A Layered Approach to XML Canonicalization

A Position Paper for the W3C Workshop on Next Steps for XML Signature and XML Encryption

Ed Simon, XMLsec Inc.
edsimon@xmlsec.com
Background

- XML Canonicalization enables reliable textual and binary comparison of XML documents through the removal of irrelevant differences in structure and content.
- Current approach is to write a single specification that details how all parts of XML instances are to be canonicalized.
- Proposed alternative approach is to layer canonicalization rules according to the XML stack: core, schema-specific, namespace-specific.
- Potential advantages include flexibility and significant optimization of processing.
Canonicalization Layers

• Core – Normalizes the elements, attributes, and whitespace of an XML instance

• Schema-Aware – Normalization of schema-aware aspects including default attributes, schema-defined data types, etc

• Namespace-Aware – Normalization of XML information set nodes that belong to, or are contained by nodes that belong to, an XML node declared with a particular namespace. Includes the normalization of namespace declarations themselves.
Core Canonicalization

- Defined much as per W3C XML Canonicalization version 1.1
- Only canonicalizes what can be derived from the text of the XML instance
- Includes formatting of XML elements and attributes, whitespace, line breaks, CDATA, entities, etc...
- No namespace normalization (but don't worry, it's coming!)
Core Canonicalization Example

```xml
<xsl:stylesheet version='2.0'
    xmlns="http://www.w3.org/1999/xhtml"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:ns1="http://www.xmlsec.com/namespaces/a">
  <xsl:template match="/">
    <p>Total Amount: <xsl:value-of ... select="ns1:expense-report/ns1:total"/></p>
  </xsl:template>
</xsl:stylesheet>

...will be, after core canonicalization, found to be identical to...

```xml
<xsl:stylesheet version="2.0"
    xmlns:ns1="http://www.xmlsec.com/namespaces/a"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns="http://www.w3.org/1999/xhtml">
  <xsl:template match="/">
    <p>Total Amount: <xsl:value-of ... select="ns1:expense-report/ns1:total"/></p>
  </xsl:template>
</xsl:stylesheet>
Schema-Aware Canonicalization

- Is Canonicalization that is aware of the impact of an XML instance's associated schema
- Specific schema language not important. Can be XML Schema, RelaxNG, or any form of XML-applicable schema language
Schema-Aware Canonicalization: Example Schema

```xml
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">
 <xs:element name="p">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="value" type="xs:integer"/>
     </xs:sequence>
     <xs:attribute name="language" type="xs:token" default="en-ca"/>
   </xs:complexType>
 </xs:element>
</xs:schema>
```
Schema-Aware Canonicalization:
Example Schema

Schema-Aware Canonicalization using the aforementioned schema would recognize these two XML fragments as equivalent:

\[
\begin{align*}
\text{<p language="en-ca">}<value>+10</value></p> \\
\text{and} \\
\text{<p><value>10</value></p>}
\end{align*}
\]
Namespace-Aware Canonicalization

- More than the simple normalization of namespaces
- Is the normalization, in view of namespace-specific semantics and processing, of the XML information set
Namespace-Aware Canonicalization: Source to be canonicalized

<Distinguished_Name xmlns="http://example.org/dn">
  <Attribute
    Name="rfcXXXX:cn"
    xmlns:rfcXXX=..."http://example.org/rfcXXXX">
    Sam</Attribute>
</Distinguished_Name>
Namespace-Aware Canonicalization: Canonicalization Result

<Distinguished_Name xmlns="http://example.org/dn">
  <Attribute
    Name="rfcXXXX:commonName"
    xmlns:rfcXXX=...
    "http://example.org/rfcXXXX"
  >Sam</Attribute>
</Distinguished_Name>
Namespace-Aware Canonicalization: XSLT Examples

<xsl:stylesheet version="2.0"
    xmlns:ns1="http://www.xmlsec.com/namespaces/a" ...>
    <xsl:template match="/">
        <p>Total Amount: <xsl:value-of ...
            ...select="ns1:expense-report/ns1:total"/></p>
    </xsl:template>
</xsl:stylesheet>

...would not be equivalent to the following under only core canonicalization because of the different nsX namespaces. But under XSLT-aware canonicalization (an instance of namespace-aware canonicalization), they will be...

<xsl:stylesheet version="2.0"
    xmlns:ns2="http://www.xmlsec.com/namespaces/a" ...>
    <xsl:template match="/">
        <p>Total Amount: <xsl:value-of ...
            ...select="ns2:expense-report/ns2:total"/></p>
    </xsl:template>
</xsl:stylesheet>
Namespace-Aware Canonicalization and Core Canonicalization

- The specification for a particular XML application (e.g. XSLT, SAML, etc.) acts much like a profile of the core XML specifications – it restricts the structure, semantics, and processing

- Those restrictions may be used to provide high degrees of optimization to canonical XML which, as studies show, is a significant resource consumer
Namespace Normalization

• Agree that simply replace original namespace prefixes with enumerated ones (e.g. “ns1”, “ns2”, “ns3”, ...) has its shortcomings

• But so does not replacing the namespace prefixes which are arbitrarily set anyway

• Instead, use a canonical namespace prefix that can is derived from the namespace itself such as by:
  - Using the namespace itself with non-qualifier-allowed characters escaped
  - Using a base64'ed hash (escaped as necessary) of the namespace (e.g. like TinyUrl URI suffix)
Namespace Normalization

Examples

Namespace to be canonicalized: “http://example.org/dn”

- Using escaping:
  ```xml
  <x xmlns:http__COLON__SLASH__SLASHexample__DOTorg__SLASHdn="http://example.org/dn">
    <http__COLON__SLASH__SLASHexample__DOTorg__SLASHdn:commonName>Sam</http__COLON__SLASH__SLASHexample__DOTorg__SLASHdn:commonName>
  </x>
  ```

- Using hashing and base64'ing (and substituting “/” with “_” and “=” with “-”):
  ```xml
  <x xmlns:BeP8rW4G5Umk1Oz6Mi_="http://example.org/dn">
    <BeP8rW4G5Umk1Oz6Mi_:commonName>Sam</BeP8rW4G5Umk1Oz6Mi_:commonName>
  </x>
  ```
Advantage of Namespace-based Namespace Normalization

- The value of the canonical namespace prefix is independent of the element's context AND is not arbitrary
- It is dependent solely on the value of the namespace URI
- "Show me a namespace, I will show you an unambiguous canonical namespace that works everywhere in any situation"
Declaration of Canonicalization Processing

```xml
<Canonicalization
 xmlns="http://w3c.org/xml/canonicalization">
  <Target Namespace="http://example.org/dn">
    <CanonicalizationMethod Algorithm="...
        "http://example.org/dn/canonicalization"/>
  </Target_Namespace>

  <Target
      Namespace="http://www.w3.org/1999/XSL/Transform"
      Version="2.0"
      <CanonicalizationMethod Algorithm="...
          "http://example.org/xslt/2_0/canonicalization"/>
  </Target_Namespace>

</Canonicalization>
```
END