Improving the Security of Web Applications using XML Digital Signatures

Prof. Mayuri A. Jain
Computer Engineering Department
A. P. Shah Institute of Technology
Thane-West, Maharashtra, India
majain@apsit.edu.in

Prof.(Dr.) Pravin P. Adivarekar
Computer Engineering Department
A. P. Shah Institute of Technology
Thane-West, Maharashtra, India
ppadivarekar@apsit.edu.in

Prof. Sukhada S. Aloni
Computer Engineering Department
A. P. Shah Institute of Technology
Thane-West, Maharashtra, India
ssaloni@apsit.edu.in

Abstract— In today's competitive business world, where all the enterprises need to expand their business environments over the Web for consumers, employees, and partners, digital security will play a very important role in establishing the trust and credibility. Current security technologies commonly implemented are insufficient for securing business transactions on the Web. Secure sockets layer (SSL) over HTTP (HTTPS) is the standard method used to secure Web content. However, SSL addresses only a subset of security requirements. It can establish a reliable private connection and authenticate peer identities but doesn't offer non-repudiation, which is an equally critical security requirement for today's business applications. Also, because SSL is a transport-layer protocol, it only protects data while in transit between the client (browser) and Web server.

Keywords- non-repudiation; security; web applications; public key cryptography; XML Digital Signature

1. INTRODUCTION

Due to rapid development of technology, organizations are in the process of heading towards paperless environments. They have channeled their efforts to achieve this by adopting automation and by digitalizing paper documents through a series of techniques that include imaging, scanning and Electronic Document Interchange. The digital form of document enables easy management, circulation and makes it available anytime and anywhere by using Internet. Most commonly deployed security mechanisms, generally suitable for low-value business-to-consumer transactions, do not provide the enhanced security or flexibility required for protecting important high-value commercial transactions and the sensitive data exchanges that comprise them.

For instance, Secure sockets layer (SSL) over HTTP (HTTPS) is the standard method used for secured interchange of sensitive data between a browser and Web server, but once received, the data is all too frequently left unprotected on the server. In fact, SSL protects the data at a point in its travels when its confidentiality is, relatively speaking, far less likely to be attacked. However, SSL addresses only a subset of security requirements. It can establish a reliable private connection and authenticate peer identities but doesn't offer non-repudiation, which is an equally critical security requirement for today's business applications. Also, because SSL is a transport-layer protocol, it only protects data while in transit between the client (browser) and Web server. It doesn't cover requirements vis-à-vis end-to-end message-level protection.

Now-a-days, investment on system/data security in any organization has increased greatly, due to numerous attack threats lurking everywhere. So, Digital Signature is one of the many ways to authenticate the data at any point of time. One of the major advantages of Digital Signature is that it ensures authenticity, non-repudiation and integrity of data, which in today's date is of great importance. From using traditional Digital Signature methods we have come a long way today to using XML Digital Signatures. W3C has introduced the syntax and promoted the usage of XML Digital Signatures.

2. LITERATURE REVIEW

The main requirement in the management of digital documentation is its equivalence, from a legal perspective, to paperwork, affixing a signature on a digital document is the fundamental principle on which are based the main processes of authorization and validation, apart from the specific area of application. The popularity and growth of Internet have been a driving force to make extensive use of technology to extend business operations with digitalization of documents. However the need has also been felt to maintain authenticity and security of information. A replay attack resilient mechanism was therefore required to embed digital signature into

documents, along with a timestamp from authorized time stamping authority.

Using digital signatures and SSL in combination complements each technology's innate capabilities. A digital signature, which requires a public key infrastructure (PKI), is a well-accepted, legally valid, electronic equivalent of a handwritten signature in most parts of the globe. Digital signatures can also replace handwritten signatures in workflow-based Web transactions, resulting in completely automated, faster, and efficient work flows. Web applications across several industry segments can benefit from digital signatures [1].

3. FUNDAMENTAL CONCEPTS

As important as protecting the confidentiality of business messages is ensuring their long-term authenticity (who sent them?), data integrity (have they been modified in transit?), and support for non-repudiation (can the sender deny sending them?); in other words, functionality that ubiquitous, existing Internet security technologies (e.g., SSL and username/password) do not provide alone. The globally-recognized method for satisfying these requirements for secure business transactions is to use digital certificates to enable the encryption and digital signing of the exchanged data. A brief introduction to these concepts is provided below. The term "public key infrastructure" (PKI) is used to describe the processes, policies, and standards that govern the issuance, maintenance, and revocation of the certificates, public, and private keys that the encryption and signing operations require.

Public Key Cryptography and Digital Signature

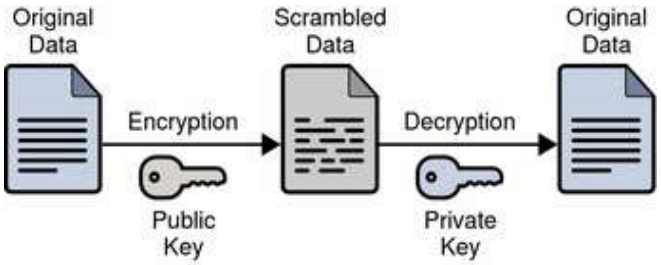

Figure 1: Public Key Cryptography

Public key cryptography allows users of an insecure network, like the Internet, to exchange data with confidence that it will be neither modified nor inappropriately accessed. This is accomplished through a transformation of the data according to an algorithm parameterized by a pair of numbers -- the so-called public and private keys. Each participant in the exchange has such a pair of keys. They make the public key freely available to anyone wishing to communicate with them, and they keep the other key private and protected. Although the keys are mathematically related, if the cryptosystem has been designed and implemented securely, it is computationally infeasible to derive the private key from knowledge of the public key.
The nature of the relation between the public and private keys is such that a cryptographic transformation encoded with one key can only be reversed with the other. This defining feature of public key encryption technology enables confidentiality because a message encrypted with the public key of a specific recipient can only be decrypted by the holder of the matching private key (i.e., the recipient, if they have properly protected access to the private key). Even if intercepted by someone else, without the appropriate private key, this third party will be unable to decrypt the message. It's crucial that confidentiality can be supported without requiring that the sender and recipient exchange any "secret" data in addition to the message itself, as is the case in symmetric cryptography. Indeed it was precisely the requirement of key exchange between sender and recipient, which quickly becomes impractical as the number of participants increases, that motivated the invention of public-key cryptography.

Although confidentiality is typically the first aspect of cryptography that comes to mind, the special relationship between public and private keys also enables functionality that has no parallel in symmetric cryptography; namely, authentication (ensuring that the identity of the sender can be determined by anyone) and integrity (ensuring that any alterations of the message content can be easily spotted by anyone). These new features and, through them, support for non-repudiation (ensuring the origin or delivery of data in order to protect the sender against false denial by the recipient that the data has been received or to protect the recipient against false denial by the sender that the data has been sent) provide electronic messages with a mechanism analogous to signatures in the paper world, that is, a digital signature.

Imagine Anna has a box too. But this one is a box with a very special lock. This lock has three states: A (locked), B (unlocked) and C (locked). And it has two separate (yes, two) keys. The first one can only turn clockwise (from A to B to C) and the second one can only turn anticlockwise (from C to B to A). Anna picks the first one of the keys and keeps it to herself. We will call this key her "private" key, because only Anna has it. We will call the second key, her "public" key: Anna makes a hundred copies of it, and she gives some to friends and family, she leaves a bunch on her desk at the office, she hangs a couple outside her door, etc. If someone asks her for a business card, she hands him a copy of the key too. So, Anna has her private key that can turn from A to B to C. And everyone else has her public-key that can turn from C to B to A. We can do some very interesting things with these keys. First of all, imagine you want to send Anna a very personal document. You put the document in the box and use a copy of her public-key to lock it. Remember, Anna's public-key only turns anticlockwise, so you turn it to position A. Now the box is locked. The only key that can turn from A to B is Anna's private key, the one she's kept for herself. That's it! This is what we call public-key encryption: Everyone who has Anna's public-key (and it's easy to find a copy of it, she's been giving them away, remember?), can put documents in her box, lock it, and know that the only person who can unlock it is Anna.

To create a digital signature for a message, the data to be signed is transformed by an algorithm that takes as input the private key of the sender. Because a transformation

determined by the sender's private key can only be undone if the reverse transform takes as a parameter the sender's public key, a recipient of the transformed data can be confident of the origin of the data (the identity of the sender). If the data can be verified using the sender's public key, then it must have been signed using the corresponding private key (to which only the sender should have access).

For signature verification to be meaningful, the verifier must have confidence that the public key does actually belong to the sender (otherwise an impostor could claim to be the sender, presenting her own public key in place of the real one). A certificate, issued by a Certification Authority, is an assertion of the validity of the binding between the certificate's subject and her public key such that other users can be confident that the public key does indeed correspond to the subject who claims it as her own.

Largely due to the performance characteristics of public-key algorithms, the entire message data is typically not itself transformed directly with the private key. Instead a small unique thumbprint of the document, called a "hash" or "digest", is transformed. Because the hashing algorithm is very sensitive to any changes in the source document, the hash of the original allows a recipient to verify that the document was not altered (by comparing the hash that was sent to them with the hash they calculate from the document they received). Additionally, by transforming the hash with their private key, the sender also allows the recipient to verify that it was indeed the sender that performed the transformation (because the recipient was able to use the sender's public key to "undo" the transformation). The hash of a document, transformed with the sender's private key, thereby acts as a digital signature for that document and can be transmitted openly along with the document to the recipient. The recipient verifies the signature by taking a hash of the message and inputting it to a verification algorithm along with the signature that accompanied the message and the sender's public key. If the result is successful, the recipient can be confident of both the authenticity and integrity of the message.

XML Digital Signature

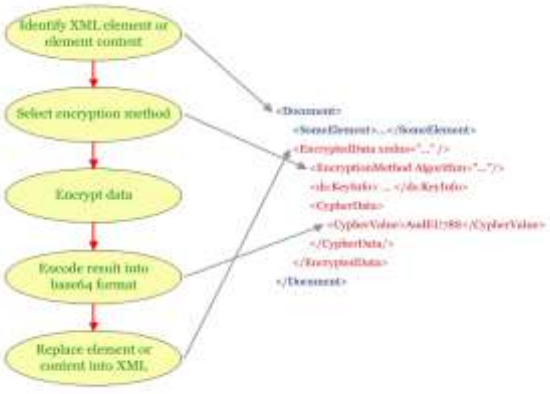

Figure 2: Flow of XML Digital Signature

The very features that make XML so powerful for business transactions (e.g., semantically rich and structured data, text-based, and Web-ready nature) provide both challenges and opportunities for the application of encryption and digital signature operations to XML- encoded data. For example, in many workflow scenarios where an XML document flows stepwise between participants, and where a digital signature implies some sort of commitment or assertion, each participant may wish to sign only that portion for which they are responsible and assume a concomitant level of liability. Older standards for digital signatures provide neither syntax for capturing this sort of high-granularity signature nor mechanisms for expressing which portion a principal wishes to sign [2].

Two new security initiatives designed to both account for and take advantage of the special nature of XML data are XML Signature and XML Encryption. Both are currently progressing through the standardization process. XML Signature is a joint effort between the World Wide Web Consortium (W3C) and Internet Engineering Task Force (IETF), and XML Encryption is solely W3C effort [2].

XML signature is form of digital signature designed for use in XML transactions. The XML Digital Signature standard defines a schema that is used for storing the result of a digital signature operation applied to (in most cases) XML data [3]. Like non-XML digital signatures, XML signatures add authentication, data integrity, and support for non-repudiation to the data that is object of XML digital signing process. However, unlike non-XML digital signature standards, XML signature has been designed to both account for and take advantage of the Internet and XML.

A fundamental feature of XML Signature is the ability to sign only specific portions of the XML content rather than the complete document. This is relevant when a single XML document may have a long history in which the different components are authored at different times by different parties, each signing only those elements relevant to it. This flexibility will also be critical in situations where it is important to ensure the integrity of certain portions of an XML document, while leaving open the possibility for other portions of the document to change. Since data security – in form of data verification and authorization - represents an important part of information system security paradigm this article is addressing the questions and possibilities of XML Digital Signature usage in everyday information system security procedures.

4. INTRODUCTION TO XML

XML signatures are digital signatures designed for use in XML transactions. The standard defines a schema for capturing the result of a digital signature operation applied to arbitrary (but often XML) data. Like non-XML-aware digital signatures (e.g., PKCS), XML signatures add authentication, data integrity, and support for non-repudiation to the data that they sign. However, unlike non-XML digital signature standards, XML signature has been designed to both account for and take advantage of the Internet and XML [4].

A fundamental feature of XML Signature is the ability to sign only specific portions of the XML tree rather than

the complete document. This will be relevant when a single XML document may have a long history in which the different components are authored at different times by different parties, each signing only those elements relevant to itself. This flexibility will also be critical in situations where it is important to ensure the integrity of certain portions of an XML document, while leaving open the possibility for other portions of the document to change. Consider, for example, a signed XML form delivered to a user for completion. If the signature were over the full XML form, any change by the user to the default form values would invalidate the original signature.

An XML signature can sign more than one type of resource. For example, a single XML signature might cover character-encoded data (HTML), binary-encoded data (a JPG), XML-encoded data, and a specific section of an XML file.

Signature validation requires that the data object that was signed be accessible. The XML signature itself will generally indicate the location of the original signed object. This reference can

- be referenced by a URI within the XML signature;
- reside within the same resource as the XML signature (the signature is a sibling);
- be embedded within the XML signature (the signature is the parent);
- have its XML signature embedded within itself (the signature is the child).

THE COMPONENTS OF AN XML SIGNATURE

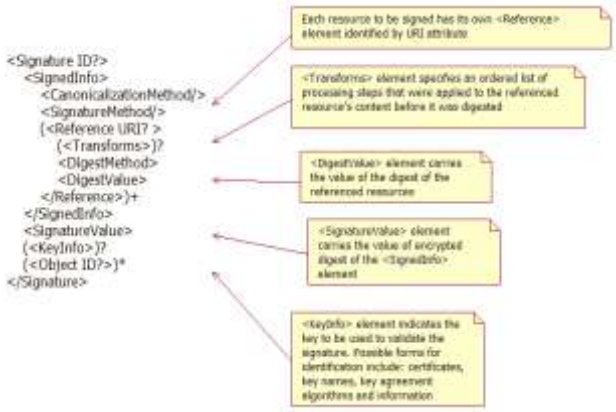

Figure 3: Format of XML Digital Signature

How to Create an XML Signature

Here is a quick overview of how to create an XML signature; please refer to the XML Signature specification for additional information.

1. Determine which resources are to be signed.

This will take the form of identifying the resources through a Uniform Resource Identifier (URI).

- "http://www.abccompany.com/index.html" would reference an HTML page on the Web
- "http://www.abccompany.com/logo.gif" would reference a GIF image on the Web
- "http://www.abccompany.com/xml/po.xml" would reference an XML file on the Web
- "http://www.abccompany.com/xml/po.xml#sender1" would reference a specific element in an XML file on the Web

2. Calculate the digest of each resource.

In XML signatures, each referenced resource is specified through a *<Reference>* element and its digest

```
<Reference URI="http://www.abccompany.com/news/2000/03_27_00.htm">
 <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
 <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
</Reference>
<Reference
 URI="http://www.w3.org/TR/2000/WD-xmldsig-core-20000228/signature-example.xml">
 <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
 <DigestValue>UrXLDLBIta6skoV5/A8Q38GEw44=</DigestValue>
</Reference>
```

Figure 4: Calculate the digest of each resource.

(calculated on the identified resource and not the *<Reference>* element itself) is placed in a *<DigestValue>* child element like:

The *<DigestMethod>* element identifies the algorithm used to calculate the digest.

3. Collect the Reference elements

Collect the *<Reference>* elements (with their associated digests) within a *<SignedInfo>* element like

```
SignedInfo Id="foobar">
 <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.abccompany.com/news/2000/03_27_00.htm">
  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
  <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
 </Reference>
 <Reference
    URI="http://www.w3.org/TR/2000/WD-xmldsig-core-20000228/signature-example.xml">
  <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
  <DigestValue>UrXLDLBIta6skoV5/A8Q38GEw44=</DigestValue>
 </Reference>
</SignedInfo>
```

Figure 5: Collect the Reference elements

The <*CanonicalizationMethod*> element indicates the algorithm was used to canonize the <*SignedInfo*> element. Different data streams with the same XML information set may have different textual representations, e.g. differing as to whitespace. To help prevent inaccurate verification results, XML information sets must first be canonized before extracting their bit representation for signature processing. The <*SignatureMethod*> element identifies the algorithm used to produce the signature value.

4. Signing

Calculate the digest of the <SignedInfo> element, sign that digest and put the signature value in a <SignatureValue> element.

<SignatureValue>MC0E LE=</SignatureValue>

5. Add key information

If keying information is to be included, place it in a <*KeyInfo*> element. Here the keying information contains the X.509 certificate for the sender, which would include the public key needed for signature verification.

```
<KeyInfo>
 <X509Data>
  <X509SubjectName>CN=Ed Simon,O=XMLSec Inc.,ST=OTTAWA,C=CA</X509SubjectName>
  <X509Certificate>MIID5jCCA0+gA...lVN</X509Certificate>
 </X509Data>
</KeyInfo>
```

Figure 6: Add key information

6. Enclose in a Signature element

Place the <*SignedInfo*>, <*SignatureValue*>, and <*KeyInfo*> elements into a <*Signature*> element. The <*Signature*> element comprises the XML signature.

<?xml version="1.0" encoding="UTF-8"?>

<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

<SignedInfo Id="foobar">

<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.abccompany.com/news/2000/03_27_00.htm">

<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />

<DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>

</Reference>

<Reference

URI="http://www.w3.org/TR/2000/WD-xmldsig-core-20000228/signature-example.xml">

<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

<DigestValue>UrXLDLBIta6skoV5/A8Q38GEw44=</DigestValue>

</Reference>

</SignedInfo>

<SignatureValue>MC0E~LE=</SignatureValue>

<KeyInfo>

<X509Data>

<X509SubjectName>CN=Ed Simon,O=XMLSec Inc.,ST=OTTAWA,C=CA</X509SubjectName>

<X509Certificate>

MIID5jCCA0+gA...lVN

</X509Certificate>

</X509Data>

</KeyInfo>

</Signature>

Verifying an XML Signature

A brief description of how to verify an XML signature:
Verify the signature of the <*SignedInfo*> element. To do so, recalculate the digest of the <*SignedInfo*> element (using the digest algorithm specified in the <*SignatureMethod*> element) and use the public verification key to verify that the value of the <*SignatureValue*> element is correct for the digest of the <*SignedInfo*> element.
If this step passes, recalculate the digests of the references contained within the <*SignedInfo*> element and compare them to the digest values expressed in each <Reference> element's corresponding <*DigestValue*> element.

5. CONCLUSION

Digital signatures can help build more secure and efficient Web applications across business segments. As XML becomes a vital component of the emerging electronic business infrastructure, we need trustable, secure XML messages to form the basis of business transactions. One key to enabling secure transactions is the concept of a digital signature, ensuring the integrity and authenticity of origin for business documents. XML Signature is an evolving standard for digital signatures that both addresses the special issues and requirements that XML presents for signing operations and uses XML syntax for capturing the result, simplifying its integration into XML applications.

REFERENCES

[1] Goswami, S.; Misra, S.; Mukesh, M., "A PKI based timestamped secure signing tool for e-documents," High Performance Computing and Applications (ICHPCA), 2014 International Conference on , vol., no., pp.1,6, 22-24 Dec. 2014

[2] An Introduction to XML Digital Signatures [online]. Available : https://www.xml.com/pub/a/2001/08/08/xmldsig.html

[3] Ponnapalli, Harigopal K.B.; Saxena, Ashutosh, "A Digital Signature Architecture for Web Apps," IT Professional , vol.15, no.2, pp.42,49, March-April 2013.

[4] Geric, S.; Vidacic, T., "XML digital signature and its role in information system security,"MIPRO, 2012 Proceedings of the 35th International Convention, vol., no., pp.1520,1525, 21-25 May 2012.