

## Trust Assertion XML Infrastructure

Phillip Hallam-Baker  
*VeriSign Inc.*

### Abstract

The Trust Assertion XML Infrastructure (TAXI) is described. TAXI is a PKI research project that had the objective of developing technology that would assist the deployment of PKI. Parts of the TAXI architecture have since been realized in open standards, notably the XKMS [XKMS] and SAML [SAML] specifications, other parts of the TAXI architecture such as XTAML [XTAML] and XKASS [XKASS] have been published as research notes for public review and possible standardization at a later date. The paper describes the architectural principles underlying the design decisions taken in these specifications.

## 1 Cryptography and Trust

Public Key cryptography permits secure communication to be established between any parties provided only that each has trustworthy knowledge of the public key of the other. The means by which that trustworthy knowledge is obtained is known as Public Key Infrastructure (PKI).

PKI secures the interface between the abstract world of electronic communications and the concrete offline world. PKI is complex and subtle because the world is complex and subtle.

The deployment of PKI in the real world has been subject to numerous disputes about architecture, factional schisms and political intrigues. While some of these disputes have technical merit few have advanced the cause for PKI. The quest for the perfect PKI has too often been the enemy of deployment of a good PKI.

This paper describes the Trust Assertion XML Infrastructure (TAXI), a research project that was undertaken in the summer of 2000 with the objective of developing technology that would assist the deployment of PKI. Parts of the TAXI architecture have since been realized in open standards, notably the XKMS [XKMS] and SAML [SAML] specifications, other parts of the TAXI architecture such as XTAML [XTAML] and XKASS [XKASS] have been published as research notes for public review and possible standardization at a later date.

Standards documents intended to describe a normative specification should not provide any discussion of the architectural principles. This paper is intended to make good this omission and

to explain how the different components of the TAXI architecture were intended to fit together. In view of the developments since the original TAXI architecture was developed this paper makes use of the terminology and concepts used in the XKMS and SAML specifications rather than those of the original documents.

### 1.1 Certificates

The traditional model of Public Key Infrastructure is based on the model proposed by Lauren Kohnfelder's in 1978 [Kohn78]. An email user A may obtain the public key of email user B by consulting a directory. The need for online access to the directory could be avoided by signing individual directory entries to form a 'certificate'.

The PKI most closely associated with certificates is X.509 [X.509], which realizes the Kohnfelder model in the context of the X.500 directory [X.500]. The influence of the Kohnfelder model is also seen in PKI proposals that attempt to escape from the certificate model including PGP [PGP], SPKI [SPKI] and even DNSSEC [DNSSEC]. All share the basic principle of using signed data to bind the public key of a user to a sign that identifies them. Regardless of whether the signed data is called a 'certificate', a 'key signing' or a 'signed record', the differences in how the signed data is generated and used are considerably less important than the similarities.

The X.509 specification was originally developed as a part of the OSI network standard developed as a joint standard of ISO/IEC and the ITU. As increasing use was made of the X.509 standard by Internet protocols an IETF working group was formed to describe the use of X.509 in

that context. Over time the IETF Public Key Infrastructure X.509 (PKIX) [PKIX] group has specified additional protocols that extend the use of X.509 so that the terms X.509 and PKIX are often used interchangeably.

### 1.1.1 Trust Topology and Names

For many years the PKI debate centered on the topology of trust. Certificate hierarchies, heterarchies and Webs of Trust were advanced each with merits and demerits. In the authors view this debate obscured rather than clarified the issues that should have been at the center of the debate, namely:

- The ability of relying parties to locate a public key for a particular purpose
- The ability of relying parties to locate a trust path that validates a public key
- The ability of relying parties to control the trust criteria that are applied

A highly constrained trust topology such as a strict hierarchy makes the process of locating keys and trust paths easier to implement and manage than a less constrained topology. An unconstrained topology in which all participants are peers appeals to the spirit of egalitarianism by obviating the need (but not precluding the existence) of centralized control.

While the trust topology debate has continued, real world deployment of PKI has largely converged on a single model in which multiple trust providers issue certificates and relying parties decide which trust providers to rely on.

### 1.1.2 Naming

A certificate binds a public key to a name; the question of naming has thus been at the center of many PKI debates. In the DNSSEC and the original X.509 architecture the certificate hierarchy precisely matches that of a hierarchical naming scheme.

A name is a signifier that bears only a conventional relationship to the signified [Sebok]. It follows therefore that if the trust providers are to be true peers they must have equal capacity to define naming conventions. This principle is embodied in the Rivest and

Lampson SDSI paper [SDSI] that introduces a naming scheme in which all names are relative and “Alice” becomes “The person who Doug calls Alice”.

This relativist naming scheme proposed appears unlikely to provide much value in practice. While all names are ultimately subjective the ability to communicate depends on the parties having established a vocabulary of shared terms. While names are defined in many ways and there is ultimately no single authority that is responsible for assigning names there is in practice little ambiguity. Names are chosen to facilitate communication. If Bob, Doug and Carol are in regular communication and each use the name ‘Alice’ to refer to a different individual a means of resolving the ambiguity will be found. It is more likely that the convention chosen will involve a property of the people called Alice that distinguishes them from each other than the speaker.

### 1.1.3 Beyond Email

The certificate-based model of PKI was developed to address the problem of sending secure email messages within the specific constraints of the early ARPANET. The X.509 specification was originally designed to support secure email in the context of the X.500 directory and X.400 mail. X.509 has since been extended to meet many requirements that were originally out of scope. In the process the X.509/PKIX specifications have grown larger and more complex.

Despite their complexity, the X.509/PKIX specifications are in several ways incomplete. In a commercial environment it is far more likely that Alice would issue a check in error than lose her safe key. PKIX provides no fewer than four methods of determining the validity status of a certificate. No mechanism is provided to determine the validity status of a signed document.

## 1.2 Client Complexity

One of the principal objections made to the deployment of traditional PKI is the complexity of the specification. Full support for the industry standard X.509/PKIX specification requires a very large and complex client implementation

that very few applications support directly (figure 1).

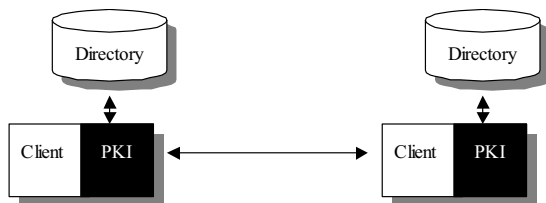


Figure 1 Client Complexity in Traditional PKI is High.

While PKI is ubiquitously supported in mainstream email, browser and operating systems software, ‘sophisticated’ PKI features such as cross-certification, OCSP etc. are not. Such features are typically only supported by PKI ‘plug-ins’ provided by third party PKI vendors. Plug-ins of this type have proved expensive to deploy and maintain, particularly since each PKI client must be configured with the location of the local PKI repository. A new plug in deployment is required each time there is a change to the PKI configuration, support for new PKI features is required or the base application is upgraded.

### 1.2.1 Historical Complexity and Necessary Complexity

Part of the complexity of PKIX is due to the process by which the specification developed. The CRL specification was developed as a certificate blacklist mechanism. As the number of certificates grew, CRLs grew to unacceptable size leading to various extensions to mitigate the problem. At the same time the Online Certificate Status Protocol (OCSP) was developed to provide real-time reporting of certificate status. Despite the close relationship between CRLs and OCSP the data formats and protocols associated with each are separate.

Although much of the complexity of PKIX could be reduced through a thorough re-design process, the main reason that the PKIX specification is complex is that it attempts to address a complex problem. In many instances it has been the attempt to address a complex problem with a too-simple solution that has led to complexity.

PKI is complex because trust relations in the real world are complex and cannot necessarily be

reduced to a series of standardized machine-readable data formats. The choice for a PKI architect therefore is not whether there is complexity but how it is managed and where it is placed.

### 1.2.2 Directory as Certificate Repository

The close relationship between the X.500 and X.509 specifications led many to assume that digital certificates ‘should’ be stored an X.500 or LDAP directory. This assumption leads to the conclusion that the deployment of a PKI at either a local or global level is dependent on the deployment of a directory.

While many companies have deployed local directories these are almost without exception considered internal resources whose contents are company confidential.

While the X.500 or LDAP protocols might form a basis for a certificate retrieval protocol, the directory data model is not. The underlying principle of the directory data model is that the directory server supports a generic query mechanism to a hierarchical data structure. This model is ill suited to the needs of a certificate repository that is servicing highly specific queries against a heterarchical PKI topology.

### 1.2.3 The Client Deployment Trap

The problem that appears to have brought deployment of new PKIX features to a halt is the client deployment trap. For a PKI feature to be useful every client must first support it. For mainstream application vendors to support a feature it must first be useful. None of the mainstream PKI enabled applications (Netscape Communicator, Microsoft Outlook, Lotus Notes) provide native support for cross-certification. The feature would have little value until it was widely supported and will not be supported in any degree until it provides value.

### 1.2.4 The End to End Principle

The *end-to-end principle* is one of the key architectural principles of the design of the Internet. Under the end-to-end principle the network core is as simple as possible, a packet switching network that provides no guarantees as to the reliability or order of packet delivery.

Sophistication is achieved at the ends of the communication where acknowledgement of received packets is made and packets are reassembled into order.

The end-to-end principle has been applied to security to establish the doctrine that security enhancements should be applied at the 'ends' of the communication. For example an email message should ideally be secured from the sender to the recipient.

The difficulty raised by this interpretation of the end-to-end principle is that the ends of the communication are devices while the ends of the trust relationship are people and/or organizations. The sophisticated management of trust relationships is complex and subtle and has proved to be beyond the level of complexity that developers of client applications will tolerate.

Properly understood, the end-to-end principle argues that complexity must be eliminated where possible and where it cannot be eliminated must be confined to those parts of the network infrastructure that are capable of supporting it.

### **1.2.5 Trust Management**

The use of cryptography and PKI typically appeals to individuals of independent character. As a consequence PKI architectures have emphasized the role of individual choice in the configuration of their trust relationships. This approach is a poor match to enterprise needs where trust relationships between enterprises are by definition established at an enterprise level.

The PKI approach that requires PKI configuration to take place at the client end does not meet the needs of enterprises attempting to manage their trust relationships at the enterprise level. The client centric model of PKI requires that all trust relationships be expressible in a data format supported by the client and that the client support all the necessary location and retrieval protocols.

As the number of PKI enabled devices increases the trust management problem increases. Even highly motivated individuals managing their personal devices are unlikely to want to maintain their trust configuration separately on the laptop, desktop, handheld, mobile phone etc.

## **1.3 New Challenges**

Despite the numerous objections made against it, the deployment of PKI has been a success by most ordinary measures. Millions of people use PKI each day, in most cases without being aware that they have been using it. PKI is already established that provides to anyone with a need secure email, a secure means to make payments over the Internet, a Virtual Private Network.

Although PKI has succeeded by most ordinary measures it has failed against its perceived potential. Security remains an optional extra used in cases of need, PKI enabled cryptography has not yet become the ubiquitous default.

This qualified success poses a considerable challenge to the deployment of alternative approaches. Attempts to replace X.509 completely have largely failed completely or been confined to a single narrow area of application. New PKI infrastructure can only be justified if it enables new applications of cryptography that were impossible or impractical with the existing infrastructure.

### **1.3.1 Constrained Devices**

As the cost of processing power has decreased the number of devices with embedded CPUs has increased dramatically. Far from eliminating the constraints of CPU power on PKI, improvements in processor performance have increased them as manufacturers attempt to embed PKI into mobile phones, personal organizers and all manner of network devices.

In addition to lacking the processing capability to support sophisticated a sophisticated PKI client implementation, constrained devices often lack user interface capabilities that are appropriate to the task. The task of adding a root certificate into a PC web browser is supported by a rich user interface that presents the user with all the necessary information. While it is possible to add a root certificate into a mobile phone with a 20-button keypad and a 20-character display, it is unlikely that the process can be made acceptable to many consumers.

X.509 has been adapted to meet the constraints of wireless use in the WAP specification [WAP]. The modifications include the use of compressed 'WAP Certificates' and a messaging protocol

that uses a certificate identifier in place of the certificate itself to save bandwidth on constrained links.

### 1.3.2 Financial Transactions

Financial services applications operate under requirements that are quite distinct from the email application that has traditionally formed the PKI paradigm. Unlike email applications, financial services applications can depend upon the availability of network connectivity at all times. Financial services have a well-defined trust model that is backed by regulation, insurance and contracts that define the liabilities of the parties and operate in an environment in which the precise timing of operations can transfer liability from one party to another.

As a result of these different constraints financial services applications have traditionally been at the cutting edge of PKI, leading to developments such as the Online Certificate Status Protocol (OCSP) [OCSP].

The Identrus architecture [Identrus] applies PKIX and OCSP to provide real time validation of public keys in the context of the ‘four-corners’ model common to many financial transactions. This architecture demonstrates a significant limitation of the use of the certificate model designed to support offline messaging to an online application. Relying applications must support OCSP processing *in addition* to certificate processing, the PKIX architecture does not support the use of an online protocol *instead* of certificate processing.

### 1.3.3 Web Services

Web Services [SOAP] are a set of industry standards based on XML that allow applications running on different machines to exchange data. The goal of Web Services is to reduce or eliminate interface costs, the cost of exchanging data between computer systems. Interface costs represent two of the largest costs of running information systems:

- Computers generate messages that are sent to the customer by letter post or fax and entered manually into another computer system

- The largest cost in the deployment of a new software system is often interfacing the new system to the legacy systems already deployed.

Web Services offer the promise of enabling a new and more cost effective IT strategy in which communications that currently require human intervention are automated. Web Services have the potential to change the way that Enterprises communicate both internally and externally.

While X.509 certificate meet some of the PKI requirements of Web Services the use of an ASN.1 based PKI to support an XML based messaging infrastructure is unsatisfactory. While the overhead required to support ASN.1 and X.509 on a server platform is quite reasonable, the same overhead is unreasonable for many of the intended clients.

## 2 Trust Assertion XML Infrastructure

The TAXI architecture is based on the following principles:

- Minimize the complexity of client deployment, configuration and management.
- Separate the client implementation from the structure of the underlying PKI.

The TAXI architecture makes extensive use of the XML Signature [XML-SIG] <KeyInfo> element that allows a public key to be identified using practically any means including:

- The Public Key parameters (e.g. RSA modulus and exponent)
- Any naming scheme (e.g. URI, X.500 Common Name)
- X509 Certificate, CRL, OCSP token
- SPKI, PGP key signing.
- A URL for the retrieval of any of the above

### 2.1 Architecture

The TAXI architecture is divided into four tiers that represent increasing complexity from the first to the fourth as follows:

#### Tier 1 Location

The location service is a Web service that allows a client to locate information concerning a public key analogous to

the directory function in the PKIX model

**Tier 2 Validation**

The validation service is a Web service that allows a client to delegate both the retrieval and processing of public key information. The validation service is analogous to a highly extended form of the PKIX OCSP protocol.

**Tier 3 Trust Assertion**

A trust assertion contains a unique identifier, one or more statements, conditions and advice. Trust assertions combine the roles of PKIX certificates, attribute certificates and in some instances signed documents themselves.

**Tier 4 Status Assertion**

A Status Assertion is an assertion that makes a statement about the validity of one or more other assertions. Status Assertions combine the roles of CRLs and OCSP in the PKIX model.

**2.2 Specifications**

**2.2.1 XKMS**

The XKMS specification consists of a registration protocol and an enquiry protocol. These protocols may be used independently.

The XKMS enquiry protocol is the XML Key Information Service Specification (X-KISS) which supports two service tiers:

**Tier 1: Locate**

The client sends one <KeyInfo> element to the service and requests that the trust service provide a <KeyInfo> element that identifies the same key but is in a different format (e.g. X.509 certificate converted to key parameters).

**Tier 2: Validate**

The trust service validates the trustworthiness of the information returned according to service specific criteria.

**2.2.1.1 Tier 1 Location**

A client receives a signed XML document. The <KeyInfo> element in the signature specifies a retrieval method for an X.509 certificate. The client lacking the means to either resolve the URL or parse the X.509 certificate to obtain the public key parameters delegates these tasks to the trust service (Figure 2).

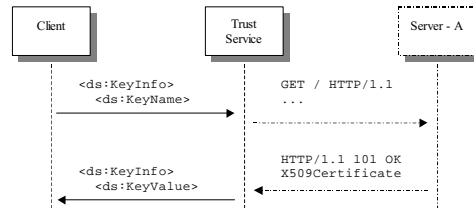


Figure 2: Key Location Service

**2.2.1.2 Tier 2 Validation**

The Validate service allows a client to delegate all trust processing functions to a trust service. As with the Locate service the client creates a query that specifies the information the validation service is to locate. Unlike the location service however the validation service is responsible for ensuring the trustworthiness of the data returned before relying upon it.

A client receives a signed XML document and queries the trust service to determine whether the signing key is trustworthy. In this case an X.509 certificate authenticates the signing key. The Trust Service builds a certificate trust path, then validates each certificate in the path against the relevant Certification Revocation List. The client is shielded from this complexity however and the trust service returns only the information of specific interest to the client; the key parameters, the data bound to the key and the validity of the binding (Figure 3).

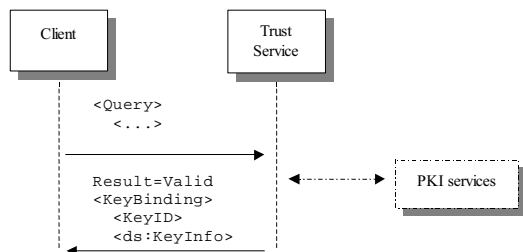


Figure 3 Key Validation Service

Delegation of trust processing functions to a trust service makes enterprise-wide control and oversight of PKI configuration possible. This is essential in Business-to-Business applications where the important trust relationships are between enterprises and not between individuals or the applications they use.

## 2.2.2 X-KRSS

XML Key Registration Service Specification (X-KRSS) defines a protocol for a trust service that accepts registration of public key information. Once registered, the public key may be used in conjunction with other web services including X-KISS.

X-KRSS is designed to support all of the functions associated with the public key lifecycle:

**Registration.** The registration function supports registration of an association of a public key and additional data (such as a name) to create a 'key binding'. Private keys may be generated either locally by the client (desirable for signing keys) or by a central key generation service (desirable in cases where key recovery is supported). Requests may be authenticated with either a limited use shared secret or a digital signature.

**Renewal.** XKMS allows a PKI to be operated without digital certificates ever being issued, eliminating the need for certificate renewal. In cases where certificates are issued by the underlying PKI renewal processing may be performed automatically without the need for client interaction.

**Revocation.** An authorized party may request that the trust service revoke a key binding. This may be necessary because the key has been

compromised or because information contained in the key binding is incorrect.

**Recovery.** Private key recovery is essential when an end user has lost their private key and requires access to their encrypted data. The X-KRSS recovery function provides an authenticated means of re-issuing a private key to a user.

X-KRSS may be configured hierarchically in the manner of a Local Registration Authority. This allows a registration request to be authenticated by a local trust service then passed on to another trust service where actual processing is performed.

## 2.2.3 SAML

The Security Assertion Markup Language [SAML] specifies both the TAXI Tier 3 trust assertion framework and specific assertion statements to support federated authentication and authorization applications.

Each trust assertion is encoded in a common XML package, which at a minimum consists of:

### Basic Information.

Each assertion must specify the version of the SAML assertion syntax, a unique identifier that serves as a name for the assertion, a unique identifier for the issuer and the time instant of issue.

### The Asserted Statement(s)

The statement(s) that are asserted by the issuer of the assertion.

In addition an assertion may contain the following additional elements:

### Conditions.

The assertion status may be subject to conditions. The status of the assertion might be dependent on additional information from a validation service. The assertion may be dependent on other assertions being valid. The assertion may only be valid if the relying party is a member of a particular audience.

### Advice.

Assertions may contain additional information as advice. The advice element

MAY be used to specify the assertions that were used to make a policy decision.

Relying applications may ignore advice elements but are required to understand all the conditions elements in an assertion if they are to rely on it.

SAML defines three Assertion Statements as follows:

#### **Authentication Assertion**

An authentication assertion contains a statement made by the issuer that asserts the subject was authenticated by a particular means at a particular time.

#### **Authorization Decision Assertion**

An authorization decision assertion contains a statement made by the issuer that asserts the request for access by the specified subject to the specified object has resulted in the specified decision on the basis of some optionally specified evidence.

#### **Attribute Assertion**

An attribute assertion contains a statement made by the issuer that asserts the specified subject is associated with the specified attribute(s).

### **2.2.4 XTAML**

One of the most common objections made to the XKMS trust service model is that it does not provide a means of establishing and maintaining the trust relationship between the client and the trust service. XKMS cannot eliminate the need to implement X.509 if a certificate is still required to secure this trust relationship. XTAML is designed to meet this need in the context of a large scale PKI deployment in which a root of trust might be embedded in a large number of devices and consequently there is a need to be able to manage the private keys associated with the root of trust itself in a highly controlled offline environment that is independent of the online private keys used to authenticate actual Web Services transactions.

By design XTAML supports only the most limited delegation model. In the X.509 model a certificate signing certificate may be used to delegate a signing authority that is restricted to particular domains and/or certification policy. In contrast the XTAML delegation model provides

only 'all or nothing' delegation required to support the requirements of online/offline key management.

The XML Trust Axiom Markup Language (XTAML) defines SAML Trust Assertions that support the management of trust axioms. A trust axiom is a 'root of trust' analogous to a root certificate in a certificate based PKI. An important application of trust axioms is managing the trust relationship between a client and a trust service.

XTAML defines SAML statement elements for specifying axiomatic and delegate keys and for asserting the validity status of another assertion. A new condition element is defined that makes the validity status of an assertion dependent on online verification. Two new advice elements are defined to allow an assertion to provide advice on the reissue of the assertion and for issue of related assertions.

### **2.2.5 XKASS**

Another objections made to the use of XML Signature to authenticate Web Service requests and responses such as XKMS is the processing overhead required to create and verify digital signatures.

XKASS provides a means of using a lightweight Message Authentication Code (MAC) to authenticate Web Service messages by means of a shared secret established through a key agreement mechanism. The design of the XKASS is similar to the Just fast Keying [JFK] proposal made to the IETF IPSEC working group but requires only one round trip in the typical case instead of two made possible by a different approach to the handling of Denial of Service attacks.

### **2.3 Assertion Calculus**

The SAML specification defines a framework for encoding Trust Assertions but does not provide a general framework for defining the semantics of assertion statements. One means of attaching specific semantics to an assertion statement is by means of an assertion calculus that sets out the rules by which a set of assertions are reduced to specific actions in response to a query.

Each assertion calculus is specific to an application such as access control or management of financial instruments. The laws of the assertion calculus comprise a formal specification

For example in an access control application an attempt to access a resource would generate a query of the form:

[Q1] Is *Alice* granted *Read* access to the *Accounts file*?

Given the assertions:

[A1] *Alice* is a member of the *Finance* group

[A2] The *Finance* Group is granted *Read* access to the *Accounts file*

The query may be answered by applying the rule

[R1] IF (*P* is a member of the *Q* group) AND  
(The *Q* group is granted *X* access to *Y*)  
THEN  
*P* is granted *X* access to *Y*.

[P1] Applying R1 to A1 and A2, substituting *Alice* for *P*, *Finance* for *Q*, *Read* for *X* and *Accounts file* for *Y* we obtain:

[A3] *Alice* is granted *Read* access to the *Accounts file*

If the rules of the assertion calculus are labeled and specified in a suitable form the proof might be encoded in XML and attached to assertion A3 encoding the conclusion as advice.

While an application might employ a general purpose theorem

One of the principal advantages of the assertion calculus approach is that it allows an assertion generator to incorporate an integral verification step that independently verifies the correctness of the assertion by verifying the proof. Such a process is well within the capabilities of current formal methods tools, which cannot currently be said for the process of generating a proof in an arbitrary calculus.

### 3 Applications

The TAXI architecture reduces the complexity of a large number of PKI applications of which we present a representative sample only.

#### 3.1 Facilitating Deployment of Traditional PKI

The principal design goal for TAXI was to facilitate the deployment of traditional PKI by eliminating the need for a 'fat client' to support sophisticated PKI functionality. This goal is realized in the XKMS specification that allows a simple client to access a sophisticated PKI by means of the XKMS Web Service interface.

XKMS enables deployment of sophisticated PKI topologies such as the Federal Government Bridge CA [FBCA] without the need to deploy PKI plug-in applications to support the specific topology.

#### 3.2 Wireless LAN Configuration

A wireless LAN protocol such as 802.11b allows a user within range of an access point access to a LAN without the need for a physical connection. By eliminating the need for physical access a wireless LAN protocol removes a control that mitigates two significant security risks, first anyone within range might intercept network traffic, second unauthorized use of the network.

Recent analysis [Borisov01] of the 802.11b WEP cryptographic protocol [WEP] has demonstrated that the WEP protocol provides inadequate protection against the interception risk and little protection against the unauthorized use risk.

The risk of unauthorized use arises from the fact that every user of the network shares the same authentication key. The risk of unauthorized use could have been controlled if a sufficiently lightweight PKI had been available to the developers. For example XKMS might be used to permit network interface cards to be granted or denied access to the network on the basis of a private key embedded in the card.

#### 3.3 Negotiable Financial Documents

Many financial transactions are represented by the exchange of negotiable documents. In many cases these documents are bearer instruments.

For example a ship has fulfilled its obligations to the dispatcher when it discharges its cargo to the first person to present a valid bill of lading at the destination port.

Replacing paper documents with electronic representations offers many advantages including lower costs for the carrier and its customers. In addition an electronic instrument is more readily traded than one restricted to physical form.

A tier 3 trust assertion may be used to create an electronic bill of lading that tracks the current ownership of a specific asset (e.g. a cargo) and manages transfer of that asset from one owner to another by means of tier 4 status assertions. A potential purchaser of a cargo may determine if the seller is currently the owner of the cargo by validating the assertion stating ownership.

### 3.4 Commercial Registry

Many business applications involve some form of registry. For example in the US it is possible to gain security for a debt by registering a charge against assets of the debtor in a commercial registry.

A commercial registry does not normally require exceptional levels of availability, it is however essential that the registry ensure an exceptional level of data authenticity and persistence. Although the human interface to such a registry is likely to require customization to the applicable law, the type of assets registered, language, etc. the functions requiring exceptional levels of data authenticity and persistence are common to all registries.

Entries in the commercial registry may be represented by tier 3 trust assertions. Discharge or voiding of entries may be represented by means of tier 4 status assertions.

## 4 Acknowledgements

Thanks are due to Warwick Ford, Barbara Fox, Brian LaMachia, Jeremy Epstein, David Solo and Mack Hicks for their many helpful comments on the original TAXI research project. Thanks are also due to the members of the SAML and XKMS working groups who have helped to turn theory into practice, in particular Stephen Farrell, Shivram Mysore, Eve Maler,

Joe Pato, Jeff Hodges, Prateek Mishra, David Orchard, Hal Lockhart, Carlisle Adams, Tim Moses, Bob Blakely, Marlena Erdos, Scott Cantor, Chris McLaren, Krishna Sankar, Irving Reid, Daniel Ash, Joseph Reagle and Blair Dillaway.

## 5 References

- [DNSSEC] Eastlake, D. and C. Kaufman, *Proposed Standard for DNS Security*, RFC 2065, January 1997.
- [FBCA] W. T. Polk and N. E. Hastings, *Bridge Certification Authorities: Connecting B2B Public Key Infrastructures*. NIST 2001 <http://csrc.nist.gov/pki/documents/B2B-article.pdf>
- [Identrus] Identrus, web site <http://www.identrus.com/>
- [JFK] W. Aiello, S.M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A.D. Keromytis, O. Reingold *Just Fast Keying (JFK)*, Internet draft <http://www.ietf.org/internet-drafts/draft-ietf-ipsec-jfk-00.txt>
- [Kohn78] Kohnfelder, L. M. (1978). *Towards a Practical Public-Key Cryptosystem*. *Laboratory for Computer Science*. Cambridge, Massachusetts Institute of Technology.
- [LDAP] T. Howes, M. Smith, *The LDAP Application Program Interface*. RFC 1823 August 1995.
- [OCSP] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*. RFC 2560 June 1999.
- [PGP] Atkins, D., Stallings, W. and P. Zimmermann, *PGP Message Exchange Formats*, RFC 1991, August 1996.
- [PKIX] Public Key Infrastructure X.509, Internet Engineering Taskforce.
- [SAML] P. Hallam-Baker and Eve Maler, *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)* <http://www.oasis->

[open.org/committees/security/docs/draft-sstc-core-25.pdf](http://open.org/committees/security/docs/draft-sstc-core-25.pdf)

<http://www.xmltrustcenter.org/research/docs/X-KASS-31.pdf>

[SDSI] Ron Rivest and Butler Lampson, SDSI - A Simple Distributed Security Infrastructure [SDSI], <http://theory.lcs.mit.edu/~cis/sdsi.html>

[XKMS] W. Ford, P. Hallam-Baker, B. Fox, B. Dillaway, B. LaMacchia, J. Epstein, J. Lapp, XML Key Management Specification (XKMS), W3C Note 30 March 2001, <http://www.w3.org/TR/xkms/>

[Sebiok] T. Sebiok, *Signs: An Introduction to Semiotics*. Toronto: University of Toronto Press, 1994

[XTAML] P. Hallam-Baker, *XML Trust Axiom Markup Language 1.0*, VeriSign Inc. September 2001. <http://www.xmltrustcenter.org/>

[SOAP] D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Frystyk Nielsen, S Thatte, D. Winer. *Simple Object Access Protocol (SOAP) 1.1*, W3C Note 08 May 2000, <http://www.w3.org/TR/SOAP/>

[XML-SIG] D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer, B. Fox, E. Simon. *XML-Signature Syntax and Processing*, World Wide Web Consortium. <http://www.w3.org/TR/xmlsig-core/>

[SPKI] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, T. Ylonen. *SPKI Certificate Theory*, RFC 2693, September 1999.

[Borisov01] Nikita Borisov, Ian Goldberg, and David Wagner. *Intercepting mobile communications: The insecurity of 802.11*. In Proceedings of MOBICOM 2001, 2001. <http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf>

[WAP] WAP Certificate profile Specification, <http://www1.wapforum.org/tech/terms.asp?doc=WAP-211-WAPCert-20010522-a.pdf>

[WEP] *LAN MAN Standards of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specification. IEEE Standard 802.11, 1977 Edition, 1977*

[X.500] ITU-T Recommendation X.501: *Information Technology - Open Systems Interconnection - The Directory: Models*, 1993.

[X.509] R. Housley, W. Ford, W. Polk, D. Solo. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. RFC 2459, January 1999.

[XKASS] P. Hallam-Baker, *XML Key Agreement Service Specification (XKASS)*, May 2001, XML Trust Center Research note,