Applying XML signatures to the definition of an XML schema for digital ballots

Augusto Jun Devegili
devegili@ulbra-to.br

Heres Edison Valdivieso Tobar Neto
heresneto@ulbra-to.br

Centro Universitário Luterano de Palmas
Universidade Luterana do Brasil
1501 Sul Av. Teotônio Segurado, s/n
77054-970, Palmas, TO, Brazil

Abstract: *This article presents the definition of an XML schema of digital ballots for official elections using XML signature, thus defining the syntax of the digital ballot and providing the services of identification, integrity and non-repudiation of the ballot. The proposed schema was validated in regard to the XML Schema standard and an application was developed in order to allow the configuration of digital ballots for official elections.*

Keywords: *digital ballot, electronic voting, XML signature, XML schema*

1 Introduction

An electronic voting system is a distributed application comprised of a set of mechanisms and cryptographic protocols [1]. Being a distributed application, it is necessary for data to flow among computers. Such data may be defined and coded in various formats such as ASN.1 [2]. Due to the growth of Internet, XML [3] has been increasingly used to interchange structured data. Several XML extensions have been proposed such as digital signatures [4] and encryption [5].

One possible way to define the syntax of XML documents is via an XML schema. Therefore, digital ballots may be represented as XML documents and their syntax defined by a specific XML schema. Digitally signed, an XML ballot provides two important services: source identification (authentication) [6] and modification detection (integrity) [7]. The work described in this paper proposes an XML schema which incorporates XML signatures and may be used in electronic voting ballots for official elections. The Election and Voter Services Technical Committee[1] from OASIS – Organization for the Advancement of Structured Information Standards is also known to be doing similar work, however little information is publicly disclosed and, as far as it was possible to know, their schema is not currently considering XML signatures.

[1] http://www.oasis-open.org/committees/election/

This document is organized as follows: chapter 2 gives a brief introduction on electronic voting schemes; chapter 3 discusses the XML standards which have been used; chapter 4 presents our XML schema proposal. Finally, chapter 5 concludes this paper.

2 Electronic voting schemes

According to [1], "an electronic voting scheme is a distributed application constituted by a set of cryptographic mechanisms and protocols that jointly allow an election to take place entirely over a computer network, in a secure way, even assuming that the legitimate participants can have a malicious behaviour". An electronic voting process should contain the following phases [8]: (i) configuration, where general information is defined and voting is initialized; (ii) enrollment, where authorized voters are registered; (iii) casting, where voters cast their votes; and (iv) tallying, where votes are tallied and results are published. Electronic voting is not applied only to elections; it may be used in group decision making or general polls [8].

In the configuration phase several features are defined, such as the voting unique identifier, enrollment period and voting period, as well as the structure of the ballot (items and options). It is often necessary to define a voting authority, responsible for the configuration phase, who must publish the standard ballot that will be used in the voting phase [8].

3 XML standards

XML schemas are an alternative to DTDs in the definition of the syntax of an XML document. It aims at solving the following DTD's problems [9]: (i) DTDs do not allow the definition of data types (e.g. numbers, strings, dates) and the domain of possible values (e.g. positive integers, number of digits); (ii) DTDs have their own syntax originated from SGML which does not follow the general XML syntax; (iii) it is not trivial to combine different DTDs to generate a resulting DTD, thus reducing the extensibility of these definitions; (iv) the compatibility between DTDs and namespaces is limited and it is not possible to use URIs, the basic mechanism for defining namespaces; and (v) it is complex (or even impossible) to define restrictions based on relationships among elements. DTDs are more appropriate to describe narrative content; XML schemas are more appropriate to describe complex structured data with defined types. An XML schema must contain: (i) the processing instruction that indicates that the schema itself is an XML document; (ii) a reference to the namespace defined by W3C; (iii) the definition of the root element as well as attributes and other elements. A *simple* element does not contain attributes or child elements, as opposed to *complex* elements.

In [10] an XML namespace is defined as "a collection of names, identified by a URI reference [RFC2396], which are used in XML documents as element types and attribute names". As a result, it is possible to have two different elements with the same name as long as they are in different namespaces. Prefixes are used so that namespaces may be referenced in an XML document. Prefixless elements are in the standard namespace (identified by the `xmlns` attribute) and it is necessary to define the target namespace for elements defined in an schema.

XML Signature describes how to sign data, especially, but not limited to, XML

documents [11]. The signature itself is an XML element. When the signature is located apart from the document that was signed, it is named *detached*. If the signature is inside the document, it may be of two types: *enveloping*, where signed data are a child element of the signature, or *enveloped*, where the signature is child of the element that is being signed [4]. The main elements of an XML signature are: `Signature`, the root element that contains signature information; `SignedInfo`, containing the hash of the signed data; `SignatureValue`, containing the signature itself; and `KeyInfo`, in which data related to the key used to sign the document is kept and, if appropriate, the digital certificate related to that key.

4 An XML schema for digital ballots in official elections

We propose the schema listed in appendix A for digital ballots in official elections. This schema has been validated in regard to the XML schema specifications [12, 13, 14] using the tools Oracle XDKJava [15] and Apache Xerces-J [16]. XML ballots were validated in regard to our schema using Java classes from Xerces-J. We use classes from XSS4J [17] to produce an *enveloped* signature of the ballot.

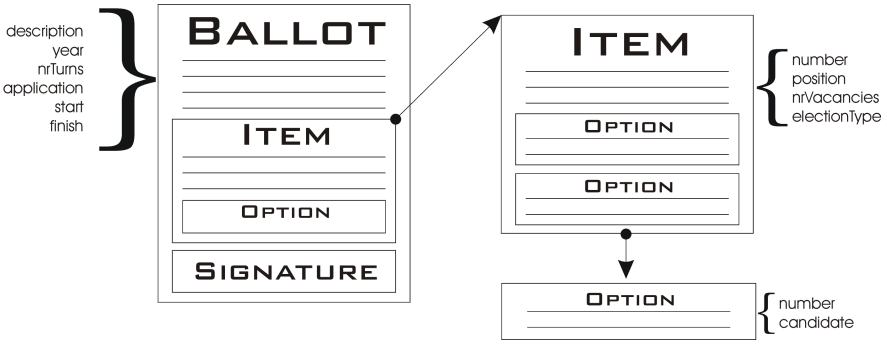

Figure 1: Ballot scheme

Figure 1 presents a graphical view of the XML schema that was defined for digital ballots. The root element, named `ballot`, is defined in the namespace `http://www.labsec.ufsc.br/ostracon`. As this schema uses XML signatures, the digital signature namespace `http://www.w3.org/2000/09/xmldsig#` is imported, identified by the prefix `dsig`.

The root element `ballot` is defined as being of the type `TBallot`, which contains the elements `description`, `year`, `nrTurns` and `applicationType`, all of them being simple types (element `applicationType` may contain the values `Election`, `Simulation` and `Training`). Both the voting starting date/time and end date/time are defined as being of complex type `TDateTime`. The ballot signature is contained in the element `Signature`, whose definition is referenced from the XML signature namespace. Only one attributed is defined, namely `id`, which identifies the element that must be signed: the whole ballot, identified by `ballot`.

A ballot is comprised of a set of items. Elements of type `item` may occur infinite times within element `ballot`. An item identifies a chair, how many vacancies are available for that

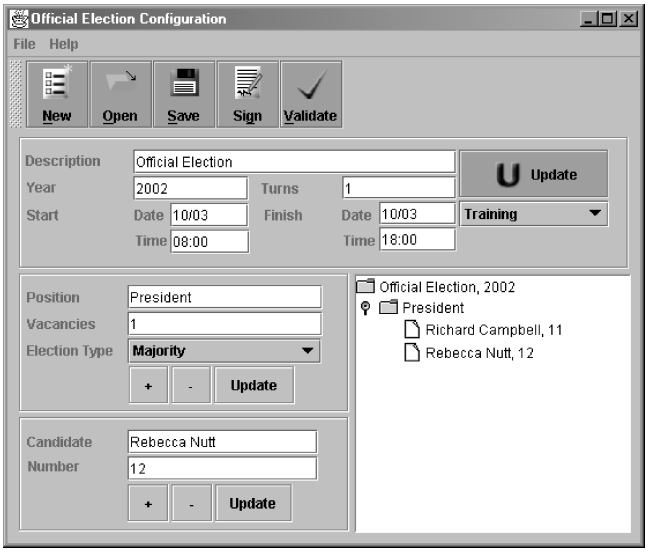

Figure 2: User interface for the ballot configuration user agent

chair and the type of the election (majority or proportional). Furthermore, all options for that item are listed. Every option has two data: the candidate number and his name.

Once the XML schema is defined, the person in charge of configuring the election may use the interface shown in figure 2 to define the election parameters. Button `Open` allows the opening of an existing XML ballot; button `Save` saves the configuration in an XML document; button `Sign` signs the XML ballot and button `Validate` validates the XML ballot in regard to the previously defined schema. In order to sign the ballot, both the private key and its related X.509v3 digital certificate are used; they are stored by `keytool`, part of the Sun Java Runtime Engine.

5 Conclusions

This paper presented the definition of an XML schema for digital ballots in official elections. The definition of an XML schema provides for the validation of XML documents containing digital ballots, assuring that their syntax is valid in regard to their definition. Furthermore, in order to allow the verification of ballot integrity and trusted origin, it is digitally signed, being the signature an XML element inside the ballot.

The use of XML schemas instead of DTDs provides for greater control of ballot syntax, such as possible values for element `applicationType`. XML namespaces allows limiting names inside a proper context, isolating them from other contexts and making it possible to use same names in different namespaces.

Ballots using our schema may be used in electronic voting protocols in two situations: when a user voting agent needs a blank digital ballot to show the options to the user, it is necessary to verify whether the ballot is correct for the current election. As the blank ballot is signed by the voting authority, the user agent is sure it is dealing with the appropriate ballot. Moreover, in electronic voting protocols based on scrutinizers such as those in [8, 18], ballots may be

signed so that it is possible to check that they were verified by the scrutinizers. And, in regard to report [19] by the Caltech/MIT Voting Technology Project, ballots may be part of FROGs – devices in charge of physically registering votes.

As for future work we envision the definition of XML schemas for different voting styles besides official elections and probably the definition of an XML metaschema that generalizes different ballot schemas. Other information regarding the voting process may be also defined by XML schemas and, if appropriate, signed with XML signatures.

References

[1] RIERA, Andreu. *Design of Implementable Solutions for Large Scale Electronic Voting Schemes*. 1999. Tese (Doutorado em Ciência da Computação) – Departament d'Informàtica, Universitat Autònoma de Barcelona, Barcelona.

[2] INTERNATIONAL TELECOMMUNICATION UNION. *Recommendation X.680 (12/97) – Information Technology – Abstract Syntax Notation One (ASN.1): Specification of Basic Notation*. Telecom Standardization, 1997.

[3] WORLD WIDE WEB CONSORTIUM. *Extensible Markup Language (XML) 1.0 (Second Edition)*. [s.l.]: W3C, out. 2000. Disponível em: <http://www.w3.org/TR/2000/REC-xml-20001006>. Acesso em: 2 out. 2001.

[4] WORLD WIDE WEB CONSORTIUM. *XML-Signature Syntax and Processing (W3C Candidate Recommendation)*. [s.l.]: W3C, 20 ago. 2001. Disponível em: <http://www.w3.org/TR/2001/PR-xmldsig-core-20010820/>. Acesso em: 2 out. 2001.

[5] WORLD WIDE WEB CONSORTIUM. *XML Encryption Syntax and Processing (WG Working Draft)*. [s.l.]: W3C, 26 jun. 2001. Disponível em: <http://www.w3.org/TR/2001/WD-xmlenc-core-20010626/>. Acesso em: 2 out. 2001.

[6] DIFFIE, Whitfield; HELLMAN, Martin E. New directions in cryptography. *IEEE Transactions on Information Theory*, v. IT-22, n .6, p.644–654, 1976.

[7] DAVIES, D. W.; PRICE, W. L. The application of digital signatures based on public-key cryptosystems. In: *Proc. Fifth Intl. Computer Communications Conference*, p. 525–530, 1980.

[8] DEVEGILI, Augusto Jun. *Farnel: Uma proposta de protocolo criptográfico para votação digital*. 2001. Dissertação (Mestrado em Ciência da Computação) – Departamento de Informática e Estatística, Universidade Federal de Santa Catarina, Florianópolis.

[9] HAROLD, Elliotte Rusty. *XML Bible*. New York: Hungry Minds, 2001.

[10] WORLD WIDE WEB CONSORTIUM. *Namespaces in XML (W3C Recommendation)*. [s.l.]: W3C, 14 jan. 1999. Disponível em: <http://www.w3.org/TR/1999/REC-xml-names-19990114/>. Acesso em: 2 out. 2001.

[11] INTERNET ENGINEERING TASK FORCE. *RFC2807 – XML Signature Requirements*. [s.l.]: IETF, jul. 2000. Disponível em: <http://www.ietf.org/rfc/rfc2807.txt>. Acesso em: 2 out. 2001.

[12] WORLD WIDE WEB CONSORTIUM. *XML Schema Part 0: Primer (W3C Recommendation)*. [s.l.]: W3C, 2 maio 2001. Disponível em: <http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/>. Acesso em: 2 out. 2001.

[13] WORLD WIDE WEB CONSORTIUM. *XML Schema Part 1: Structures (W3C Recommendation)*. [s.l.]: W3C, 2 maio 2001. Disponível em: <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>. Acesso em: 2 out. 2001.

[14] WORLD WIDE WEB CONSORTIUM. *XML Schema Part 2: Datatypes (W3C Recommendation)*. [s.l.]: W3C, 2 maio 2001. Disponível em: <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>. Acesso em: 2 out. 2001.

[15] ORACLE. *Oracle XML Developer's Kit for Java*. Disponível em: <http://web06-02.us.oracle.com/tech/xml/xdk_java/content.html>. Acesso em: 2 out. 2001.

[16] APACHE. *The Apache XML Project*. Xerces Java Parser. Disponível em: <http://xml.apache.org>. Acesso em: 2 out. 2001.

[17] IBM. *XML Security Suite*. XML-Signature Implementation. Disponível em: <http://www.trl.ibm.com/projects/xml/xss4j>. Acesso em: 2 out. 2001.

[18] KU, Wei-Chi; WANG, Sheng-De. A secure and practical electronic voting scheme. In: *Computer Communications*, Amsterdam: Elsevier Science, v. 22, n. 3, p.279–286, fev. 1999.

[19] CALTECH–MIT/VOTING TECHNOLOGY PROJECT. *Voting – What is, what could be.* [s.l., s.n.], jul. 2001. Disponível em: <http://vote.caltech.edu/Reports/july01/July01_VTP_%20Voting_Report_Entire.pdf>. Acesso em: 2 out. 2001.

Appendix

A XML schema for digital ballots

```
<?xml version="1.0"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
xmlns="http://www.labsec.ufsc.br/ostracon"
targetNamespace="http://www.labsec.ufsc.br/ostracon"
elementFormDefault="unqualified"
attributeFormDefault="unqualified">

<xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
  schemaLocation="xmldsig-core-schema.xsd"/>

<xsd:element name="ballot" type="TBallot"/>

<xsd:complexType name="TBallot">
  <xsd:sequence>
    <xsd:element name="description" type="xsd:string"/>
    <xsd:element name="year" type="xsd:gYear"/>
    <xsd:element name="nrTurns" type="xsd:positiveInteger"/>
    <xsd:element name="application" type="TApplication"/>
```

```
    <xsd:element name="start" type="TDateTime"/>
    <xsd:element name="finish" type="TDateTime"/>
    <xsd:element name="item" type="TItem" minOccurs="1" maxOccurs="unbounded"/>
    <xsd:element ref="dsig:Signature"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>

<xsd:simpleType name="TApplication">
  <xsd:restriction base="xsd:token">
    <xsd:enumeration value="Election"/>
    <xsd:enumeration value="Simulation"/>
    <xsd:enumeration value="Training"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="TItem">
  <xsd:sequence>
    <xsd:element name="number" type="xsd:positiveInteger"/>
    <xsd:element name="position" type="xsd:string"/>
    <xsd:element name="nrVacancies" type="xsd:positiveInteger"/>
    <xsd:element name="electionType" type="TElectionType"/>
    <xsd:element name="option" type="TOption" minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="TElectionType">
  <xsd:restriction base="xsd:token">
    <xsd:enumeration value="Majority"/>
    <xsd:enumeration value="Proportional"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="TOption">
  <xsd:sequence>
    <xsd:element name="number" type="xsd:positiveInteger"/>
    <xsd:element name="candidate" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="TDateTime">
  <xsd:sequence>
    <xsd:element name="date" type="xsd:gMonthDay"/>
    <xsd:element name="time" type="xsd:time"/>
  </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```