XML DIGITAL SIGNATURE AND RDF

Russell Cloran and Barry Irwin

Rhodes University

{R.Cloran,B.Irwin}@ru.ac.za

ABSTRACT

The XML Signature working group focuses on the canonicalisation of XML, and the syntax used to sign an XML document. This process focuses on the semantics introduced by the XML language itself, but ignores semantics which a particular application of XML may add.

The Resource Description Framework (RDF) is a language for representing information about resources on the Web. RDF has a number of possible serialisations, including an XML serialisation (RDF/XML), popularly used as the format for exchanging RDF data. In general, the order of statements in RDF is not important, and thus the order in which XML tags occur in RDF/XML can vary greatly whilst still preserving semantics.

This paper examines some of the issues surrounding the canonicalisation of RDF/XML and the signing of it, discussing nesting, node identifiers and the ordering of nodes. Existing RDF serialisation formats are considered as case studies of partially canonical RDF formats.

XML DIGITAL SIGNATURE AND RDF

1 INTRODUCTION

On an increasingly hostile Internet, document exchange is vulnerable to interception and tampering. Documents exchanged over the Internet should therefore be verified for integrity and authenticity through the use of strong cryptographic methods such as the increasingly common notion of digital signatures, which provides a means of doing this. The XML Signature working group[1] endeavours to "develop an XML compliant syntax for representing the signature of Web resources and portions of protocol messages (anything referencable by a URI)"[1]. This includes signing portions of (or whole) XML documents, and their work includes work on canonical XML[2]. The Resource Description Framework (RDF)[3] is a core technology in the emerging Semantic Web[4], which will enable different forms of distributed processing, e-commerce, and more powerful and pervasive agents on personal devices. It is thus foreseeable that a large number of users will want to sign and verify signatures of RDF data in the near future. Only a small amount of work has been done on signing RDF documents and RDF graphs. We investigate some of this work, beginning with an introduction to RDF, XML digital signatures, and an overview of the problems we encounter when trying to sign RDF documents.

This is followed by some of the concepts needed in order to successfully sign RDF. We then go on to examine some work which solves the problem of RDF signatures either partially or completely, whether by design or coincidence.

1.1 Resource Description Framework

RDF "is a language for representing information about resources in the World Wide Web"[5]. RDF is designed to have a very simple data model, which makes adoption of the standard attractive to developers, and accessible to end users. In RDF resources being described have a set of properties, each of which has a value. These values can either be literal values or another resource. Conceptually, resources (or literal values) can be regarded as nodes, and the properties as edges. This data model leads to a directed, labelled graph. This model can also be thought of as a set of individual statements, each of which have a subject, a predicate (property) and an object (value). Figure 1 shows the relationship between the set of statements, and the labelled graph which they describe.

1.2 RDF/XML

The XML serialisation of RDF[6] provides a syntax for RDF based on the popular, robust, internationalised language, XML. RDF uses Uniform Resource Identifiers[7] (URIs) to identify resources. These are encoded using the XML namespace rules[8]. These rules use QNames (qualified names) as the XML tag, and each QName consists of a prefix (the XML namespace) and a local part. The local part is appended to the URI of the XML namespace to obtain the full RDF URI reference. The RDF/XML syntax is often called a "striped syntax"[9, 5] because

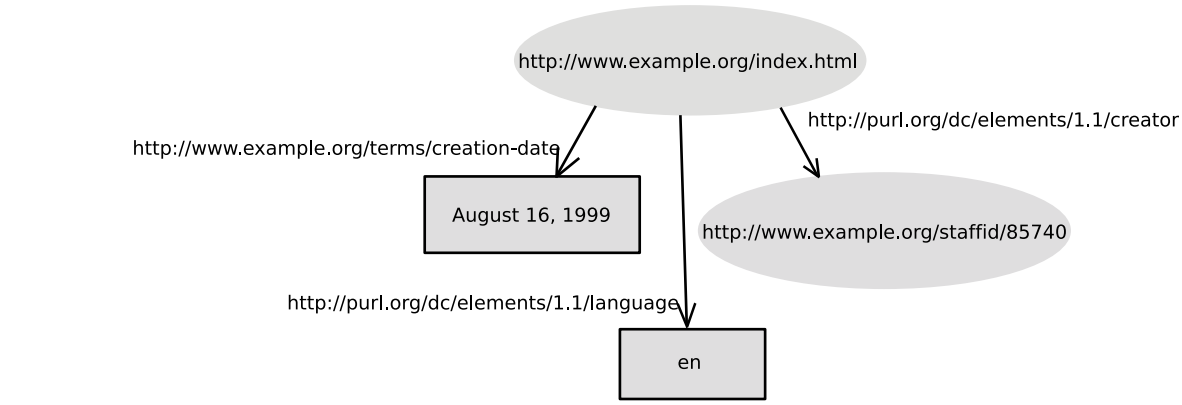

Figure 1: A set of RDF triples map naturally to a labelled, directed graph[5]

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/rdf-syntax-ns#"
     xmlns="http://example.com/some-dlg-schema#">

1:<Person>
2:    <name> John  </name>
2:    <livesWith>
3:       <Person>
4:          <father>
5:             <Person>
6:                <name> Fred </name>
5:             </Person>
4:          </father>
3:       </Person>
2:    </livesWith>
1:</Person>

</rdf:RDF>
```

Figure 2: RDF striped syntax[9]

when reading the serialisation, tags are encountered in an alternating pattern between the nodes and arcs. The striping concept is illustrated in Figure 2.

The logical organisation of an XML document is a tree format. This encoding therefore creates a target tree structure for the RDF serialisation tool. In order to map a graph into a tree in all cases, it is necessary to create links between nodes within the document. Such links may be created either by named blank nodes, or by internal document references, using the RDF ID property (an XML Schema ID, similar to the ID attribute in (X)HTML). Because a graph can be mapped into a tree in a number of different ways, and because the order in which statements appear in the RDF/XML document is not always important, serialising an RDF graph into RDF/XML may lead to a number of different, yet valid, serialisations.

RDF/XML allows linking between nodes at arbitrary places of the tree, and an RDF/XML document therefore often resembles the graph concept of the RDF model more closely than it does the set of individual statements to the human reader. This is also true because for each

resource, its properties may be listed without stating what the resource is a second time, allowing a number of statements with the same subject to be grouped into one part of the document. This freedom of expression is the cause of some of the problems which occur when trying to sign RDF/XML documents, as the XML processor has no concept that the variations in the XML documents convey the same meaning for the application consuming the RDF parser.

1.3 XML Digital Signature

XML Digital Signature[1] is a W3C recommendation, and Internet draft standard (RFC), which provides a way to sign arbitrary digital information, and represent this signature in XML. XML digital signature was designed to allow any data which can be referenced as a URL to be signed, and it thus allows the signing of XML sub-documents through document fragment identifiers.

The process of generating an XML signature follows the steps described in [1]:

1. Generate the reference
   (a) Apply the transforms, as determined by the application, to the data object
   (b) Calculate the digest value over the resulting data object
   (c) Create a REFERENCE element, including the (optional) identification of the data object, any (optional) transform elements, the digest algorithm and the DIGESTVALUE.
2. Generate the signature
   (a) Create SIGNEDINFO element with SIGNATUREMETHOD, CANONICALIZATIONMETHOD and REFERENCE(s)
   (b) Canonicalize and then calculate the SIGNATUREVALUE over SIGNEDINFO based on algorithms specified in SIGNEDINFO
   (c) Construct the SIGNATURE element that includes SIGNEDINFO, OBJECT(s) (if desired, encoding may be different than that used for signing), KEYINFO (if required), and SIGNATUREVALUE

We see here that XML signature provides opportunity for the application (XML document containing RDF data in our case) to create a canonical version of its data prior to signature calculation.

1.4 XML Canonicalisation

Something which is canonical has been "reduced to the simplest and most significant form possible without loss of generality" or is "conforming to the orthodox or recognised rules"[10]. XML Canonicalisation "refers to the process of applying the XML canonicalisation method to an XML document or document subset"[2].

XML Canonicalisation allows two XML documents to be compared for identity in a bitwise manner, after they have both been transformed into the canonical form. This is the crucial first step in allowing any type of digital signatures (PGP is another example which is distinct from XML digital signatures) to be applied across XML documents, as a signing operation requires that the initial input and the input to be checked are identical.

The basic process of XML canonicalisation can be broken down into 4 steps, as described in [2]:

1. Normalize line feeds
2. Normalize attribute values
3. Replace CDATA sections with their character content
4. Resolve character and parsed entity references

Notably, the process of creating a canonical XML document does not explicitly mention about a canonical document form. It should be assumed that canonical XML should be generated from a canonical document form, if more than one exists.

1.5 Signed RDF

Canonical XML only addresses the semantics introduced by XML and not the application which is using it. RDF/XML introduces semantics which mean that a number of different XML documents can represent the same RDF model, due to the flexibility of the RDF/XML serialisation standard. Some of the factors in RDF/XML that cause this to happen are:

- Typed nodes are optional (see Section 2.3)
- An RDF graph may be broken into a tree in a number of ways (see Section 1.2)
- Order is not always important (with some exceptions, such as the `SEQ` property)

The consequence of this is that without some means to create a canonical RDF/XML document from any model, RDF documents cannot easily be compared for identity unless some means of a canonical serialisation is created.

2 CANONICAL RDF

2.1 Levels of semantics

In "Towards the Semantic Web" Broekstren, Kampman and van Hermelen[11] suggest that RDF may be queried at three levels: the syntax, structural and semantic levels. We propose that these basic distinctions may also be applied to the process of creating a canonical RDF document. These are discussed below in a top down fashion.

2.1.1 Semantic level

The semantic level is the highest level of understanding that the application has, and the model is divorced furthest from its physical representation. At this level RDF is considered as one or more labelled graphs, with some predefined semantics. At this level, issues such as blank node identifiers are not considered, but we consider rather the structure of the graph.

Two graphs are said to be isomorphic if there is a one-to-one correspondence between their vertices, and a one-to-one correspondence between the edges joining corresponding vertices. Two RDF graphs can therefore said to be equivalent if they are found to be isomorphic. Following this, testing RDF graph equality and graph isomorphism have equivalent complexity, as discussed by Carroll in [12].

Testing for graph isomorphism is a problem for which no polynomial time solution has been found, nor has it been proven that it is NP-complete[13]. Carroll[14] notes that it is therefore theoretically possible to verify the signature of an arbitrary RDF graph in polynomial time. Carroll's methods for improving on this are discussed in section 2.2.

At this level we refer to the RDF data as a model or a graph, as compared to the syntactic level, where we refer to the serialised format as an RDF document.

2.1.2 Structural level

At the structural level, RDF is considered as a set of triples. The semantics of the graph are broken down into a series of individual node-arc-node relationships. Other than the knowledge that these statements are part of the same RDF model, they are unrelated. At this level, the problem of blank node identifiers arises, as making a link from a labelled node to an unlabelled node and then from the unlabelled node to another node needs to have some sort of link. This link is created using blank node identifiers. The canonicalisation method described by Carroll in [14] operates chiefly at the structural level, although the specific implementation described relies on the N-Triples serialisation described in the RDF Test Cases[15].

Renaming of blank node identifiers and the ordering of triples for use at the syntactic level are operations which occur at the structural level. The first operation requires some knowledge of the semantic level, namely, when renaming a blank node it is necessary to rename all blank nodes with the same identifier in order to retain the semantics. This particular operation is the most difficult part of any of the processes of creating a canonical RDF model or document, and is essentially part of the process of testing for isomorphism. An example of this process is described in [14], and discussed in Section 2.2. Ordering of the triples is also discussed in [14], but is a relatively straightforward process of lexically ordering the triples that make up the graph by subject, predicate and then object.

2.1.3 Syntactic level

At a syntactical level we consider the serialised RDF. In the case of RDF/XML, this would be the XML document that is produced by serialisation. The flexibility of the RDF/XML means that it is likely that only a subset of the RDF/XML features will be used, as is done in R3X[16], discussed in Section 2.3. Because the result of an RDF/XML serialisation is an XML document,

we should also apply the XML canonicalisation rules described in 1.4 in order to produce a canonical XML document, either as part of the serialisation process, or after it.

The process of signing the document also happens at this level, since some sort of serialisation of the RDF model is required in order to sign it. Efforts such as the WOT vocabulary[17] and existing XML signature solve this problem adequately, if we can obtain a canonical serialisation.

2.2 Carroll's canonicalisation method

The techniques described by Carroll in [14] are designed to obtain a canonical version of an RDF graph. The N-Triples serialisation is used both for processing and the ultimate output format, as described in the paper. The method arguably operates at all three of the levels we consider above. Most of the difficult work is done at the structural level in order to test that two graphs are the same at the semantic level. The general outline of Carroll's algorithm is:

1. Replace blank nodes (in subject and object positions) with a placeholder, storing the original blank node identifier
2. Sort the triples lexically
3. From top to bottom, replace blank nodes in statement subjects with an incrementing blank node identifier, ensuring all instances of the same blank node get replaced with the same new identifier, if the statement (with placeholder blank node) is not the same as the previous or following statement (with placeholder blank node).
4. Repeat the previous step for blank nodes in statement objects

This process is deterministic for a subset of RDF models. A further "precanonicalisation" step is required to ensure that this process works for all RDF models. This step involves inserting statements which add no meaning to the document between carefully selected nodes. This precanonicalisation arguably changes the some of the semantics of the graph.

These algorithms described by Carroll provide a useful step in obtaining a canonical RDF model, although a change from the N-Triples syntax is required in order to harness the power and interoperability introduced by the XML language serialisation of RDF.

2.3 R3X

Redland Restricted RDF/XML (R3X) is a subset of the RDF/XML serialisation created by Morten Frederiksen. There are currently two implementations of R3X: one as PHP code which output R3X directly from a stream in the Redland framework, and another as an XSL transform which takes as input an RDF/XML document which is output by the default Redland RDF/XML serialisation component. R3X is, in fact, less restricted than the default output of the Redland library's serialisation library, which only uses a very limited subset of the productions available in RDF/XML. R3X was originally designed as a means of simplifying XSL stylesheets which converted RDF to HTML for user display, and so is not entirely designed for the goal of obtaining a canonical RDF document. It is an interesting observation nonetheless, as it shows some of the key ideas which may be useful in creating a canonical RDF document.

When it is considered that R3X is a serialisation procedure, it is notable that it is a bridge between the structural and syntactic levels described in Section 2.1. It does not operate at one of the levels, but rather provides the transition from the higher level to the lower.

Observing the output of R3X provides some insight to the production of an RDF/XML document which Carroll's method doesn't, since Carroll's method is not concerned with RDF/XML at all.

Firstly, it is observed that grouping by subject (obtained automatically by ordering lexically sorting statements) drastically shortens the document, and improves human readability, although this should not be considered an important goal. This has a welcome side effect in that statements always occur at the "top level" of the document, eliminating nesting issues.

Secondly, XML namespaces are declared for every node produced in the R3X serialisation. Since the Canonical XML recommendation[2] does not alter namespace declarations, it is important that a canonical RDF serialisation outputs namespace information in a deterministic manner.

Lastly, we note that the R3X output does not in any way order sub-nodes, unless they are part of an RDF `SEQ`. Such a result is expected to occur naturally as a side effect of an operation similar to Carroll's algorithm, or is a trivial step which can be performed prior to serialisation.

3 REAL WORLD

It is interesting for us to note that the use of cryptography together with RDF in the field has been limited to one application which we have seen. We mentioned earlier the WOT (Web of Trust) vocabulary, which uses PGP (Pretty Good Privacy) as a cryptographic basis. WOT allows PGP signing events to be described, including the association of a signature with a document, a person with a signing event, and a signing event with a signature.

The WOT vocabulary has a certain appeal to the "hacker" community, who are often early adopters of new technology. It is available now, and uses the well known and popular technology, PGP. However, we have only noted one use of the WOT vocabulary, Edd Dumbill's FOAFBot[1]. The FOAFBot is an "IRC Community Support Agent", and is designed to provide answers to users of an IRC channel. The FOAFBot also attributes its sources by name, if there is cryptographic proof that the source made the statement.

> <edd> foafbot, edd's name
> <foafbot> edd's name is 'Edd Dumbill', according to Dan Brickley, Anon35, Niel Bornstein, Jo Walsh, Dave Beckett, Edd Dumbill, Matt Biddulph, Paul Ford

Notice "Anon35", a person who has made a statement that there is a person with the IRC nickname "edd" who has the full name "Edd Dumbill", but has not backed this up with a PGP signature. Other users have signed the document in which the statements supporting the fact occurs, and so are attributed by name.

We speculate that the very low usage rate of cryptography together with RDF occurs for three main reasons. Firstly, the Semantic Web at this point in time is reminiscent of the early days of the Internet. Most of the users are researchers who are more interested in experimental results rather than securing their communications. Secondly, the marginal gain from providing signed

[1] http://usefulinc.com/foaf/foafbot

RDF data is low, as not many agents can process either XML Digital Signature or statements as they occur in the WOT vocabulary. We again speculate that a wider uptake of Semantic Web technologies will ensure that cryptography is used where required.

4 CONCLUSION

We conclude with some thoughts on signing RDF data, particularly RDF/XML documents.

A breakdown of the levels of RDF into syntax, structure and semantics is does not give a strictly accurate picture of where a particular technique falls when dealing with canonical RDF, but it gives us a good idea of the sorts of operations that it is performing. Methods which operate at the syntactic level will often have to have access to the structure, and methods which alter the semantic level will change the structure of the model.

Creating canonical RDF can be broadly broken down into two categories: creating a canonical version of the model and creating a canonical serialisation of the model. Creating a canonical version of the model involves operations such as renaming blank nodes, ordering triples, and inserting meaningless statements if necessary, while creating a canonical serialisation of that model involves a normal serialisation procedure, but uses only a subset of the RDF/XML specification.

There are well defined document signing standards such as PGP or even XML Digital Signature which may be used today to sign RDF/XML (or other serialisations of RDF) for assurance at the document level. XML Digital Signature may even allow the signature of components of an RDF document. Informal observation leads us to believe that these standards are not in widespread usage.

Creating a canonical RDF/XML serialisation is possible, and is a necessary step for the success of the Semantic Web.

References

[1] D. Eastlake, J. Reagle, and D. Solo, (eds), "XML-Signature syntax and processing," Tech. Rep. Internet RFC 3275, IETF, Mar. 2002. http://www.ietf.org/rfc/rfc3275.txt.

[2] J. Boyer, "Canonical XML." http://www.w3.org/TR/xml-c14n.

[3] E. Miller, R. Swick, and D. Brickley, (eds), "RDF and RDF Schema," 2003. http://www.w3.org/RDF/.

[4] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, pp. 34–43, May 2001. http://www.sciam.com/2001/0501issue/0501berners-lee.html.

[5] F. Manola and E. Miller, (eds), "RDF primer." W3C Recommendation, 2004. http://www.w3.org/TR/rdf-primer/.

[6] D. Beckett, (ed), "RDF/XML syntax specification (revised)." http://www.w3.org/TR/rdf-syntax-grammar/.

[7] T. Berners-Lee, R. Fielding, and L. Masinter, "Uniform resource identifiers (URI): Generic syntax," Tech. Rep. Internet RFC 2396, IETF, 1998. http://www.ietf.org/rfc/rfc2396.txt.

[8] T. Bray, D. Hollander, and A. Layman, (eds), "Namespaces in XML," Jan. 1999. http://www.w3.org/TR/REC-xml-names.

[9] D. Brickley, "RDF: Understanding the striped RDF/XML syntax," Aug. 2002. http://www.w3.org/2001/10/stripes/.

[10] "WordNet 2.0." http://wordnet.princeton.edu/cgi-bin/webwn2.0?stage=1&word=canonical.

[11] J. Davies, D. Fensel, and F. van Harmelen, eds., *Towards the Semantic Web: Ontology-Driven Knowledge Management*, ch. 5. John Wiley and Sons, 2003.

[12] J. Carroll, "Matching RDF graphs," in *Proceedings of the First International Semantic Web Conference* (I. Horrocks and J. Hendler, eds.), pp. 5–15, 2002.

[13] S. Skiena, *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, pp. 181–187. Addison-Wesley, July 1990.

[14] J. J. Caroll, "Signing RDF graphs," Tech. Rep. HPL-2003-142, Hewlett-Packard, Jul 2003.

[15] J. Grant and D. Beckett, (eds), "RDF Test Cases." W3C Recommendation, 2004. http://www.w3.org/TR/rdf-testcases/.

[16] M. Frederiksen, "Transforming RDF/XML with XSLT." http://www.wasab.dk/morten/blog/archives/2004/05/30/transforming-rdfxml-with-xslt.

[17] D. Brickley, "wot 0.1." http://xmlns.com/wot/0.1/.