VeriSign®

White Paper

XML Key Management

XML Trust Services

Contents

Executive Summary

XML (Extensible Markup Language), the flexible data framework that allows applications to communicate on the Internet, has become the preferred infrastructure for e-commerce applications. All of those transactions require trust and security, making it mission-critical to devise common XML mechanisms for authenticating merchants, buyers, and suppliers to each other, and for digitally signing and encrypting XML documents like contracts and payment transactions.

XML Trust Services—a four-component suite of open specifications for application developers developed in partnership with industry leaders including Microsoft, Ariba, webMethods, and Netegrity—makes it easier than ever to integrate a broad range of trust services into B2B and B2C applications. XML complements Public Key Infrastructure (PKI) and digital certificates, the standard method for securing Internet transactions.

To simplify the integration of PKI and digital certificates with XML applications, VeriSign, Microsoft, and webMethods have created the open **XKMS** (XML Key Management Specification) specification. Developers can take advantage of XKMS to integrate authentication, digital signature, and encryption services, such as certificate processing and revocation status checking, into applications in a matter of hours—without the constraints and complications associated with proprietary PKI software toolkits. With XKMS, trust functions reside in servers accessible via easily programmed XML transactions. Developers can allow applications to delegate all or part of the processing of XML digital signatures and encrypted elements to VeriSign, shielding the application form the complexity of the underlying PKI.

I. Background

Over the last two years, the **eXtensible Markup Language** (**XML**) has rapidly emerged as the *lingua franca* for electronic data exchange in business applications.

In parallel, **Public Key Infrastructure** (**PKI**) and digital certificates have continued their expanding adoption by net marketplaces, Internet merchants, and suppliers as the *de facto* strong foundation for authenticating users, Web sites, and business partners.

As XML has gained momentum as the preferred format for exchanging business information on the Web, the need for standard mechanisms for applications to provide entity authentication and privacy services for XML documents has grown. Today's marketplaces are therefore eager for XML and PKI to work together in fulfilling the widely held expectations for cryptographically secure, XML-coupled business applications.

A. The Need for XML Trust Infrastructure

For XML to become the premier language of electronic commerce, it must include standard mechanisms for digitally signing and encrypting XML elements. Such features are especially important for documents that represent sensitive content or financial commitments such as contracts, price lists, quotations, or payment transactions. Because digital signing and encryption both involve cryptographic keys—and decisions whether to trust them—one is quickly led to consider the **trust management** issue for XML applications.

A joint IETF-W3C working group has just completed a proposed standard for **XML Digital Signatures**, but its charter expressly limits its work to that specific area. Recently, a new working group within the W3C has also been chartered to develop standards for **XML Encryption**. This paper gives an overview of how VeriSign is working to build on these emerging standards foundations to provide a **higher-level trust infrastructure** for XML as part of its XML Trust Services.

II. VeriSign XML Trust Infrastructure Goals

The well-known simplicity of XML to provide portability of data between disparate business systems contrasts with the complexity of traditional PKI implementation. Therefore, a key architectural goal in the **XML Key Management Specification** and XML Trust Services is to shield XML application developers from the complexity of traditional PKI implementation.

The XML Key Management Specification permits **delegation of trust processing** decisions to one or more specialized trust processors. It enables XML-based systems to rely on complex trust relationships without the need for complex or specialized end-entity PKI application logic on the client platforms where XML processing is taking place.

By allowing the XML application developer to delegate trust decisions to specialized trust processors, the architecture shields XML client implementation from the complexity of the underlying PKI.

XML applications that incorporate digital signature or encryption functionality reap the following benefits from the VeriSign XML Trust Infrastructure:

- **No need to delay PKI deployment pending client support.** The XML Key Management Specification moves the complexity of PKI and trust processing to server side components instead.
- **Be "future proof" against new PKI developments.** Application developers benefit when the impact of future PKI developments is restricted to server-side components.
- **Allow mobile devices to access full-featured PKI** through ultra-minimal footprint client interfaces.

III. Architectural Overview

The XML Key Management Specification is based on a comprehensive, open, and standards-based approach to adding trust processing to cryptographic XML applications. The architecture has been designed to complement the emerging W3C standards activities in the XML Digital Signature and XML Encryption Working Groups.

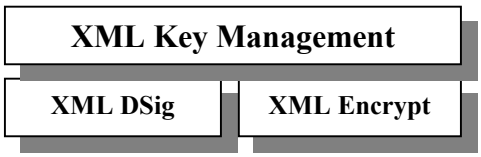

Figure 1: Relationship between XML DSig, XML Encrypt, and the XML Key Management Services

XKMS describes mechanisms that allow XML-aware applications to easily leverage public-key infrastructure in support of digitally signed and/or encrypted XML documents. The primary objective is to allow a user of a public key—when used to verify a digital signature or encrypt data—to locate the required key and to associate naming or attribute information with the holder of the corresponding private key.

There are two major subparts of the XML Key Management Specification:

- Central to the XML Trust Infrastructure is the **XML Key Information Service Specification (X-KISS)**, which defines protocols to support the processing, by a relying party, of Key Information associated with a XML digital signature, XML encrypted data, or other public key usage in an XML-aware application. Functions supported include locating required public keys given identifier information, and binding of such keys to identifier information.
- The **XML Key Registration Service Specification (X-KRSS)** defines protocols to support the registration of a key pair by a key pair holder, with the intent that the key pair subsequently be usable in conjunction with the XKMS.

Each of these protocols describes protocol exchanges that consist of a simple request and response exchange with a Trust Service.

The XML interface protocols of the XML Trust Infrastructure do not require any particular public key infrastructure (such as X.509), but are instead designed to be compatible with such infrastructures, including such traditional standards such as X.509v3, SPKI, and PGP.

We give a brief technical overview of X-KISS and X-KRSS in turn.

A. X-KISS: Technical Overview

The XML Key Information Service Specification (X-KISS) helps XML developers obtain cryptographic key information associated with signed and/or encrypted XML documents. To see how X-KISS works, it's useful to look at an example.

Suppose that a developer's application has just received a digitally signed, W3C XML Signature Specification -compliant XML document, and the application needs to verify the digital signature. Within the context of the XML Signature specification, **KeyInfo** is an optional element that enables the recipient(s) to obtain the key needed to (locally) validate the signature.

```
<element name="KeyInfo">
<complexType>
 <choice maxOccurs="unbounded">
  <any processContents="lax" namespace="##other"
          minOccurs="0" maxOccurs="unbounded"/>
  <element name="KeyName" type="string"/>
  <element ref="ds:KeyValue"/>
  <element ref="ds:RetrievalMethod"/>
  <element ref="ds:X509Data"/>
  <element ref="ds:PGPData"/>
  <element ref="ds:SPKIData"/>
  <element name="MgmtData" type="string"/>
 </choice>
 <attribute name="Id" type="ID" use="optional"/>
 </complexType>
</element>
```

Figure 2: The XML Schema for KeyInfo as defined by XML Digital Signature

Such a KeyInfo element may contain keys, names, certificates and other public key management information, such as in-band key distribution or key agreement data. The XML Digital Signature specification defines a few simple types, but applications are often left to define their own key identification and exchange semantics within this element type through the XML-namespace facility.

Using X-KISS, the developer's application may send such a KeyInfo element that it does not have the ability to parse to a Trust Service for processing—for example, an embedded

X.509 certificate (the X509 element) could be sent to the server component to be parsed and its embedded public key information "pulled apart" by the trust service.

In this way, complex certificate processing logic is moved to server side components, achieving the key goal of the XML Trust Infrastructure application.

For more information see the X-KISS Specification itself.

B. X-KRSS: Technical Overview

Existing certificate management protocols—such as those defined by PKIX—either focus on one support of a single part of the certificate lifecycle (typically certificate issuance) or are too complex for the lightweight, small footprint XML applications that are being deployed today.

The goal of the X-KRSS Specification is to respond to the need for a complete, XML client-focused key lifecycle management protocol.

X-KRSS supports the entire certificate lifecycle in a single, compact specification

- Key Registration
- Key Revocation
- Key Recovery

We consider each of the operations in turn.

1. X-KRSS: Registration, Revocation, & Recovery

In the registration phase, an XML application key pair holder registers its public key with trusted infrastructure via a **registration server**. The public key is sent to the registration server using a digitally signed request specified by KRSS (XML Type <Register>). Such a Register request may also optionally include:

- **Name and attribute information**: The holder is requesting the infrastructure to store the name and attribute information and subsequently deliver it to an entity using the public key. The infrastructure may register this information as requested or may modify or replace it, in accordance with applicable policy.
- **Authentication information**: Information that the infrastructure may need to authenticate the key holder, but which is not generally supplied to public key users.
- **Proof-of-possession of private key**: When registering a digital signature key pair, using the corresponding private key to digitally sign the registration request acts as proof-of-possession. However, with other types of key pairs, a separate proof-of-possession field may be needed.

The registration server responds with an XML formatted confirmation response (XML Type `<Confirm>`, which indicates status of the registration (accepted, rejected, or pending) and a confirmation of name and attribute information registered with the public

key. Except in the case of rejection, a key pair identifier is returned in the Confirm for subsequent referencing purposes.

The registration will typically be preceded by generation of the key pair in the key pair holder system.

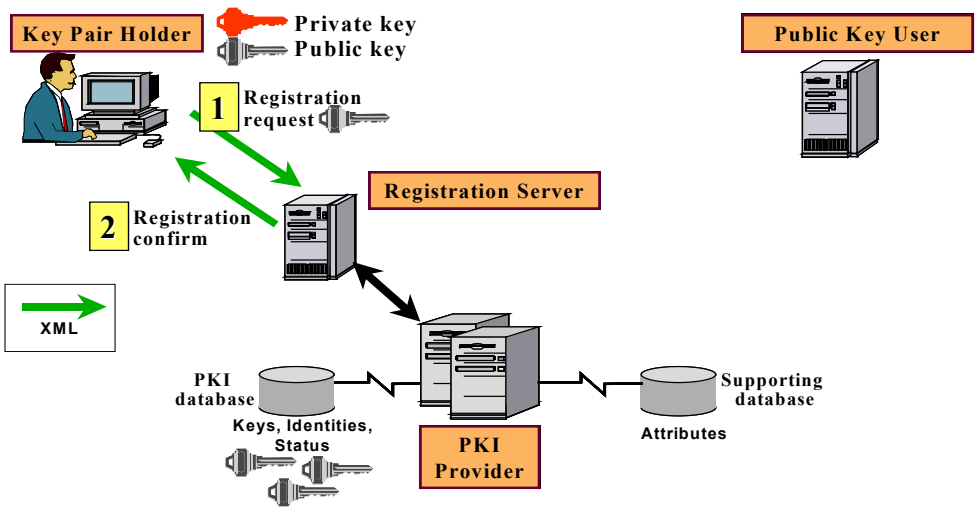

Figure 3: X-KRSS Key Registration

Revocation is handled via a similar protocol.

Finally, the use of desktop (i.e. file system) encryption—as well as more broad XML client encryption applications—mandates some form of **key recovery** provision. For historical reasons this usage is not supported by standardized protocols. In X-KRSS, such support is built in.

KRSS may be also used as a lightweight interface to an underlying PKI such as X.509v3. Finally, KRSS may also be used to support registration of 'certificate-less' keys in applications where a full feature PKI is not required.

IV. Summary: Responding to Today's and Tomorrow's Applications

The VeriSign XML Trust Infrastructure delivers a fully functional PKI environment where only lightweight cryptography and simple XML-based runtime logic is available to client components.

Today's XML applications—for example, on mobile wireless devices—often operate in highly constrained memory environments. A 1Mb footprint for PKI client code is simply unacceptable.

By adopting the VeriSign XML Trust Infrastructure, clients can reduce runtime footprints to absolute minimum while preserving Interoperability with X.509—for example may be required to support widely deployed standards such as SSL or S/MIME.

We conclude with a reiteration of the key benefits of the architecture.

Reduce Development, Deployment Times
In this way the following traditional applications benefits of XML development are preserved:

- **Provide a "pure XML," developer-friendly syntax.** By avoiding the introduction of PKI toolkits on the client side, the trust infrastructure permits developers to work in a simple XML environment.
- **Allow rapid implementation of trust with standard XML toolkits.** The only client logic beyond traditional XML parsing runtimes is cryptographic support for XML Digital Signatures and XML Encryption.
- **No need for proprietary plug-ins** to support enterprise PKI

Simplify Enterprise Management

- Single point of administration for entire enterprise
- Manage trust relationships as relationships between enterprises rather than relationships between individual employees

V. For More Information

To access the XKMS specification and learn more about VeriSign's XML Trust Services, see http://www.verisign.com/developer/xml/index.html

VERISIGN, INC.
1350 CHARLESTON ROAD
MOUNTAIN VIEW, CALIFORNIA 9404
WWW.VERISIGN.COM