PATENT SPECIFICATION

(11) S85986

(21) Application No. S2010/0277

(22) Date of Filing of Application: 05/05/2010

(45) Specification Published: 09 May 2012

(51) Int. Cl. (2011)
H04L 9/00
H04L 9/32

(54) Title: Securing a computer network using OWL2 digital certificates

(72) Inventor: EAMON O'TUATHAIL

(73) Patent Granted to: EAMON O'TUATHAIL, Irish, 24 Thomastown Road Dun Laoghaire Co. Dublin, Ireland

S85986

Securing a computer network using OWL2 digital certificates

Field of invention

The present invention relates to securing a computer network using Public Key Infrastructure (PKI) based on semantic web concepts.

With the universal acceptance of web standards (such as XML and URI) and the growing interest in semantic web standards (such as OWL2 and RDF) it is advantageous to build PKI based on this foundation (with signing via XML digital signatures), rather than on older constructs that are little known and have little use outside of security/networking circles – such as ASN.1, Distinguished Names (DNs) and Object Identifiers (OIDs).

Background to the invention

The humble username and password still remain the dominant authentication mechanism worldwide, despite the fact that most security experts believe Public Key Infrastructure (PKI) offers a far superior level of security (and broader range of security services – e.g. data integrity and confidentiality). PKI has not been a failure, but it certainly has not been a huge success either. PKI has been successfully deployed in some large corporate networks (where there are skilled system administrators available) and certain other areas (e.g. server-side authentication/confidentiality for eCommerce sites) but in general it has had limited impact.

At the heart of PKI is a very simple concept – securing a computer network using an electronic document known as a digital certificate that ties together the subject's identifier and public key, and a Certification Authority (CA) makes this information and additional security assertions securely available so that others (who trust the CA) may rely upon it. What should be a very simple infrastructure has evolved into something that takes far too much effort, money and time to get right.

One of the issues is that what one needs to learn about existing PKI (X.509, ASN.1, DNs, OIDs) have little or no use outside of PKI for many administrators, and are simply unknown to general computer users.

The present invention aims to reimagine how PKI works using the modern semantic web and to secure computer networks based on the result.

Moving from ASN.1 to OWL2

The biggest difference between X.509 and the present invention is that the former uses ASN.1 and the latter the W3C standard, Web Ontology Language, version 2 (OWL2), which is part of the W3C's Semantic Web initiative (http://www.w3.org/standards/semanticweb/).

The number of people using ASN.1 on new projects is decreasing, whereas the number using OWL2 is increasing, and is likely to accelerate as the Semantic Web evolves. ASN.1 users are concentrated in the ITU/ISO standards arena, whereas OWL2 is seeing a much more diverse userbase (CC/PP for describing device capabilities and user preferences, Dublin Core for metadata, many ontologies and plenty of custom applications).

OWL2 is a key building block for the semantic web and hence is increasingly likely to be part of the general skillset of developers, administrators and experienced users in future. OWL2 is an assertional language - a certificate is a set of one or more assertions (e.g. this public key belongs to this subject) and hence it is an ideal basis for specifying certificates.

The real goal of the semantic web is to enable software agents make inferences based on statements. Though this is a work in progress, as advances are made it becomes more useful to have verified identity information, such as certificates, readily available (rather than having to translate them from other formats).

One advantage ASN.1 has over XML-based OWL2 is that it is more compact. In the age of huge hard disks and fast networks, such an advantage is not overwhelming.

Moving from DNs to URIs

Identity (of people, computers, resources) is the foundation of securing computer networks. In the XML world, URIs provide uniqueness. In practice, URIs have been shown to work extremely well. URIs have massive public acceptance, appearing on everything from buses to newspapers to breakfast cereal boxes. The following URI is far more acceptable to end-users than a similar DN:

- mailto:john.doe@example.com

In stark contrast, the Distinguished Names (DNs) used by X.509 have zero public acceptance, and even within the development community, relatively few have a good understanding of them.

The proponents of DNs will argue they offer more than just uniqueness for the subject's identifier – they also offer a hierarchical naming scheme. True, but this is much too rigid for the real world. People move locality and country and change organizational units often. DNs imposes a static formal hierarchical structure when in most cases it is far more fluid. The Semantic Web is all about classification, so additional statements may be added to the certificate to indicate organizational association if needed.

Moving from OIDs to URIs

ASN.1 uses Object Identifiers (OIDs) to uniquely identify objects in a well-known hierarchy. Standard-setting organizations and individual companies may apply for an object node in the hierarchy and any objects they created can be placed below their node and marked as "belonging" to them.

The present invention uses URIs to identify constructs that need to have identity. There is no need to apply for a URI – so, unlike OIDs, there is no global registry, but such a registry brings little tangible benefit.

Moving from ASN.1 Extensions to Certified Statements

The significant new feature in version 3 X.509 certificates was the introduction of extensions. This allowed certification authorities to append to the certificate custom information in addition to the subject's public key. Examples could include a photograph or biometric information. Custom extensions have been created by companies that write certification authorities and specialist security companies, but very few general corporate development teams have create such extensions, despite the fact that they could be highly useful.

X.509 certificate extensions must be written in ASN.1, and this is a major problem for the general developer community which does not know ASN.1.

With the present invention, certificates contain a range of OWL2/RDF statements,

including custom statements added by local certification authorities. Since it is a syntax that will be better known, hopefully we will see a significant increase in the use of certificate extensions.

With the semantic web, many statements may be made about subjects – and such statements may come from many different places. Whether we believe them or not is a totally different issue - much like the web today – some places we trust, and some not so. The benefit of having such statements inside the certificate is that the certification authority is vouching for their authenticity.

Removal of Little Used Fields

Two fields that X.509 certificates may contain but the OWL2 digital certificates in the present invention does not, are:

- Issuer Unique Identifier – allows an issuer's name to be reused
- Subject Unique Identifier – allows a subject's name to be reused

These fields are optional in X.509 and section 4.1.2.8 of IETF RFC2459 recommends that names not be reused and that CAs should not generate certificates with unique identifiers.

Removal of Field Containing Duplicated Information

The Signature field in the X.509 certificate stores the algorithm identifier for the algorithm that the CA uses to sign the certificate. (An aside: this field really should be called algorithm identifier, because that is what it contains and not a signature). This information is repeated in the X.509 certificate when the actual certificate is stored (the encrypted field).

With the present invention, the XML Digital Signature contains information about the algorithm used to sign the certificate and this information is not supplicated inside the certificate.

Adding Extension Fields to Main Body of Certificate

With the present invention, certain fields that would normally appear as extensions in

X.509 certificates (e.g. Key usage, CRL distribution point) have moved to the main body of the certificate. It is recommended that most certificates have these fields filled in, and it is more clearly indicated by not marking them as "extensions".

Statement of invention

Accordingly, there is provided a method for securing a computer network comprising the steps of:

- defining an OWL2 digital certificate to store the principal's identity and public key,
- modifying a security protocol to access the principal's identity and public key from said certificate,
- connecting computer devices via a computer network, and
- securing said network using said security protocol.

In one embodiment, the security protocol is Transport Layer Security (TLS).

In one embodiment, the security protocol is XML Digital Signatures.

In one embodiment, the security protocol is XML Encryption.

In one embodiment, the security protocol is the Security Assertion Markup Language (SAML).

The principle advantage of the presented invention is that providing PKI based on Semantic Web standards is likely to see greater use of PKI and its easier integration with other information processes.

Brief description of the drawings

Fig. 1 shows the major components.

Detailed description

The present invention relates to securing computer networks using OWL2-based digital certificates.

As shown in figure 1, it relates to a computer network (1) comprising of a client computer (2), a server computer (3) and a network connection (4) between them; and an OWL2 digital certificate (5) and a security protocol (6) that accesses the certificate to determine the public key, identifier and other information needed for securing the network communication.

It is assumed that the reader has ready access to computers, network equipment and standard security protocols (TLS/SSL, SAML, XML Digital Signatures/Encryption etc.), and hence our discussion below focuses on what is unique in the presented invention.

Our discussion of the invention is divided into two parts:

- Part 1 discusses the layout of the OWL2 certificate itself
- Part 2 explains how such a digital certificate can be used with computer networks and security protocols

PART 1 – OWL2 Digital Certificate Format

The primary role of a certificate is to specify an assertion by a certification authority that a particular public key is associated with a particular subject. Additional assertions (extensions) may also be made by the Certification Authority (CA) about the subject and they too must be securely stated.

Certificate Statements

An OWL2 certificate must contain the following statements defined using the Web Ontology Language, version 2:

- VersionUri [URI]: Indicates which certificate version format is used
- CertificateUri [URI]: A unique identifier for this particular certificate
- SubjectUri [URI]: A unique identifier for the subject
- hasSubjectPublicKeyInfo [depends on algorithm]: the public key information, algorithm identifier and parameters

- IssuerUri [URI]: which certification authority issued this certificate
- Validity Period – NotValidAfter and NotValidBefore (date): The period in which this certificate is to be consider valid (provided it has not been revoked)

An OWL2 digital certificate may contain the following statements:

- hasCRLDistributionPoint (CRLDistributionPoint): Where to find the certificate revocation list for this CA
- KeyUsage (list of URIs): How should this certificate be used
- CertificationPracticeStatementUri: Where to locate policy information

A certificate may also contain additional statements.

VersionUri

The version of the OWL2 specification used for this certificate. Certification authorities should issue certificates with the most recent version they support. Client software should work with the most recent version they support and previous versions.

CertificateUri

This is a unique identifier that distinguishes this certificate from others issued by this Issuer. It plays the role of a "serial number" for a certificate. CertificateUri and IssuerUri together provide global uniqueness.

SubjectUri

A unique identifier for the subject whose public key info is stored in this certificate. If this certificate is for a root CA itself, then the SubjectUri and IssuerUri must be the same (e.g. a "self-signed" cert). A CA must ensure that for each subject a different SubjectUri is used. A CA may issue multiple certificates to the same subject (e.g. one for signing and one for encryption).

hasSubjectPublicKeyInfo

The public key information that needs to be associated with the subject.

This is an ObjectProperty of range PublicKeyInfo - whose sub-classes contain information for different algorithms. Two sub-classes are currently defined, one named RSAPublicKeyInfo for RSA (with DatatypeProperties Modulus and Public Exponent) and one named DSAPublicKeyInfo for DSA (with DatatypeProperties for P, Q and G).

IssuerUri

A unique identifier for the CA issuing this certificate.

NotValidBefore and NotValidAfter

The dates before and after which the certificate should not be considered valid.

hasCRLDistributionPoint

An optional ObjectProperty of type CRLDistributionPoint.

- CRLDistributionPoint has the following properties:
- CRLIssuer – where the CRL can be found (if not the same CA as who issued the certificate)
- DistributionPointUrl – The Url from where the CRL may be downloaded
- hasRevocationReason – why the revocation is needed

hasRevocationReason is an enumeration with the following values (based on RFC 2459, section: 4.2.1):

- AffiliationChanged
- CaCompromise
- CessationOfOperation

- KeyCompromise
- Superseded
- Unused

hasKeyUsage

hasKeyUsage indicates permissible usage for this certificate – it is an enumeration with the following values:

- CRLSign
- dataEncipherment
- decipherOnly
- digitalSignature
- encipherOnly
- keyAgreement
- keyCertSign
- keyEncipherment
- nonRepudiation
- codeSigning
- timeStamping
- tlsClientAuthentication
- tlsServerAuthentication

CertificationPracticeStatementUri

Information containing the policy in force during the operation of the CA and the granting of certificates. It is a URL to a document that may be displayed to end-users and administrators.

Sample Certificate

```
<Certificate rdf:ID="JohnDoe">

  <VersionUri
  rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
http://www.clipcode.com/Identity/OWL2/March10</VersionUri>

  <NotValidBefore
    rdf:datatype="http://www.w3.org/2001/XMLSchema#date">
    2010-03-02</NotValidBefore>

  <SubjectUri
    rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
    mailto:john.doe@example.com</SubjectUri>

  <IssuerUri
    rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
    http://www.cert_auth.example.com</IssuerUri>

 <CertificateUri
    rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
    http://www.cert_auth.example.com/1234</CertificateUri>

  <NotValidAfter
    rdf:datatype="http://www.w3.org/2001/XMLSchema#date">
    2012-03-01</NotValidAfter>

</Certificate>
```

Securing the Certificate with XML Digital Signatures

XML digital signatures (as defined in the W3C XML Signature Syntax and Processing standard - http://www.w3.org/TR/xmldsig-core/) are used to provide integrity for the certificate. OWL2 certificates must be signed by the Certification Authority (CA) using XML digital signatures.

PART2-Using OWL2 digital certificates with computer networks & security protocols

This section discusses use of OWL2 certificates with a range of well-known security protocols.

Most protocols that make use of certificates are not tied to a particular format (e.g. X.509), but rather allow the format to be configured as a normal part of the protocol.

Some however, need to be slighted extended to support the new certificate format type.

XML Signature and XML Encryption

XML Encryption (http://www.w3.org/TR/xmlenc-core/) provides for the encryption of-, and XML Digital Signatures provides for the signing of XML data. In both cases, the cert data may be provided by a KeyInfo element. The KeyInfo has the following definition:

```
<element name="KeyInfo" type="ds:KeyInfoType"/>
   <complexType name="KeyInfoType" mixed="true">
     <choice maxOccurs="unbounded">
       <element ref="ds:KeyName"/>
       <element ref="ds:KeyValue"/>
       <element ref="ds:RetrievalMethod"/>
       <element ref="ds:X509Data"/>
       <element ref="ds:PGPData"/>
       <element ref="ds:SPKIData"/>
       <element ref="ds:MgmtData"/>
       <any processContents="lax" namespace="##other"/>
       <!-- (1,1) elements from (0,unbounded) namespaces -->
     </choice>
     <attribute name="Id" type="ID" use="optional"/>
   </complexType>
```

It contains an unbounded choice of a variety of keying data. It is noted that included in the choice is the wildcard any, which allows third parties to extend the available KeyInfo structures. The presented invention provides a schema which defines an element, Owl2CertData, that may be placed within KeyInfo. Its identifier is:

- http://www.clipcode.com/identity/20100304/Owl2Data

and it has the following schema:

```
<?xml version="1.0" encoding="utf-8"?>

<xs:schema targetNamespace=
    "http://clipcode.com/Identity/owl2-xmldsig-keyinfo.xsd"
     elementFormDefault="qualified"
     xmlns=
     "http://clipcode.com/Identity/owl2-xmldsig-keyinfo.xsd"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

     <xs:element name="Owl2CertData"
           type="Owl2CertDataType" />

     <xs:complexType name="Owl2CertDataType">

          <xs:sequence>

               <xs:element name="Owl2Certificate"
                     type="rdf:RDF" />

          </xs:sequence>

          <xs:attribute name="IssuerUri" type="xs:anyURI" />

          <xs:attribute name="SubjectUri" type="xs:anyURI" />

    </xs:complexType>

</xs:schema>
```

It is noted that the signing of OWL2 certificates themselves make use of XML Signatures.

Transport Layer Security (TLS) / Secure Sockets Layer (SSL)

TLS is the updated version of SSL.

Unlike XML Digital Signatures, the IETF TLS specification (RFC 2246 - http://www.ietf.org/rfc/rfc2246.txt]) Version 1.0 is not flexible in the certificate formats it uses – the current TLS specification states that Distinguished Names (DNs) must be used and that the certificate formats must be X.509 (e.g. Section 7.4.2 of RFC 2246 states: "All certificate profiles, key and cryptographic formats are defined by the IETF PKIX working group").

To use OWL2 certificates with TLS, TLS needs to be extended in a small number of places. The version number needs to be incremented, and two messages, Certificate Request and Certificate need to be extended.

TLS Version Number

The TLS Record Layer contains a two-byte version field, one byte for major version and one byte for minor version. To indicate usage of the updated version of TLS, the minor version number field is to be set to 2 (it is 1 for TLS 1.0 and 0 for SSL). The major version number remains at 3, as it is for TLS 1.0 and SSL.

Implementations of this updated version of TLS must support OWL2 certificates and may support X.509 certificates.

TLS Certificate Request Message

This message is sent from a non-anonymous server to a client and contains a list of identifiers for Certification Authorities which the server finds acceptable.

With X.509, CA identifiers are Distinguished Names and with the present invention, they are URIs. The Certificate Request message has been extended to include an identifier format and support for carrying URIs or DNs.

Note: the syntax used in RFC2246 does not directly permit the specification of extensibility – hence we use "..." to indicate the possibility of additional formats beyond the present invention and X.509 certificates.

Using the syntax of RFC2246:

```
enum {uri (1), dn (2), ...} CAIdentifierFormatType;
```

```
opaque Uri<1..2^16-1>;
opaque DistinguishedName<1..2^16-1>;
struct {
          ClientCertificateType certificate_types<1..2^8-1>;
          CAIdentifierFormatType identifier_format;
          union {
     DistinguishedName dn_certificate_authorities<3..2^16-1>;
     Uri uri_certificate_authorities<3..2^16-1>;
                         ...
          }
     }
} CertificateRequest;
```

TLS Certificate Message

With TLS, the server sends the Certificate message after the Server Hello message and the client sends it after receiving Server Hello Done, if it has received a Certificate Request message from the server during the initial handshake. This message has been extended to include an identifier for the certificate format and support for carrying OWL2 certificates.

```
enum {Owl2Pki (1), x509 (2), ...} CertificateFormatType;
opaque Owl2Cert<1..2^24-1>;
opaque ASN.1Cert<1..2^24-1>;
       struct {
                    CertificateFormatType certificate_format;
                    union {
     Owl2Cert Owl2_certificate_list<0..2624-1>;
     ASN.1Cert x509_certificate_list<0..2^24-1>;
          }
```

```
} Certificate;
```

Security Assertion Markup Language (SAML)

SAML is defined by the OASIS Security Services Committee (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security).

It is noted SAML uses KeyInfo from XML Signature, and what is specified in section 5.1 of this specification regarding KeyInfo also applies to its usage in SAML.

SAML Name Identifier Format

The Format attribute for Name Identifiers is used in <NameId>, <NameIDPolicy> and <Issuer> elements (section 8.3 of [SAML]) and it defines which format the name is in. For names used with OWL2 certificates, use the following URI:

http://clipcode.com/Identity/20100302/Owl2PKINameFormat

SAML AuthenticationContext

[SAML-AUTH-CTX] states that for SAML: "Authentication context is defined as the information, additional to the authentication assertion itself, that the service provider may require before it makes an entitlements decision with respect to an authentication assertion."

The SAML specification defines authentication contexts for X5.09 and SPKI certificates, and based on these is the following authentication context for OWL2 certificates:

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema targetNamespace=
  "http://clipcode.com/Identity/20100302/Owl2PKI-AC"
   xmlns:ac="urn:oasis:names:tc:SAML:2.0:ac"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="http://clipcode.com/Identity/20100302/Owl2PKI-AC"
   finalDefault="extension">

<xs:import namespace="urn:oasis:names:tc:SAML:2.0:ac"

 schemaLocation=
 "clipcode-saml-schema-authn-context-owl2pki-0.1.xsd" />
```

```
<xs:annotation>
    <xs:documentation>
Authentication context to be use with OWL2PKI certificates
    </xs:documentation>
</xs:annotation>
<xs:complexType name="AuthnContextDeclaration">
    <xs:complexContent>
        <xs:restriction
base="ac:AuthnContextDeclarationBaseType">
      <xs:sequence>
<xs:element ref="ac:Identification" minOccurs="0" />
<xs:element ref="ac:TechnicalProtection" minOccurs="0" />
<xs:element ref="ac:OperationalProtection" minOccurs="0" />
<xs:element ref="AuthnMethod" />
<xs:element ref="ac:GoverningAgreements" minOccurs="0" />
<xs:element ref="ac:Extension" minOccurs="0"
                                    maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="ID" type="xs:ID" />
        </xs:restriction>
    </xs:complexContent>
</xs:complexType>
<xs:element name="AuthnMethod" type="AuthnMethodType" />
<xs:complexType name="AuthnMethodType">
    <xs:complexContent>
        <xs:restriction base="ac:AuthnMethodBaseType">
```

```
                    <xs:sequence>

                        <xs:element ref="AuthnMechanism" />

                        <xs:element ref="Authenticator" />

                <xs:element ref="ac:AuthenticatorTransportProtocol"
                  minOccurs="0" />

                <xs:element ref="ac:Extension" minOccurs="0"
                  maxOccurs="unbounded" />

                    </xs:sequence>

                </xs:restriction>

            </xs:complexContent>

        </xs:complexType>

        <xs:element name="AuthnMechanism"
                type="PasswordAuthnMechanismType"/>

        <xs:complexType name="PasswordAuthnMechanismType">

            <xs:complexContent>

                <xs:restriction
                base="ac:PrincipalAuthenticationMechanismType">

                    <xs:sequence>

                        <xs:choice>

                            <xs:element
                             ref="ac:RestrictedPassword" />

                        </xs:choice>

                    </xs:sequence>

                    <xs:attribute name="preauth"
                        type="xs:integer"
                        use="optional" />

                </xs:restriction>

            </xs:complexContent>

        </xs:complexType>
```

```
<xs:element name="Authenticator" type="AuthenticatorType" />
    <xs:complexType name="AuthenticatorType">
        <xs:complexContent>
            <xs:restriction base="ac:AuthenticatorBaseType">
                <xs:choice>
                    <xs:element ref="DigSig" />
                </xs:choice>
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="DigSig" type="DigSigType" />
    <xs:complexType name="DigSigType">
        <xs:complexContent>
            <xs:restriction base="ac:PublicKeyType">
                <xs:attribute name="keyValidation" fixed=
"http://clipcode.com/Identity/20100302/Owl2PKI-AC"/>
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

Protocol Bindings

Section 3.1.2.1 of the SAML Bindings specification states:

Unless otherwise specified, in any SAML binding's use of SSL 3.0 [SSL3] or TLS 1.0 [RFC2246], servers MUST authenticate to clients using a X.509 v3 certificate. The client MUST establish server identity based on contents of the certificate (typically through

examination of the certificate's subject DN field, subjectAltName attribute, etc.).

With the introduction of configurable certificate format support in TLS that includes supports Owl2 certificates, this should now be changed to:

Unless otherwise specified, in any SAML binding's use of SSL 3.0 [SSL3] or TLS 1.0 [RFC2246] or TLS 1.1, servers MUST authenticate to clients using a certificate valid for the SSL/TLS version, such as X5.09 certificate or an OWL2 certificate. The client MUST establish server identity based on contents of the certificate (typically through examination of the certificate's subject Uri – for OWL2 certificates, or subject DN field, subjectAltName attribute, etc. for X.509 certificates).

Claims

1. A method for securing a computer network comprising the steps of:

   defining an OWL2 digital certificate to store the principal's identity and public key,

   modifying a security protocol to access the principal's identity and public key from said certificate,

   connecting computer devices via a computer network, and

   securing said network using said security protocol.

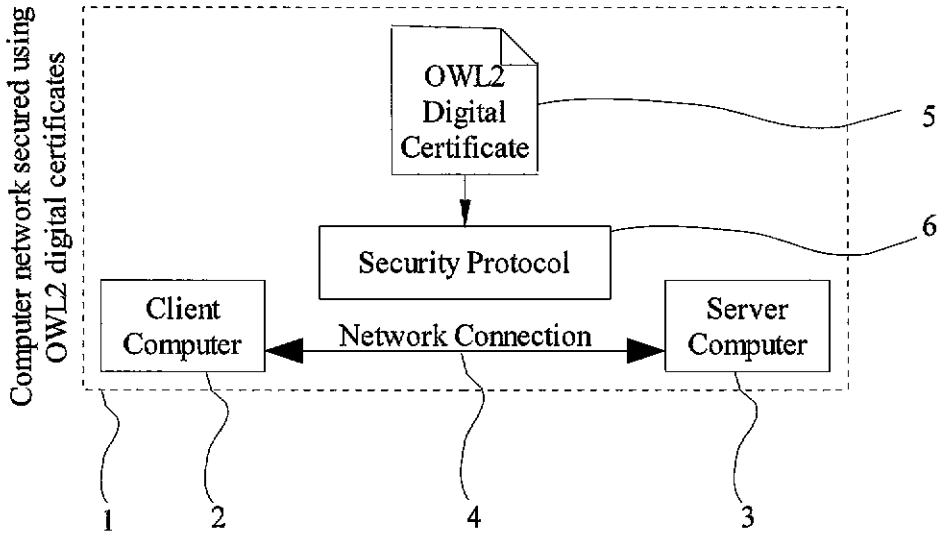
2. A method as claimed in claim 1, wherein the security protocol is Transport Layer Security (TLS).

3. A method as claimed in claim 1, wherein the security protocol is XML Digital Signatures.

4. A method as claimed in claim 1, wherein the security protocol is XML Encryption.

5. A method as claimed in claim 1, wherein the security protocol is the Security Assertion Markup Language (SAML).

Fig. 1


Computer network secured using OWL2 digital certificates
OWL2 Digital Certificate
5
Security Protocol
6
Client Computer
Network Connection
Server Computer
1
2
4
3