XML ELECTRONIC SIGNATURES

Application according to the international standard XML Signature Syntax and Processing

DI Gregor Karlinger

Graz University of Technology

Institute for Applied Information Processing and Communication Technologies

Inffeldgasse 16a, 8010 Graz, Austria

Gregor.Karlinger@IAIK.at

Abstract The deployment of electronic signatures becomes vital as an increasing part of business transactions are handled via electronic channels such as the World Wide Web because they facilitate integrity, signature assurance and non-repudiation over electronic data. Extensible Markup Language (XML) has been widely accepted as a generic language for designing electronic documents used to interchange data between different applications. In June 1999 the World Wide Web Consortium (W3C) has established the XML-Signature working group together with the Internet Engineering Task Force (IETF). Its proposed work will address the digital signing of documents using XML syntax. By Fall 2000 the standard draft "XML-Signature Syntax and Processing" reached its Candidate Recommendation stage appearing as a rather stable specification. This paper presents selected topics treated by the draft I consider to be of interest for people familiar with other signature protocols, such as Cryptographic Message Syntax, but have limited knowledge of the XML domain.

Keywords: XML, electronic signature, standardization

1. INTRODUCTION

The Extensible Markup Language (XML) [Bray et al., 2000] is a set of rules for designing text formats for structured documents and data on the Web. Instance documents both can be easily parsed automatically and read by humans. Although the standard specification only exists since early 1998, it has been widely accepted and is used in various fields of current web technology.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: [10.1007/978-0-387-35413-2_36](https://doi.org/10.1007/978-0-387-35413-2_36)

R. Steinmetz et al. (eds.), *Communications and Multimedia Security Issues of the New Century*

© IFIP International Federation for Information Processing 2001

As XML is intended to be used in lots electronic transactions via the internet, standard mechanisms for applying electronic signatures on such XML data are urgently needed. Therefore in June 1999 the W3C together with the IETF established the *XML-Signature* working group, whose objective has been stated to develop a standard for both the structure representing XML signatures and the necessary processing steps for signing and verifying a XML document instance.

The current state of the working group's efforts is represented by the October 2000 working draft *XML-Signature Syntax and Processing* [Eastlake et al., 2000], which has reached the W3C *Candidate Recommendation* stage. Therefore the draft can be seen as a rather stable document.

In this paper I introduce the main ideas, which have formed the starting point for the working group activity, and their consideration in the current draft. Lots of requirements for the signature standard stem from the XML application domain. Therefore I will try to focus on questions which are likely to be asked by people being familiar with other mechanisms for signing electronic documents, such as *Cryptographic Message Syntax (CMS)* [Housley, 1999], but have little knowledge regarding XML. Additionally I will make some annotations resulting from my experience as an early implementor of the *XML-Signature* draft.

2. SIGNATURE TYPES

A couple of basic requirements have been defined for *XML-Signature*, regarding the location of the XML signature with respect to the document which is to be signed: Signatures can be applied to both XML and non-XML documents; signatures can reside in the same XML document as the content to be signed or can be sourced out into an extra XML document; the content which should be signed can also be filed at a certain place inside the XML Signature element.

XML-Signature introduces three notations to express the relationship between the XML Signature element and the content which is to be signed:

Detached Signature : The signature is over content external to the XML Signature element. This definition typically applies to data items residing in a place different from the XML document containing the Signature element. But it also includes the instance where the Signature element and a data item reside within the same XML document but are sibling XML elements.

A characteristic use case for a *Detached Signature* is the signing of a non-XML document: A new XML document with the Signature

element as its root contains a reference to the data item which is located in a different place.

Enveloped Signature : The signature is over content that contains the XML *Signature* element as an offspring. Obviously, the creator of an *Enveloped Signature* must take care not to include the XML encoding of the digest values and the signature value of the signature itself into the calculation process. A means to achieve this will be introduced in chapter 5.

This type of signature applies for instance, if a whole XML document should be signed and the XML *Signature* element is to be inserted at a predefined place inside this document.

Enveloping Signature : The signature is over content found within an XML *Object* element in the *Signature* itself. The *Object* element is a container specified in *XML-Signature* which can hold an arbitrary number of both XML and non-XML data items.

A typical use case for this signature type is the signing of a couple of small data items which do not reside in a document of their own and are therefore incorporated into a new XML document using the *Signature* element as its root.

Please note that these notations are not mutual exclusive since a *Signature* element can contain an arbitrary number of references to different data items. For example it can hold a reference to an external document and a reference to a data entity residing inside a *Object* container element. Such a signature can be characterized as detached and enveloping at the same time. So, strictly speaking, the three types of signatures reflect the relationship between one certain reference of an XML signature and the location of corresponding data entity.

3. SEVERAL SIGNERS AND DATA ITEMS

A couple of principal questions on the possibilities of *XML-Signature* concern the relationship between a signature instance (XML *Signature* element) on the one hand and the number of possible signing subjects (signers) and signing objects (data items to be signed) on the other hand.

Firstly, have a look at the number of signers per signature instance. This relationship is modelled as a one-to-one mapping, i. e. one signature instance must be constructed per signer. In order to support several subjects signing the same signing objects, a XML document instance will be needed which contains one *Signature* element per signer. Each of these elements maintains the same couple of references pointing to the data items to be signed by all subjects.

In opposition to the relationship explained above, the one between signature instance and signing object is modelled as a one-to-many mapping. This means that a **Signature** element can contain an arbitrary number of references to signed data items.

4. INCLUDING DATA FOR SIGNING

The cryptographic value included in an XML signature is computed in a two-step fashion:

- At first, for each data item to be signed, a **XML Reference** element will be inserted into the XML signature. Such a **Reference** element contains amongst other parts an identifier for the data item, and a message digest over the data item.
- In the second step, the cryptographic signature will - simply speaking - be computed over the XML encoding of all included **Reference** elements.

As already mentioned above, one must provide an identifier for each data item to be signed by the XML signature. This must be done using *Unified Resource Identifiers (URIs)* [Berners-Lee et al., 1998].

The most popular subset of URIs is the *Unified Resource Locator (URL)* [Berners-Lee et al., 1998], which is used to identify a non-ambiguous resource on the network. Additionally, an URI allows to refer to a resource which is assigned a unique name (*Unified Resource Name - URN* [Moats, 1997]). In this case the application must know how to obtain the tagged resource.

An empty value ("") for the URI indicates a reference to the document as a whole containing the **Signature** element. Please remember in this case, that the actual range of the XML document to be signed has to be restricted to avoid a self-signing of the signature.

Finally, the *IDREF* mechanism introduced in the XML 1.0 specification [Bray et al., 2000] is supported by using a certain subset of a URI: If a URI is *reference-only* (for example "#FirstDataItem"), it tags a XML element residing in the same document as the signature, which bears an **ID** attribute having the value specified in the reference part of the URI (in the example above: "FirstDataItem").

5. SHAPING DATA BEFORE SIGNING IT

XML-Signature provides the concept of *Transforms* for addressing the issue of manipulating a data item before actually computing the digest value over it. The request for such a facility comes from various application domains. The original document can be manipulated

in a reproducible way and therefore need not be stored explicitly for dereferencing it later.

Transforms works as adaptor between fetching the original data item and computing its message digest. The following steps describe the basic idea:

- Fetch the original data referred to by the Reference's URI attribute.
- Apply each transform in specified by a list being part of the Reference element. The original data forms the input for the first transform, the output of the first transform is used as the input for the second transform, and so on.
- The output of the last transform (or the original data if there are no transforms specified) is now used as the input for the digest computation.

Since the list of transforms is represented in the signature's XML structure as part of the Reference clause, it is covered by the signature value. Nevertheless the list is intended to be seen only as a hint how the signer has finally obtained the input for the message digest computation. The verifier is not forced to do exactly the same processing. As long as he knows a way to obtain the final data forming the input for the cryptographic processing, signature validation will work correctly.

Please note that the concept of *Transforms* does not impose any security risks here because the result of the last transform finally forms the input for the digest computation. The transform manipulations must be seen as auxiliary means which can be used to shape the actual input for the cryptographic processing.

As an important example for a transform, I would like to mention the *Enveloped Signature Transform* specified in the *XML-Signature* standard document. One can use this transform to avoid the self-signing in case of an enveloped signature (see section 2). It simply cuts off the XML Signature element from the referenced data item. Another important transform will be introduced in the following section.

6. PARTLY SIGNED XML DOCUMENTS

Two means specified in *XML-Signature* to provide for selecting parts of an XML document for signing; a shortcut mechanism and a powerful extended mechanism.

The shortcut mechanism uses the *IDREF* feature introduced in the XML 1.0 specification. A so called *ID attribute* can be assigned to an

XML element, whose value must be unique in the scope of the whole XML document bearing the element. Therefore the XML element can be identified using the value of its *ID attribute*. Now, in *XML-Signature* the fragment identifier part of a URI can be used to hold the value of an *ID attribute*. For example, a reference URI with value "#FirstDataItem" refers to that XML element in the XML document bearing the XML signature, whose *ID attribute* has the value "FirstDataItem".

The extended mechanism for selecting parts of a XML document to be signed uses a *XPath Transform*. *XPath* [Clark and DeRose, 1999] is a language for addressing parts of an XML document and has been published as a W3C Recommendation. *XML-Signature* makes use of this language to address the requirement for precisely selecting certain parts of a XML document for signing. An XPath transform can be added to the list of the Reference's transforms to select (or filter) parts of the original data.

7. CANONICALIZATION

Digital signatures only work, if the verification calculations are performed on exactly the same bits as the signing calculations. The surface representation of signed XML data can change between signing and verification if XML parsers are used, which is very likely to happen.

Firstly, there is some surface information which gets lost when reading the information provided by an XML document, conforming to the specification of XML 1.0. For example, line endings are normalized, character references are replaced with the corresponding character, or entity references are replaced with the corresponding declared entity.

Additional surface information is filtered out because most XML parsers use either the *Document Object Model (DOM)* [Le Hors et al., 2000] or the *Simple API for XML (SAX)* [Megginson, 2000] to report the XML information to the relying application. DOM maps XML into a tree structure of nodes. SAX converts XML into a series of events such as starting element tag, text, closing element tag, etc. In either case many surface characteristics such as insignificant white space within start/end tags is lost. In addition namespace declarations are mapped over the nodes to which they apply, losing the namespace prefixes in the XML source code.

Finally there is the possibility of character set conversion, such as between UTF-8 and UTF-16, both of which all XML standards compliant processors are required to support.

For all that reasons, canonicalization algorithms are introduced in *XML-Signature*. Since surface changes as described above are very likely

to occur, both signer and verifier must employ such an algorithm. The problem can get evident in two different areas within *XML-Signature*: Signing the XML signature's `SignedInfo` element (containing the *Reference* elements) on the one hand, and digesting a data item of type XML.

When signing a data item of type XML, one can treat the XML document as if it were binary data so that no changes can occur between signing and verification. Only if the XML is processed using standard XML parser functionality, for instance due to an intended XPath transform prior to message digest calculation, a final transform achieving canonicalization must be employed to get a common surface representation for the XML fed into the digest computation.

Signing the byte representation of `SignedInfo` most likely faces the canonicalization problem since the XML Signature structure needs to be parsed in order to execute the verification process. Hence a special place is introduced within the `SignedInfo` element for specifying an obligatory canonicalization algorithm, which is executed prior to signature calculation.

8. CRYPTOGRAPHIC KEY INFORMATION

XML-Signature does not specify any rules how key management has to be treated. Issues such as key distribution and trust management are beyond the scope of that specification.

Nevertheless it provides a specific place inside the `Signature` structure where the signer can provide information about the cryptographic key used for generating the signature. Some simple types such as `KeyName` and `X509Data` are available, but applications can also put their own key identification and exchange semantics in there. Future standardization work in this area is supposed to take place.

It is important to see that this place, the `KeyInfo` element, is intended as hint for the signature verifier how to obtain the cryptographic key needed to validate the signature, and not as evidence carrier which the verifier must accept.

9. EXTENSIBILITY

The *XML-Signature* specification uses a generic algorithm concept to address the integration of arbitrary methods for signature and message digest computation, for canonicalization and for specifying transforms.

An algorithm is determined by a unique URI. Depending on the area it operates, it has some implicit and additionally some explicit input parameters. The algorithm gets its implicit parameters from the embedded

context. The explicit parameters are specified as child elements of the algorithm XML element. Finally an algorithm provides a single output parameter which forms the result of the computation.

Algorithms can be split up into four main categories, as there are signature, message digest, canonicalization and transform algorithms. They first three types must be present in the signature's XML structure as elements named `SignatureMethod`, `DigestMethod` and `CanonicalizationMethod`, while the latter can be employed as `Transform` elements inside a `Reference` clause.

For each category, *XML-Signature* defines some basic algorithms which must be supported by all implementations. Additionally an application can plug in any other arbitrary algorithms it wants to use.

10. CONCLUSIONS

Although the current *XML-Signature* draft has not yet reached its final recommendation status, and will therefore undergo certain minor changes, one can assume that the main ideas, as they have been described in the previous sections of this paper will survive. Thus, I will conclude with a couple of annotations resulting from my experience as a member of XML-Signature working group and from the insights I got when designing and implementing a JAVA toolkit conforming to this standard.

There has been lots of discussion in the working group coming from the different views on the matter from the cryptographic versus the XML application domain. XML-Signature provides powerful means for referencing arbitrary data on the web and for navigating into such documents if they are XML instances. This functionality must be seen strictly separated from cryptographic aspects. It can be used to shape the actual input for the signature process, but does not mean any softening of the security.

The specification addresses the multifarious requirements regarding the location of a XML Signature with respect to the data item(s) it secures, with a very generic concept. The possible Signature constructions have not been restricted to some common use cases, but almost any thinkable combination of enveloped, enveloping and detached signatures in a collective Signature clause is allowed. Due to this freedom a conforming implementation must be aware of its responsibility not to allow pathological signature use cases such as the self-signing of a whole Signature clause as mentioned in a previous section.

Finally, I would like to stress, that *XML-Signature* provides only basic structures and processing information for creating and verifying XML-

based electronic signatures. Additional standardization work is necessary in areas that are beyond the scope of this specification, such as exact rules for providing certain kinds of key information, or the definition of attributes which are qualifying XML-based electronic signatures. In the latter fields, there are already ongoing efforts within the *European Telecommunications Standards Institute (ETSI)* regarding the attributes necessary for a technical implementation of the EU directive on electronic signatures.

References

- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U., and Masinter, L. (1998). Uniform resource identifiers (uri): Generic syntax. RFC 2396. Retrieved from the World Wide Web on 1. February, 2001: <http://www.ietf.org/rfc/rfc2396.txt>.
- [Bray et al., 2000] Bray, T., Paoli, J., Sperberg-McQueen, C. M., and Maler, E. (2000). Extensible markup language 1.0 (second edition). W3C Recommendation. Retrieved from the World Wide Web on 1. February, 2001: <http://www.w3.org/TR/2000/REC-xml-20001006>.
- [Clark and DeRose, 1999] Clark, J. and DeRose, S. (1999). Xml path language version 1.0. W3C Recommendation. Retrieved from the World Wide Web on 1. February, 2001: <http://www.w3.org/TR/1999/REC-xpath-19991116>.
- [Eastlake et al., 2000] Eastlake, D., Reagle, J., and Solo, D. (2000). Xml signature syntax and processing. W3C Candidate Recommendation. Retrieved from the World Wide Web on 1. February, 2001: <http://www.w3.org/TR/2000/CR-xmlsig-core-20001031/>.
- [Housley, 1999] Housley, R. (1999). Rfc 2630: Cryptographic message syntax. IETF Request For Comment. Retrieved from the World Wide Web on 1. February, 2001: <http://www.ietf.org/rfc/rfc2630.txt>.
- [Le Hors et al., 2000] Le Hors, A., Wood, L., and Le Hegaret, P. (2000). Document object model (dom) level 2 core specification. W3C Recommendation. Retrieved from the World Wide Web on 1. February, 2001: <http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/>.
- [Megginson, 2000] Megginson, D. (2000). Sax: The simple api for xml. Retrieved from the World Wide Web on 1. February, 2001: <http://www.megginson.com/SAX/index.html>.
- [Moats, 1997] Moats, R. (1997). Urn syntax. RFC 2141. Retrieved from the World Wide Web on 1. February, 2001: <http://www.ietf.org/rfc/rfc2141.txt>.