

Seguridad en XML

Luis Enrique Corredera de Colsa

Director general de Flag Solutions S.L.

<http://www.flagsolutions.net>

Resumen

En este trabajo evaluaremos las capacidades que nos brindan las diversas tecnologías existentes para mejorar la seguridad de las aplicaciones que usan XML como formato de intercambio y comunicación de datos. Estudiaremos los recursos que nos ofrece la seguridad a nivel de transporte y a nivel de aplicación para construir aplicaciones interoperables mediante XML seguras remarcando las debilidades en algunos enfoques de uso. Evaluaremos también las características principales de estándares desarrollados sobre XML con objetivo de aumentar la seguridad en las aplicaciones.

1. Introducción

Las tecnologías basadas en XML han supuesto un importante cambio en la forma de pensar de los arquitectos de sistemas. Gracias a las capacidades de interoperabilidad, distribución, integración e independencia de las plataformas y tecnologías subyacentes a las aplicaciones, XML se ha consolidado como el estándar para el intercambio de datos. El tiempo de penetración de ésta tecnología ha sido increíblemente más bajo que el de cualquier otra que se haya encontrado hasta el momento, y sus aplicaciones están avanzando día a día para la descripción de contenidos, desde simples representaciones de registros existentes en bases de datos a la representación de mundos de realidad virtual, la descripción de contenidos multimedia, los nuevos estándares de codificación de contenidos Web, los servicios Web para el intercambio de información entre aplicaciones...

En lo relativo a los servicios Web los individuos más representativos en la industria de las tecnologías de la información (Microsoft, IBM, Sun, Oracle, BEA...) así como grupos de

trabajo del W3C han apostado por XML como la tecnología de intercambio de información, y se han desarrollado estándares como SOAP como protocolo de intercambio de mensajes para el acceso a objetos, WSDL para describir de forma uniforme los servicios ofrecidos y la forma de acceder e interactuar con ellos, UDDI para registrar los servicios Web y la localización de los servicios que se ajusten a las características deseadas.

Si bien los beneficios aportados por los servicios Web al ámbito comercial son importantes (arquitectura descentralizada, independencia de la plataforma, disponibilidad, acceso abierto al público, facilidad y uniformidad en el acceso), los servicios Web traen consigo importantes riesgos desde el punto de vista de la seguridad que deben ser abordados desde diferentes niveles:

- Asegurar la autenticación bidireccional entre el cliente que accede y consume los servicios Web y el proveedor de los servicios accedidos y consumidos.
- Permitir la autorización del acceso a recursos y procesos en un contexto en el que deben poder ser administrados y controlados los accesos por parte de clientes, proveedores, vendedores, competidores, intrusos...
- Identificación y autenticación única de los usuarios de modo que puedan acceder a diversos sistemas sin necesidad de autenticación en cada uno de ellos.
- Garantizar la confidencialidad de los datos intercambiados entre el cliente y el servidor. SOAP no cifra la información y ésta viaja en texto plano por Internet.
- Garantizar la integridad de los datos para protegerlos de alteraciones (accidentales o intencionadas).

- Impedir el repudio de las operaciones.
- Protección frente a los ataques típicos de otras tecnologías. Dado que los servicios Web transcurren por los puertos 80 y 443, los filtrados de puertos clásicos poco pueden hacer frente a los ataques a nivel de aplicación. Sin embargo se puede hacer uso de analizadores de tráfico de nivel de aplicación, que entienden los servicios y son capaces de interceptar posibles ataques a las aplicaciones.

2. Seguridad en el nivel de transporte

La seguridad en el nivel de transporte consiste en aplicar las características que mejoran la seguridad existente en el nivel de transporte que esté siendo usado por el servicio Web en cuestión. Esto es posible gracias a la encapsulación que tiene lugar en la pila de protocolos, desde el nivel de aplicación al nivel de transporte.

En lo referido a seguridad para servicios Web, la seguridad a nivel de transporte se emplea para establecer una seguridad *punto a punto*: la comunicación es segura desde un punto a otro punto, lo que implica que la conexión sea directa, sin intermediarios. Para lo relativo a servicios Web, intermediarios como los enrutadores de Internet no se consideran intermediarios, ya que estos han sido diseñados para soportar la seguridad a nivel de transporte.



Figura 1: Comunicación punto a punto

Cabe señalar que tecnologías que aportan seguridad a nivel de transporte también pueden empleadas a niveles inferiores, como el nivel de red. IPSec es un ejemplo de tecnología que podemos usar para asegurar la confidencialidad y en cierto modo la autenticidad de los consumidores y prestadores de servicios Web.

Podemos aseverar que en lo relativo a servicios Web en los que no existen intermediarios (por ejemplo intermediarios de SOAP), los mecanismos de seguridad subyacentes al nivel de aplicación son suficientes para mantener un grado de confidencialidad aceptable.

Por otra parte, si lo que perseguimos es una seguridad de extremo a extremo, aunque se haga uso de intermediarios de SOAP, se debe recurrir a las técnicas de seguridad propias del nivel de aplicación, incluidas nuevas tecnologías propias de XML que más adelante discutiremos.



Figura 2: Comunicación extremo a extremo

La seguridad implementada en el nivel de transporte no incide en ningún aspecto sobre el mensaje SOAP, sin embargo requiere de modificaciones en las configuraciones de los servidores, los códigos de los clientes para hacer las llamadas apropiadas para la autenticación y algunas soluciones en el hardware de la red.

3. Seguridad a nivel de aplicación

3.1. Seguridad a través de HTTP

El protocolo de transferencia de hipertexto es quizá el protocolo más conocido en Internet, y a través de él circulan la mayor parte de los contenidos Web. Como protocolo, su comportamiento es síncrono (cuando se envía una petición, hay que recibir una respuesta a la misma antes de poder continuar). Las peticiones se realizan envueltas en unas cabeceras del protocolo HTTP, y las respuestas tienen un formato muy similar a las peticiones. Ambas viajan en texto plano por Internet. El protocolo HTTP carece de estados, por lo que implica que cada par de petición y respuesta es un acto individual, sin relación teórica con los demás. Existen diferentes maneras en las que se ha dotado al protocolo

HTTP de comportamiento de sesión emulado (bien mediante cookies, registros de autenticación del navegador,...) pero en cualquier caso no forman parte de la definición del protocolo, sino que son soluciones aportadas y asumidas por los servidores y los clientes. La comunicación sin estado tiene como ventaja la facilidad para adaptar sistemas de distribución de mensajes.

El protocolo HTTP, en su versión 1.1 establece dos tipos de autenticación para restringir el acceso del cliente a los servicios:

- Autenticación básica: es un sencillo protocolo de identificación definido en la especificación del protocolo HTTP en su versión 1.0. Todos los servidores Web y navegadores incluyen la autenticación básica. La autenticación básica funciona de forma fluida a través de casi todos los cortafuegos y proxies. Los toolkits de SOAP incluyen este esquema de autenticación también. Sin embargo, el aspecto negativo de la autenticación básica es que envía las contraseñas en el formato de codificación Base64, que es reversible y muy simple. Por ello, resulta completamente necesario el uso asociado de la autenticación básica en combinación con SSL (Secure Sockets Layer).
- Autenticación de compendio (Digest): su funcionamiento es similar a la autenticación básica, pero su uso no está tan extendido. No obstante está soportada por los principales servidores Web del mercado y los navegadores. La principal diferencia con la autenticación básica de HTTP es que nunca se envía el nombre de usuario ni la clave en la cabecera HTTP, sino un resumen de la misma obtenido mediante el algoritmo MD5 aplicado sobre el usuario, la clave y un valor "nonce" enviado por el servidor a la hora de solicitar la autenticación para el acceso a un recurso. El valor *nonce* queda libre para la elección por parte del fabricante del servidor, pero es sugerido por el estándar usar una combinación de la dirección IP del cliente, el URI al que accede y una marca de tiempo. De esta manera se amplía el espectro de claves, y un ataque de diccionario contra el

resumen expuesto empieza a perder el sentido con la tecnología disponible en el momento actual (Aunque en círculos relativos a seguridad aumenta la desconfianza en el algoritmo de hash MD5, y un prestigioso equipo de investigadores de la Universidad Shandong asegura haber conseguido colisiones en el algoritmo SHA1 en un espacio de búsqueda sensiblemente inferior a las condiciones de seguridad de los algoritmos de Hashing. En las referencias se aportan recursos que justifican la rotura de MD5).

Este enfoque nos puede proveer de una falsa sensación de seguridad al ser cierto que, aunque un atacante que intercepte el tráfico entre el cliente y el servidor no será capaz de conocer el usuario ni la clave. Sin embargo, el método de autenticación por compendio es frágil ante ataques de retransmisión de los credenciales. Un enfoque más sofisticado podría aportarse mediante el modelo de ataque de "hombre intermedio", que facilite un "nonce" creado a propósito y que permita reducir el espacio de búsqueda de usuario y clave para fuerza bruta.

Los mecanismos de autenticación provistos por HTTP no proporcionan confidencialidad de la información transmitida, simplemente limitan el acceso de los usuarios no autorizados a los recursos no autorizados. Desde el punto de vista de la seguridad, el mecanismo de autenticación por compendio es mucho más deseable que la autenticación básica (que debería tender a desaparecer en beneficio de todos). Sin embargo es patente que los mecanismos de autenticación provistos por HTTP se encuentran en el nivel más bajo de seguridad, desde un punto de vista criptográfico.

Es obvia la necesidad de combinar los mecanismos de autenticación a nivel de aplicación con otras tecnologías que nos aporten confidencialidad e integridad en la información. Una alternativa elegible para fortalecer los mecanismos de autenticación expuestos es la combinación de los mismos con SSL. Cuando aplicamos SSL sobre HTTP obtenemos HTTPS (o HTTP asegurado), que nos asegura confidencialidad extremo a extremo en las

conexiones sin intermediarios, y confidencialidad punto a punto en las conexiones con intermediarios SOAP.

Otros mecanismos de autenticación que presentan debilidades similares a los provistos por el protocolo HTTP incluirían la autenticación por formulario en la aplicación Web (en la que una parte de la aplicación se responsabiliza de identificar y autenticar al usuario por las credenciales presentadas) o la autenticación integrada en Windows /NT Challenge Response (NTLM). Éste último es una implementación propia de Microsoft, análoga a la autenticación por compendio, pero solo soportada por sus servidores y sus navegadores. Además de ofrecer los mismos problemas a los mecanismos comentados anteriormente, se añade la dificultad de no estar implementados en todos los posibles sistemas que accedan a consumir los servicios ni en los servidores (lo que termina con la independencia de la plataforma en la implementación del acceso a servicios Web).

3.2. Envío de credenciales en mensajes SOAP

Al aplicar seguridad en el nivel de aplicación, podemos modificar el propio mensaje SOAP para que incorpore los credenciales del usuario que quiere consumir el servicio. Este enfoque requiere que tanto el servicio como el cliente incorporen el mecanismo de autenticación para poder ofrecer el acceso al servicio. La aplicación de este método haría viajar los credenciales en texto plano, a no ser que nos encontremos usando el protocolo SSL para la provisión del servicio Web. En cualquier caso, este enfoque es análogo a la autenticación basada en formulario comentada en el apartado anterior. Admite varias maneras de ser llevado a cabo:

- Enviar los credenciales en cada mensaje SOAP transmitido, de modo que el servidor pueda validarlos para cada petición.
- Enviar los credenciales en el primer mensaje SOAP transmitido, de modo que el servidor pueda validarlos y establecer internamente una sesión para el usuario. El servidor devuelve al usuario un identificador de sesión que el cliente podrá usar en cada petición posterior, y el servidor simplemente

comprobar si la sesión fue establecida previamente, y si el acceso al recurso solicitado es concedido al usuario propietario de la sesión indicada. Con la finalidad de aumentar la seguridad en este enfoque, es recomendable establecer periodos de caducidad para las sesiones.

Es importante remarcar que este modelo de autenticación debe ir siempre acompañado del protocolo SSL para evitar que los credenciales se envíen en texto plano.

Con la finalidad de mejorar el rendimiento de los servicios Web en entornos seguros, algunos autores proponen relajar la seguridad implementando un servicio Web de autenticación bajo HTTPS para obtener la clave de sesión y relajar la seguridad del resto de servicios, distribuyéndolos mediante HTTP sin cifrado. Bajo mi punto de vista, este enfoque es pobre e inaceptable ya elimina la propiedad de confidencialidad en la información y puede permitir que un atacante no solo pueda obtener todo el contenido de la comunicación del cliente con el servidor, sino que también pueda inyectar peticiones en nombre del cliente mediante el uso de la clave de sesión proporcionada por el cliente en la conexión no cifrada. Establecer un esquema de autenticación que no preserve la confidencialidad del cliente, la integridad de la información y que permita la suplantación de personalidad en un canal no cifrado puede llegar a ser más costoso computacionalmente que implementar los servicios Web a través de HTTPS, y nunca cubriría los requisitos mínimos de confidencialidad con las tecnologías presentadas hasta el momento.

3.3. Autenticación Kerberos y basada en tickets

La autenticación basada en tickets tiene lugar en el nivel de aplicación, lo que significa que se puede utilizar con cualquier protocolo de transporte. Por lo general, la autenticación basada en tickets usa firmas digitales, certificados digitales y tecnologías de codificación de clave pública. Dentro de una aplicación podemos incluir código para llevar a cabo la autenticación, la codificación del contenido de los mensajes,

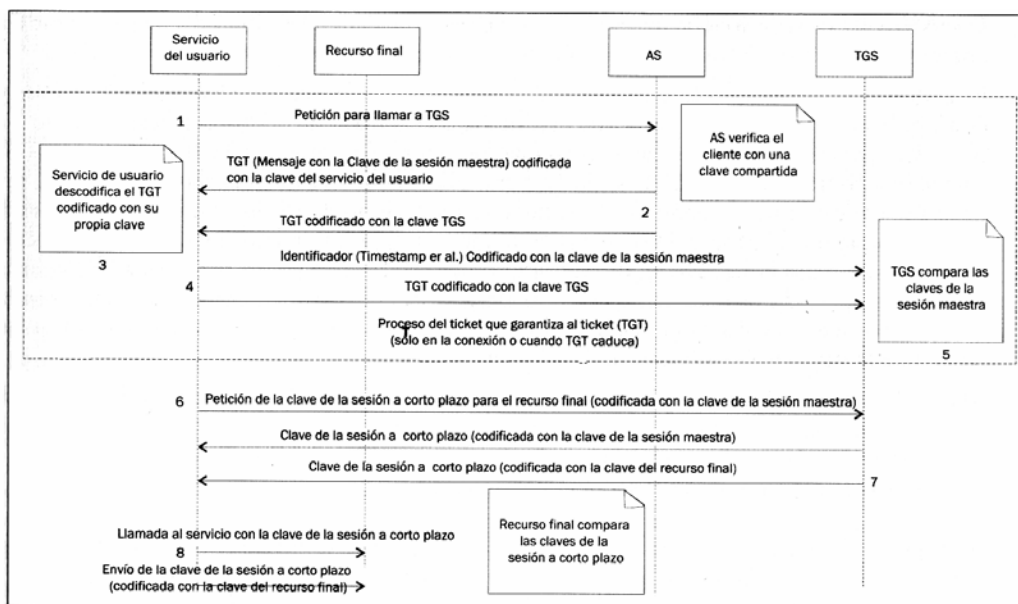


Figura 3: Diagrama de secuencia de autenticación Kerberos

firmarlos digitalmente o validar las firmas de los mensajes recibidos.

La autenticación basada en tickets, como vemos, implica un mayor esfuerzo en el desarrollo

vistos para la autenticación en el protocolo HTTP o en los mensajes SOAP.

Compañías como Microsoft está haciendo uso de la autenticación basada en tickets para los servicios de autenticación (Microsoft Passport) de una sola conexión (single sign on) y sus servicios Hailstorm.

Kerberos se desarrolló en el MIT, con el objetivo de ser un servicio de autenticación. Su punto de partida es que solo un número limitado de máquinas pueden ser muy seguras, por lo que trata todas las máquinas de la red como inseguras, a excepción de una de ellas. Kerberos permite la autenticación mutua entre usuarios y servicios, y dispone de mecanismos de delegación.

El mecanismo de funcionamiento de Kerberos es simple. La figura 3 presenta el diagrama de secuencia de Kerberos, y a continuación se

de las aplicaciones, sin embargo la flexibilidad que proporciona es elevada y la seguridad aportada es bastante superior a los esquemas

describe los pasos, asociados a la numeración de la figura:

1. Antes de acceder a cualquier servicio, el servicio del usuario solicita un ticket para contactar con el Ticket Granting Server (TGS) como si fuera cualquier otro servicio. Esta petición es enviada al Authentication Service (AS).
2. El Authentication Service crea una clave de sesión maestra y hace dos copias de la misma. Coloca una copia en un mensaje y lo codifica con la clave que pertenece al servicio de usuario. Este mensaje es el *Ticket Granting Ticket (TGT)*. El AS coloca la otra copia de la clave de la sesión en otro mensaje que codifica con una clave que pertenece al TGS y envía ambas copias al servicio de usuario.
3. El servicio de usuario recibe los dos mensajes del AS. Puede abrir el mensaje que ha sido codificado con su clave, pero no puede leer el mensaje codificado con la clave del TGS.
4. El servicio del usuario crea un mensaje nuevo y coloca una marca de tiempo que servirá como

una clave de sesión adicional y lo codifica con la clave de sesión maestra. Coge el mensaje recibido del AS y lo cifra con la clave de TGS, junto con el nuevo mensaje de autenticación adicional que contiene la marca de tiempo. Finalmente envía el mensaje al TGS.

5. El TGS recibe los dos mensajes y abre el que contiene la clave de sesión maestra (solo TGS puede abrir este mensaje, que envió originalmente el AS). El TGS usa el segundo mensaje que envió el servicio de usuario y lo descifra usando la clave maestra de sesión. Obtiene la información de autenticación adicional añadida por el servicio de usuario (timestamp, fecha, hora...). Con ello el TGS entiende la identidad del servicio del usuario. Ahora, el servicio del usuario y el servicio que otorga los tickets se pueden comunicar el uno con el otro usando la clave de la sesión maestra.
6. El servicio de usuario solicita un ticket de corto plazo al TGS para contactar con cualquier recurso de la red.
7. El TGS contesta con una clave de sesión de corto plazo para acceder al servicio de recurso de destino. Esta clave de corto plazo está codificada con la clave de sesión maestra que se encuentra en el TGT, no con la clave del servicio de usuario. El TGS envía otra copia de la clave de sesión de corto plazo al cliente, pero codificada con la clave pública de recursos que presta el servicio. La clave de sesión de corto plazo tiene un tiempo de vida muy breve, incluso de un solo uso. El propio TGT tiene un periodo de vida muy corto, normalmente de 8 horas, de modo que si es robado, se puede reemplazar muy fácilmente y su periodo de validez es muy reducido (normalmente las contraseñas no se cambian con tanta frecuencia, y si son robadas, el compromiso que supone para los sistemas es mucho mayor).
8. Ahora el servicio del usuario puede contactar con el recurso de destino usando la clave de la sesión a corto plazo y, también, le envía el ticket codificado con la clave pública del recurso. El recurso final descodifica este ticket para obtener la clave a corto plazo, que usa a su vez para descodificar la petición desde el servicio del usuario.

Si tenemos en cuenta que el TGT y otros mensajes codificados que pueden ser enviados por los

servidores se pueden intercambiar por medio de bloques de XML, parece claro que los tickets se puede intercambiar a través de la Web para facilitar la autenticación en el acceso a servicios Web.

Como podemos deducir del funcionamiento de Kerberos, su uso para la implementación de seguridad en los servicios Web nos proporciona un alto grado de integridad, confidencialidad y autenticidad.

El uso de Kerberos en XML requiere del uso de XML Encryption y de XML Signature que veremos más adelante. Existe un XML-Schema que define el tipo de documento que se debe emplear para la autenticación Kerberos basada en XML.

4. Tecnologías para seguridad inherentes a XML

A continuación veremos especificaciones que se han desarrollado para dotar los propios documentos XML de mayor seguridad, usando XML. También veremos la especificación de privacidad P3P, ya que la privacidad es un tema candente en el panorama actual, e incide en cierto modo sobre la seguridad.

4.1. XML Signature

XML Signature es un estándar desarrollado por el W3C que nos permiten autenticar al remitente, asegurar la integridad de partes de los documentos XML transportados. Puede ser aplicado a cualquier tipo de contenido digital (objetos de datos binarios), incluyendo documentos XML. A través de la firma digital nos permite ofrecer garantía de autenticidad de los datos, la identificación de la identidad del emisor del mensaje, la validación de autenticidad e integridad del mensaje y, derivado de estas características, la propiedad de no repudio en los mensajes.

Un documento XML Signature incluye varios elementos, como puede verse en la **¡Error! No se encuentra el origen de la referencia.:**

```

<Signature Id="EjemploXMLSignature"
  xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod
      Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
    <SignatureMethod
      Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
    <Reference
      URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/"
      <DigestMethod
        Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>MCOCFrVLtRlk=...</SignatureValue>
  <KeyInfo>
    <KeyValue>
      <DSAKeyValue>
        <p>...</p><q>...</q><g>...</g><y>...</y>
      </DSAKeyValue>
    </KeyValue>
  </KeyInfo>
</Signature>

```

Figura 4: Ejemplo de documento XML Signature

Existe un elemento "Signature" que encapsula la firma digital dentro del documento. Contiene tres sub-elementos: SignedInfo, SignatureValue y KeyInfo. El elemento SignedInfo contiene la descripción de dos algoritmos, y las referencias a los objetos que se van a firmar. El algoritmo descrito por el elemento CanonicalizationMethod se aplica para canonizar¹ el elemento SignedInfo, y se aplicará el primero en el orden de la generación de la firma. La canonización del documento puede llevarse a cabo de dos formas: incluyendo los comentarios u omitiendo los comentarios del documento XML. El elemento SignatureMethod indica el algoritmo que será empleado para la obtención de la firma digital a partir del objeto canonizado después de aplicar el método de canonización. En la **¡Error! No se encuentra el origen de la referencia.** se muestra

¹ La forma canónica de un documento XML es una manera de mostrar una representación física que asegura que si dos documentos XML son aparentemente diferentes, pero sus canónicos son iguales, los dos documentos son equivalentes.

información que describe qué es lo que se firma y cómo se lleva a cabo esa firma; gracias a esta información, la firma puede ser creada de nuevo para verificar su validez.

que el documento presentará una firma generada mediante el algoritmo DSA (basado en DSS). Otra posibilidad de algoritmo de firma recogida por el estándar es PKCS-1 (basado en RSASSA-PKCS1-v1_5). El resultado obtenido de la aplicación del algoritmo de firma, debidamente representado, se incluye dentro del elemento SignatureValue codificado en Base64.

Cada elemento Reference que encontremos en el elemento SignedInfo incluye una referencia al objeto que será firmado. El elemento Reference incluye un elemento DigestMethod que indica el algoritmo de firma digital que se aplicará al objeto referenciado por este elemento. También incluye un elemento DigestValue que es el valor resultante de su firma. El elemento Reference también puede incluir un elemento Transform opcional, que indique las transformaciones que se han aplicado al objeto antes de obtener valor de la firma.

El elemento `SignatureValue` contiene la firma digital asociada al elemento `SignedInfo` del documento. El elemento `KeyInfo` es un elemento opcional que se puede incluir en el documento para indicar la clave que se ha de usar para validar la firma. El elemento `KeyValue` contiene el valor de la clave que hay que aplicar a la firma. El elemento que se encuentra dentro del `KeyValue` puede ser de los siguientes tipos definidos en el esquema `xmldsig`:

- `DSAPublicKey`
- `RSAPublicKey`
- `X509Data`
- `PGPData`
- `SPKIData`
- `MgmtData`

La validación de las firmas de un documento XML Signature se lleva a cabo en dos pasos:

1. Se calcula la firma del elemento `SignedInfo` usando la información contenida en el elemento `KeyInfo` (si existiera), y se compara con el valor que se encuentra en el elemento `SignatureValue`.
2. Si la primera validación se ha llevado a cabo de manera correcta, se recalculan los `DigestValue` de cada objeto `Reference` encontrado en el elemento `SignedInfo`.

El trabajo con XML Signature es posible en Java a través de la JSR 105 (Java Specification Request #105), aprobada el día 6 de Junio de 2005. Esta especificación forma parte del WSDP de Java.

La fundación Apache también dispone de implementación de una API para la generación y validación de XML Signature.

4.2. XML Encryption

XML Encryption es un estándar desarrollado por el W3C que describe el modo de utilizar XML para representar recursos de forma digital y codificada. La especificación está pensada para ser usada junto con la especificación de firmas digitales XML Signature para poder firmar y cifrar el contenido de los documentos XML. Dispone de posibilidad de uso de varios algoritmos de codificación dejando así las puertas abiertas a la introducción de nuevos algoritmos

que aparezcan en un futuro. Otra característica de XML Encryption es el soporte para diferentes granularidades en el cifrado de un documento. Consideremos el documento XML (extraído de la especificación de sintaxis y procesado de XML Encryption) mostrado en la Figura 5: muestra un fragmento de información para un sistema de pago de un comercio. Contiene información relativa a una tarjeta de crédito: nombre, número, límite de operación, banco emisor y fecha de caducidad. Esta información es muy sensible como para circular por la red en texto plano.

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>4019 2445 0277 5567</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Figura 5: Fragmento XML sin cifrar

Podemos hacer uso de XML Encryption para cifrar partes del documento, siguiendo diferentes estrategias:

- Se puede cifrar un elemento completo, por ejemplo, en la Figura 6 se cifra el elemento `CreditCard`, lo que hace que una simple inspección del documento cifrado ni siquiera revele la existencia de información de tarjeta de crédito.

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'
    xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
</PaymentInfo>
```

Figura 6: Cifrado de un elemento completo

El elemento `CipherData` contiene la serialización del elemento `CreditCard` cifrado.

- En algunos casos puede ser conveniente que un individuo conociera que el pago se está realizando mediante una tarjeta de crédito, con un límite de 5.000 USD, pero no publicar ni el número, ni la fecha de caducidad ni el emisor. En este caso, como muestra la Figura

- 7, se recurre al cifrado del contenido del elemento CreditCard, pero no se cifra el elemento CreditCard.
- Otro posible escenario sería la situación en la que solamente fuera necesario cifrar el número de tarjeta de crédito, pero sin cifrar el propio elemento Number. En este caso estaríamos cifrando el contenido de un elemento que solo contiene información de caracteres, no contiene otros elementos.

```
<?xml version='1.0' ?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
        Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C56</CipherValue>
        </CipherData>
      </EncryptedData>
    </Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Figura 7: Cifrado del contenido de un elemento que contiene más elementos

Otras alternativas de granularidad para XML Encryption incluyen el cifrado de un documento completo, evitando cualquier detalle del mismo al no existir ningún elemento salvo los propios del cifrado. También es posible cifrar documentos cifrados (lo que se conoce como Super-Encryption).

```
<?xml version='1.0' ?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
        Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C56</CipherValue>
        </CipherData>
      </EncryptedData>
    </Number>
    <Issuer>Examples Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Figura 8: Cifrado del contenido de un elemento que contiene caracteres

XML Encryption soporta diferentes algoritmos de cifrado. Para el cifrado de bloques se citan los siguientes:

- TRIPLEDES
- AES128

- AES256
- AES192

No se especifican algoritmos para el cifrado de flujos de octetos, sin embargo si se hace referencia a otra serie de algoritmos para el intercambio de claves, resúmenes de los mensajes, y la autenticación de los mismos mediante XML Signature.

4.3. P3P (Platform for Privacy Preferences)

P3P en sí mismo no constituye ningún estándar para el incremento de la seguridad en los servicios Web, sin embargo es una especificación de estándar que permite el desarrollo de herramientas y servicios que ofrecen a los usuarios un mayor control sobre la información personal que se maneja en Internet.

La privacidad de los datos personales es una preocupación candente que en España se ve reforzada por la LOPD (también vigente bajo otras denominaciones en Europa).

Cuando un usuario accede a un servicio Web y acepta las políticas de privacidad de los proveedores de servicios, en muchos casos lo hace sin ser plenamente consciente de lo que ha aceptado. Los textos que especifican dichas políticas son largos y en muchos casos difíciles de comprender por el usuario.

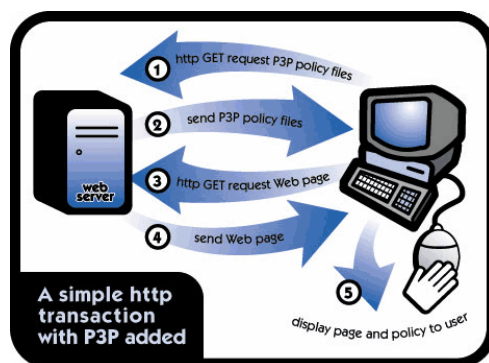


Figura 9: Transacción con P3P

P3P permite a los sitios Web usar un formato estandarizado para la especificación de sus políticas de privacidad, que puede ser procesado automáticamente por dispositivos. Un navegador que soporte P3P podrá decidir automáticamente qué hacer al encontrar un sitio con P3P y tener configurado el usuario sus restricciones en el uso de la información de carácter personal, como ilustra la Figura 9.

P3P activado en el sitio Web y en el navegador puede alertar al usuario del acceso a un sitio Web que no cumple con sus criterios de tratamiento de información personal. Supongamos el ejemplo de un catálogo de productos online:

- Las páginas del catálogo podrían ostentar información P3P relativa al registro de la bitácora del servidor y las peticiones HTTP.
- En una hipotética aplicación de carrito de la compra, para efectuar pedidos en el catálogo Web, el sitio Web puede querer establecer cookies en el navegador, con la finalidad de mantener una sesión. La información P3P debe reflejar esta característica, y el usuario aceptarla o no.
- Suponiendo que se efectúe una compra en el catálogo, en un formulario de transacción que solicite el nombre, apellidos, teléfono, número de tarjeta de crédito, dirección de envío... el servicio Web debería presentar una información P3P relativa al uso que se hará de los datos introducidos. Por ejemplo, el usuario podría tener configurado un comportamiento en el navegador que bloquee las páginas que soliciten el número de teléfono, lo que haría que la transacción se bloqueara y no se realizara el pedido. Con P3P se pueden originar reglas más complejas, de modo que el usuario podría elegir un comportamiento de bloqueo siempre que el sitio Web presentara en su política de privacidad que el número de teléfono será transferido a terceros para alguna finalidad, y aceptar la petición en caso de que solo vaya a ser usado por la entidad que lo solicita.

El hecho de la existencia de políticas P3P en un sitio Web no implica que el propietario del servicio cumpla con las mismas, y está fuera del ámbito de P3P la comprobación de dicho cumplimiento, si bien P3P supone una base para la

automatización de las comprobaciones mediante software de control o de regulación implantado por organismos de regulación de estos aspectos legales.

4.4. SAML (Security Assertions Markup Language)

SAML (Lenguaje de marcas de afirmación de seguridad) es un framework propuesto por OASIS para servicios Web que facilita el intercambio de información de autenticación y autorización de usuarios entre empresas asociadas comercialmente.

Veamos un ejemplo claro de la utilidad de SAML: un usuario se identifica en el servicio Web de su operador de líneas aéreas habitual, y reserva un pasaje en un avión. Como el usuario es consciente de que necesitará un coche en el destino, se identifica y autentifica en el servicio Web de la empresa de alquiler de vehículos y reserva su vehículo para tenerlo disponible a su llegada. Como el viaje durará más de un día, el usuario se identifica y autentifica en el servicio Web de la cadena de hoteles y reserva la habitación para su viaje.

Completar la operación ha requerido de tres identificaciones y autenticaciones por parte del usuario. Mediante el uso de SAML, suponiendo asociadas la compañía de vuelos, la cadena de hoteles y el arrendador de vehículos, el usuario solamente debería haber provisto sus datos de identificación una sola vez, y el servicio Web que hubiera recibido dichos datos, los delegaría en los sucesivos servicios para la reserva. La finalidad es similar a la estudiada en el apartado de Kerberos, pero además puede incluir información relativa al negocio de los servicios Web, como puede ser el crédito del usuario, lo que le permitirá acceder ricamente a la compra de los servicios que puede costearse el usuario.

SAML está diseñado para poder trabajar sobre HTTP, SMTP, SOAP, y otros protocolos basados en XML. Sus elementos principales son los siguientes:

- Aserciones: las aserciones que son declaraciones de hechos del usuario. Hay tres

tipos: de autenticación (donde el usuario demuestra ser quien dice que es), de atributo (que añaden información detallada relativa al usuario) y de autorización (que definen que acciones puede llevar a cabo el usuario).

- Protocolo de petición y respuesta: regula la forma en que se llevan a cabo las actividades de petición y respuesta SAML. Actualmente se soporta SOAP sobre HTTP.
- Bindings: expresan la manera exacta en la que los mensajes de SAML se incluyen en los protocolos de comunicación y se trazan para la consecución del protocolo de petición y respuesta.
- Perfiles: dictan la forma en que los mensajes SAML se pueden transferir a través de sistemas de comunicación. El perfil de SOAP para SAML dicta como insertar los mensajes SAML en SOAP, como afecta a los encabezados SOAP y como debe comportarse SOAP en caso de llegar a un estado de error en SAML.

5. Conclusiones

La rápida adopción de los servicios Web en la industria de las tecnologías de la información ha traído consigo un rápido avance en lo que a seguridad se refiere, buscando la satisfacción de las características comentadas en la introducción. Mediante Kerberos podemos asegurar la autenticación mutua entre clientes y servidores. Apoyándonos en SAML podemos permitir la autenticación e identificación única de los usuarios, facilitando la administración de su autorización por parte de los proveedores de servicios, sin olvidar que SAML no hace frente a las estrategias de autenticación, y la debilidad que supone el uso de los mecanismos propios de HTTP. Gracias a XML Encryption podemos agregar confidencialidad a los documentos XML y con XML Signature garantizamos la integridad de la información enviada y evitamos el repudio del envío de mensajes. Aún así, y con la finalidad de proteger los servicios Web de los clásicos ataques y vulnerabilidades propias de las redes de comunicación, no debemos descuidar la atención en la seguridad que se puede implantar a nivel de transporte, mediante el uso de SSL, IPSEC, firewalls, sistemas IDS, honeypots y otros

mecanismos como los analizadores de tráfico a nivel de aplicación.

Es preciso resaltar que el desarrollo de servicios Web seguros requiere la atención sobre la seguridad desde el momento más temprano del diseño de los servicios, y que el uso de los estándares vistos supone un aumento en los esfuerzos de desarrollo. Sin embargo, su forma de uso es eficaz, y nos asegura unos resultados favorables desde el punto de vista de la seguridad, aunque ligados a la fortaleza de los algoritmos criptográficos empleados. No obstante, los estándares definen el uso de los algoritmos criptográficos de forma abierta, pudiendo añadir en un futuro nuevos algoritmos más robustos que los actuales.

6. Referencias

- [1] Álvarez Marañón, G.: "Seguridad en servicios web XML". Publicado el 05/02/2003 en el boletín número 90 de Criptonomicón. <http://www.iec.csic.es/criptonomicon/boletines/boletin90.txt>
- [2] Bartel, M., Boyer, J., Fox, B., LaMacchia, B., Simon, E.: "XML-Signature Syntax and Processing". W3C Recommendation 12 February-2002. <http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/>
- [3] Cranor, L. et al: "The Platform for Privacy Preferences 1.0 (P3P1.0) Specification". W3C recommendation. 16-April-2002. <http://www.w3.org/TR/P3P/>
- [4] Fielding, R et. al.: "Hypertext Transfer Protocol -- HTTP/1.1" (RFC 2616). <http://www.ietf.org/rfc/rfc2616.txt>
- [5] Franks, J et al: "HTTP Authentication: Basic and Digest Access Authentication" (RFC 2617). <http://www.ietf.org/rfc/rfc2617.txt>
- [6] Giovanni Della-Libera et al: "Web Services Security Kerberos Binding". Dec-2003. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/ws-security-kerberos.asp>
- [7] Imamura T., Dillaway, B., Simon, E.: "XML-Encryption Syntax and Processing". W3C Recommendation 10 December 2002..

<http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/>

- [8] Kaliski, B., Staddon, J.: "PKCS #1: RSA Cryptography Specifications Version 2.0" (RFC.2437).
<http://www.ietf.org/rfc/rfc2437.txt>
- [9] Kaminsky, Dan: "MD5 to be considered harmful some day". 06/12/2004.
http://www.doxpara.com/md5_someday.pdf
- [10] Kaminsky, Dan: "MD5 to be considered harmful some day post announcement". 06/12/2004.
<http://www.securityfocus.com/archive/1/383574/30/0/threaded>
- [11] Kent, S., Atkinson, R: "Security Architecture for the Internet Protocol" (RFC 2401).
<http://www.ietf.org/rfc/rfc2401.txt>
- [12] Klima, Vlastimil: "Finding MD5 Collisions – a toy for a notebook". 05/03/2005.
http://cryptography.hyperlink.cz/md5/MD5_collisions.pdf
- [13] Kohl, J.: "The Kerberos Network Authentication Service (V5)" (RFC 1510).
<http://www.ietf.org/rfc/rfc1510.txt>
- [14] Simon E., Madsen P., Charlisle A.: "An Introduction to XML Digital Signatures". 08/08/2001.
<http://www.xml.com/pub/a/2001/08/08/xmlsig.html>

Referencias Web

- [15] Kerberos, the network authentication protocol. <http://web.mit.edu/kerberos/www/>.
Accedido el día 10 de Junio de 2005.
- [16] Apache XML Security (Apache XML Project). <http://xml.apache.org/security/>.
Accedido el día 10 de Junio de 2005.
- [17] Oficina española del W3C. "Guía breve de privacidad y P3P".
<http://www.w3c.es/divulgacion/guiasbreves/PrivacidadP3P>. Accedida el día 11 de Junio de 2005.
- [18] Oficina española del W3C. "Guía breve de seguridad".
<http://www.w3c.es/divulgacion/guiasbreves/Seguridad>. Accedida el día 10 de junio de 2005.
- [19] OASIS Consortium: "SAML v2.0 Standard Specification".
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security#samlv20 Accedida el día 13 de Junio de 2005.