    }
}
```

Como era de esperar, el resultado es el siguiente:

```
Ejecutándose como usuario: SSEELY2\example
```

Al autenticar el usuario mediante autenticación básica, implícita o X.509, también podrá utilizar listas de control de acceso (ACL) para definir los usuarios que tienen acceso al directorio. Una forma de ver la ACL de un archivo o directorio es mediante Windows Explorer. Haga clic con el botón secundario del mouse y, a continuación, haga clic en **Propiedades**. En la ficha **Seguridad** podrá agregar y quitar usuarios y grupos de usuarios, así como manipular las cosas que pueden hacer con los archivos.

No siempre es aconsejable agregar usuarios del servicio Web a Active Directory. Al contrario, puede ser preferible que guarde esta información en otro lugar. Para resolver este problema, se suele recurrir a uno de las dos soluciones siguientes. Una de estas soluciones, común para sitios Web seguros, es emitir para cada usuario un nombre y una contraseña. Estas credenciales se pueden pasar por medio de encabezados SOAP y demás mecanismos. En Cold Storage sample (en inglés) se utilizaron encabezados SOAP personalizados y un módulo HTTP para proporcionar autenticación. La segunda solución implica la creación de un servicio Web de inicio de sesión personalizado. En este caso, el llamador inicia la sesión en un canal seguro como puede ser SSL y obtiene un testigo que puede utilizar cuando llame a otros métodos del servicio Web. Este método se utilizó para Favorites Web Service (en inglés).

Utilización de seguridad de acceso al código

Hasta el momento únicamente hemos revisado modos de identificar usuarios de forma única. Una vez que sepamos quién es el usuario, podremos utilizar dicha información para autorizar al usuario para que tenga acceso a uno o más métodos del servicio Web. El usuario Example es miembro del grupo sseely2\SampleGroup. Si quiero limitar el acceso al método Web **WhoAmI** únicamente a miembros de ese grupo puedo aplicar el atributo **System.Security.Permissions. PrincipalPermissionAttribute**. Yo usaría concretamente el código que sigue:

```
[WebMethod]
[PrincipalPermissionAttribute(SecurityAction.Demand,
        Authenticated=true,
        Name=@"sseely2\Example",
        Role=@"sseely2\SampleGroup")]
public string WhoAmI() {
    return "Ejecutándose como usuario: " +
```

```
        Thread.CurrentPrincipal.Identity.Name;
}
```

Este código es un tanto extremista. Requiere que se conozca el identificador del llamador, que éste pertenezca a un grupo **sseely2\SampleGroup** y que su nombre sea **sseely2\Example**. Es más habitual requerir la pertenencia a un grupo determinado. Esta técnica proporciona un modo sencillo de conceder o denegar el acceso a determinados métodos Web. Utilice seguridad de acceso al código, ya que para proteger el acceso a nivel de .asmx no es suficiente con las listas de control de acceso.

Interoperabilidad

Faltaría a mi obligación si no menciono el tema de la interoperabilidad de los mecanismos de seguridad expuestos anteriormente. Si prevé que kits de herramientas que no sean de Microsoft tengan acceso a su servicio Web, el mecanismo de seguridad más interoperable y mejor probado es utilizar la autenticación básica para identificar al llamador y SSL para cifrar el canal. Para usar este mecanismo con la autenticación integrada de Windows necesitará agregar los nombres de usuario y contraseñas para los usuarios del servidor Web o para el controlador de dominio de Windows correspondiente. La razón es muy simple: muchas pilas de servicios Web no incluyen la porción de HTTP que sabe cómo administrar la autenticación implícita. En muchos casos, puede que la combinación SSL-SOAP no admita el envío de un certificado X.509 del cliente.

Resumen

Se pueden proteger los servicios Web mediante una combinación de características de IIS y de ASP.NET. Los servicios Web de ASP.NET utilizan una caché de credenciales para responder a las solicitudes de varios tipos de autenticación.

Tanto la autenticación básica/implícita como el SSL presentan las mismas desventajas: requieren que se intercambien algunos mensajes entre el emisor y el receptor de mensajes SOAP antes de que se pueda enviar el mensaje de forma segura. Este protocolo de enlace puede repercutir en la velocidad con que se transfieren mensajes SOAP. Acelerar estos procesos es uno de los propósitos de la especificación WS-Security (en inglés). WS-Security abandona las técnicas de protocolo de transferencia para adoptar el modelo de seguridad centrado en el mensaje. Hasta que WS-Security se conozca y utilice más, los mecanismos de seguridad basados en HTTP son la mejor manera de mantener los servicios Web protegidos.