ЦЕНТРАЛЕН РЕГИСТАР

УПАТСТВО ЗА СТАНДАРДИЗАЦИЈА И ПОТПИШУВАЊЕ НА XML ДАТОТЕКА

Верзија 1.0 **Скопје 20.01.2013 год.**

Содржина

1.1 Предуслови

Потребно е да имате квалификуван дигитален сертификат за кој се задоволени следните предуслови:

- Сертификатот е валиден (е издаден од овластените издавачи на територијата на Република Македонија и не е со изминат рок на важност);
- Сертификатот е претходно инсталиран на Вашиот компјутер;

1.2 Дигитално потпишување

XML дигиталните потписи (XMLDSIG) овозможуваат да се осигурате дека податоците не се променувани откако сте ги потпишале. За повеќе информации за XMLDSIG стандардот, проверете ја препораката од W3C XML Signature Syntax and Processing.

XML потписот се состои од Signature елемент во http://www.w3.org/2000/09/xmldsig namespace-от. Основната структура е следна:

```
<Signature>
  <SignedInfo>
    <CanonicalizationMethod />
    <SignatureMethod />
    <Reference>
      <Transforms>
      <DigestMethod>
      <DigestValue>
    </Reference>
    <Reference /> etc.
  </SignedInfo>
  <SignatureValue />
  <KeyInfo />
  <Object />
</Signature>
```

- SignedInfo елементот содржи или ги референцира потпишаните податоци спицифицира кои алгоритми се користени. SignatureMethod и CanonicalizationMethod елементите се користени од SignatureValue елементот и се вклучени во SignedInfo за заштита од промена.

  Еден или повеќе Reference елементи специфицираат ресурс што е потпишан од Uri референца; и секоја преобразба се применува на ресурсите пред потпишување. Трансформација може да биде XPath израз кој избира дефинирана подгрупа од дрвото на документот.

  DigestMethod го специфицира хеш алгоритмот пред да се се имплементира алгоритмот.

  DigestValue го содржи резултатот од применетиот хеш алгоритам на трансформираниот ресурс.

- SignatureValue елементот содржи резултат од енкодиран Base64 потпис – потпис генериран со параметри специфицирани во SignatureMethod елементот од SignedInfo елементот после имплементирањето на алгоритмот специфициран во CanonicalizationMethod.
- KeyInfo елементот опционално може да овозможи на потпишувачот да обезбеди примачи со клуч кој го валидира потписот, обично во форма на еден или повеќе X.509 дигитални сертификати.

Изглед на Signature елементот:

```
<Signature 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#rsa-sha1" />
    <Reference URI="">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature" />
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
      <DigestValue></DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue></SignatureValue>
  <KeyInfo>
    <X509Data>
      <X509Certificate></X509Certificate>
    </X509Data>
  </KeyInfo>
</Signature>
```

1.3 Стандардизација на структура (Canonical XML)

Процес от (canonicalization - c14n) на создавање на канонична форма подразбира конвертирање податоци кои имаат повеќе од едно можно прикажување во стандардизирана односно нормализирана форма.

Канонички XML документ е по дефиниција XML документ кој е во канонична форма дефинирана од Каноничката XML спецификација. Со овој процес се остранууваат празни места во рамките на тагови, користи посебни кодирања на карактери, сортира namespace референци и ги елиминира излишните, ги отстранува XML и DOCTYPE декларациите, и ги трансформира релативните URI-а во апсолутни.

Едноставен пример: 2 верзии на ист XML:

- "<node1>Data</node1> <node2>Data</node2>"
- "<node1>Data</node1> <node2>Data</node2>"

Во каноничната верзија се отстранети празните места:

- "<node1>Data</node1><node2>Data</node2>"

Листа на промени кои настануваат при конвертирање во канонична структура на XML датотека :

- Документот е кодиран во UTF-8
- Нова линија нормализирана со #xA на влез, пред парсирање
- Вредностите на атрибутите се нормализираат
- Карактерите и парсираните референци се отстранууваат
- CDATA секциите се заменуваат со содржината која ја содржат
- XML декларацијата и декларацијата за типот на документ се отстранууваат
- Празните елементи се конвертираат во тагови кои содржат почетен и завршен таг
- Празните места надвор од документ елементот и внатре во почетниот и завршниот таг се нормализираат
- Сите празни места во содржината се задржани (со исклучок на карактерите отстранети поради нова линија)
- Знаците за вредност на атрибут се поставуваат како наводници (двојни)
- Специјалните знаци во вредноста на атрибутот и содржината се заменуваат со карактерни референци
- Излишните namespace декларации се отстранууваат од секој елемент
- Default-те атрибути се додаваат на секој елемент

- Подобрување на XML: базни атрибути
- Лексигокрафски ред се наметнува на namespace декларации и атрибути на секој елемент

Делови од кодови како пример за стандардизација (canonicalization - c14n) на XML датотека се дадени во следниве програмски јазици:

- C#/.NET:
  Може да се користи класата XmlDsigC14NTransform во System.Security.Cryptography.Xml namespace-от за стандардизација (канокалиазција) на XML содржина.

```
public static void CanonicalizeXml(XmlDocument xmlDoc)
{
    var transform = new XmlDsigC14NTransform();
    transform.LoadInput(xmlDoc);
    var ms = (MemoryStream)transform.GetOutput();
    var sr = new StreamReader(ms);
    xmlDoc.LoadXml(sr.ReadToEnd());

}
```

- Java: Може да се користи Apache XML Security for JAVA библиотеката, достапна на http://santuario.apache.org/download.html. Потребно е да се додадат референци до библиотеките *xmlsec-1.5.6.jar* и *commons-logging-1.1.1.jar* и да се импортира *org.apache.xml.security.c14n.Canonicalizer* namespace-от.

```
private static String canonicalizeXml(String XmlPath) {
      File fXmlFile = new File(XmlPath);
      DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
      DocumentBuilder dBuilder;
      byte outputBytes[] = null;
      try {
            dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(fXmlFile);
            Canonicalizer c14n = Canonicalizer.getInstance(
                  "http://www.w3.org/TR/2001/REC-xml-c14n-
                  20010315#WithComments");
            outputBytes = c14n.canonicalizeSubtree(doc);
      } catch (Exception ex) {
            Logger.getLogger(Canonicalization.class.getName())
                  .log(Level.SEVERE, null, ex);
      }

      return new String(outputBytes);
}
```

1.4 Примери на програмска имплементација за дигитално потпишување

Пример за потпишување на XML датотека:

- **C#/.NET**:
  Може да се користат класите во System.Security.Cryptography.Xml namespace-от за да потпишете XML документ или дел од XML документ со дигитален потпис.

```
public static void SignXml(XmlDocument xmlDoc, RSA Key, X509Certificate Certifi
cate)
{
    if (xmlDoc == null)
        throw new ArgumentException("xmlDoc");
    if (Key == null)
        throw new ArgumentException("Key");

    var transform = new XmlDsigC14NTransform();
    transform.LoadInput(xmlDoc);
```

```
            var ms = (MemoryStream)transform.GetOutput();
            var sr = new StreamReader(ms);
            xmlDoc.LoadXml(sr.ReadToEnd());

            SignedXml signedXml = new SignedXml(xmlDoc);

            signedXml.SigningKey = Key;

            Reference reference = new Reference();
            reference.Uri = "";

            XmlDsigEnvelopedSignatureTransform env = new XmlDsigEnvelopedSignatureTrans
form();
            reference.AddTransform(env);
            signedXml.AddReference(reference);

            KeyInfo keyInfo = new KeyInfo();
            keyInfo.AddClause(new KeyInfoX509Data(Certificate));
            signedXml.KeyInfo = keyInfo;

            signedXml.ComputeSignature();

            XmlElement xmlDigitalSignature = signedXml.GetXml();

            xmlDoc.DocumentElement.AppendChild(xmlDoc.ImportNode(xmlDigitalSignature, t
rue));
        }
```

- **Java**:

```
        private JSignedXml doSign(String XmlToSign, X509Certificate m_cert, boolean inc
ludeKeyInfo) {
            SignatureOptions sigOpt = new SignatureOptions();
            sigOpt.setDataToSign(XmlToSign);
            sigOpt.setCertificate(m_cert);
            sigOpt.setPrivateKey((PrivateKey) ks.getKey(ks.getAliasFromCertificate(m_ce
rt)));
            return signedXml;
        }
```

Напомена: Централен регистар не презема никаква одговорност за целосната успешност на конкретна имплементација, користејќи ги овие примери дадени во ова упатство.