

**Executive Whitepaper**  
**David D. Croston, CEO**

## **Enterprise SOA Trust**

Today's IT environment is driven by the need to connect disparate systems into a flexible, unified, efficiently-performing whole. The connected systems may span continents, enterprise and organizational trust domains, and legal jurisdictions. In order to connect, enterprises are turning to Service-Oriented Architecture (SOA) implemented with SOAP-based Web Services and SAML-based federated identity management. This white paper highlights some of the inherent security features and demands of these new technologies and how the Trust Authority System from IAM Technology helps enterprises achieve real-time identity and data validation in a service-oriented environment with a dynamic membership and regulatory compliance requirements.

### ***SOA, Web Services, and Security***

Over the past decade, enterprises have realized that in order to be competitive (or in the case of governments, be effective) they need to be able to easily and effectively share services both within their constituent organizations and with other enterprises. The modern approach for doing so is Service-Oriented Architecture (SOA) and most implementations of service oriented architecture used SOAP-based Web Services.

Security, of course, remains paramount and in the case of Web Services, there is a suite of *Web Services Security* specifications of which the *SOAP Message Security* specification defines how to use message-level cryptography such as XML Signature and XML Encryption for SOAP messages.

Another security aspect of SOA is authentication – both for signature validation and authorization. In particular, because SOA is geared to sharing services across organizations, a flexible and robust authentication meta-layer is needed and that is why the SAML (Security Assertion Markup Language) specification has been developed.

SAML enables an assertion about an entity's authentication and/or its attributes to be shared among different organizations no matter what authentication mechanisms or identity management infrastructure they use. SAML's inherent support for federated identity has made it the core technology behind industry federated identity initiatives such as the Liberty Alliance Project and Shibboleth. There is also a *Web Services Security* specification, the *SAML Token Profile*, defining how to use SAML-based tokens within SOAP messages. And like the *SOAP Message Security* specification, the SAML core specification also defines how to use XML Signature and XML Encryption to protect messages.

XML Signature and XML Encryption are particularly valuable for securing SOAP and SAML messages because they enable parts of the message to be signed and/or encrypted in accordance with the specific

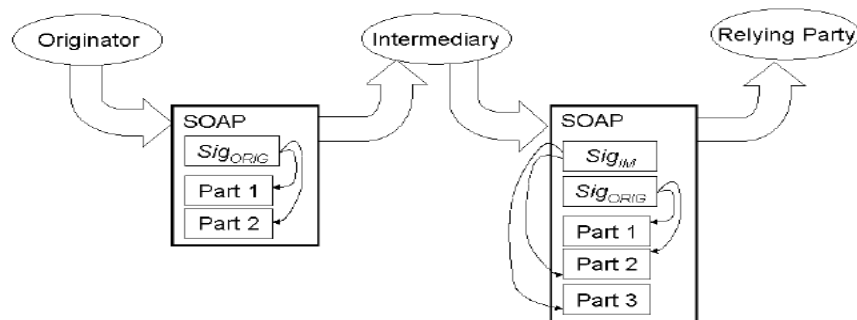
security needs of that part, the signer, and the target audience. For example, if part of a SOAP message is intended to be modified by intermediaries between the sender and the recipient, that part must be left unsigned or else the signature of the originator would become invalid. Similarly, if parts of a SAML message need to be left in the clear or need to be encrypted for different parties, that can be elegantly done with XML Encryption. The *SOAP Message Security* specification defines how to include and process XML Signature and XML Encryption objects within SOAP messages. Similarly, the SAML specification does the same for SAML assertions. Finally, both the SAML and *SOAP Message Security* specifications support signed timestamps to help ensure the freshness of the message.

### Multi-Party Processing

In a service-oriented environment, the execution of a service may result in a number of SOAP messages being exchanged among a number of different entities. It is important to note that SOAP is designed for more complicated transactions than simple request/response. The design of SOAP enables the same SOAP message to be received, modified, and forwarded to a different party than from the one it was received. In SOAP terminology, intermediaries perform different roles with respect to the same SOAP message. Some intermediaries may add information; some may delete information; and some may only read existing information all in the due process of fulfilling a service.

The *SOAP Message Security* specification describes how one or more signatures can be applied and validated to ensure the full security of the SOAP message as it travels from the initiator, through the intermediaries, and the ultimate relying party. Figure 1 illustrates a SOAP message as it is transformed from point to point. In the figure, the originator creates a SOAP message and applies a signature ( $Sig_{ORIG}$ ) that signs Parts 1 and 2. An intermediary captures the message appends a new part (Part 3), adding its own signature ( $Sig_{IM}$ ) to cover parts 2 and 3 (by counter-signing Part 2, it may be indicating that it has reviewed and approved it). The message then goes to the relying party who verifies both signatures.

Figure 1: Multi-Party Processing of a SOAP message



Besides signing parts of a SOAP message dealing with the execution of the service, the *SOAP Message Security* specification also defines the signing of security tokens and references to security tokens within SOAP messages. With regard to the latter, the *SOAP Message Security* specification defines a special transform that allows an XML Signature to point to a security token reference but actually sign the referenced object, not the reference. Hence, service-oriented environments have the flexibility of

signing the security token reference itself, the object pointed to by the security token reference, or doing both.

## **Compliance**

Both SOAP and SAML have features supporting compliance to rules regarding processing and policies. An (optional) feature of *SOAP Message Security* of particular value to service-oriented environments where compliance is a priority is the ability of a requester to require that a responder provide evidence in the response that the responder validated the signature in the request. This is so that the requestor can be confident that the response is based on unaltered information sent by the requestor. The feature, called *SignatureConfirmation*, does not disallow intermediaries from modifying non-protected parts of the SOAP message as long as those intermediaries do not maliciously or accidentally modify parts of the SOAP message signed by the requestor.

In SAML, XML Signatures are used to authenticate and protect the integrity of both requests (e.g. a request for an authentication assertion) and responses (an authentication assertion). The SAML specification states that if a SAML assertion contains an XML Signature, then the relying party must not trust the SAML assertion unless the XML Signature is valid. If the signature is valid, the relying party may then assess “the identity and appropriateness of the issuer and may continue to process the assertion in accordance with this specification and as it deems appropriate”.

SAML also uses signatures for signing the Consent attribute. The Consent attribute indicates whether or not the consent of the principal (the one whom the assertion is about) was obtained or not by the issuer of the message (either a request or response) and under what conditions. As examples of consent, consider a service that would like to obtain an assertion from an identity provider about an attribute of a user. The service may need, due to internal privacy policies or government regulations, to obtain the consent of the user before querying certain information about that user. As well, the user's identity provider may need to obtain the consent of the user before providing that information.

## **Need for Message-Level Security**

Other message-level security formats such as PKCS#7 which are not XML-aware do not inherently have the capability of selectively signing and encrypting message parts for different audiences. PKCS#7 does, however, at least provide unbroken message-level security between the originator and the relying party. In contrast to the message-level security provided by XML Signature, XML Encryption, and PKCS#7, is TLS which operates at the transport layer. For direct server to server communication that does not require application-aware, process-aware, policy-aware, auditable, provable security, TLS is sufficient. In the world of SOA, however, one needs end-to-end security that can be tailored to the specific, dynamic needs of enterprises and their applications; and that is historically auditable in order to prove compliance with the internal and external regulations. That kind of higher-level security requires message-level security.

## **Need for Speed**

In a service-oriented environment, message-level security requires access to the public key material of the constituent organizations. This means that there must be a secure, up-to-date repository of the public keys and a highly efficient means of accessing the public keys in that repository. However the nature of service-oriented environments is that they are distributed (often over a wide geographic area), may contain mobile clients, and yet, because the services are intended to enable machine-to-machine communication, high performance is a necessity. And that means that message-level security too must be of the highest performance or else it will be a bottleneck to the system's primary functionality.

### ***The IAM Trust Authority System***

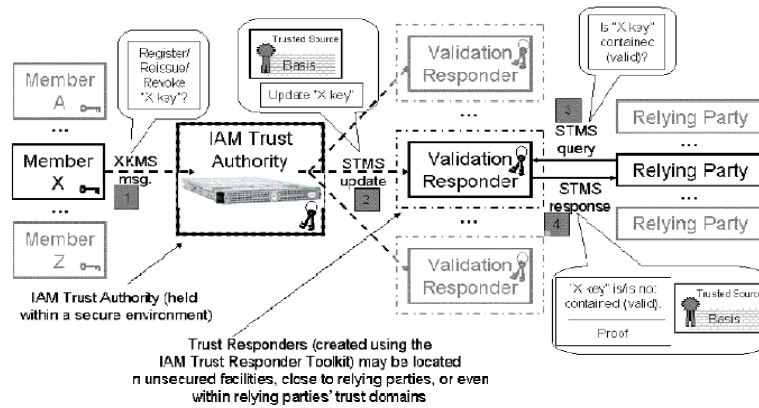
The Trust Authority System from IAM Technology is a hardware appliance that incorporates breakthroughs in cryptographic research made at Brown University for the United States Department of Defense. Specifically, the Trust Authority System enables fast and easy distribution of validated public keys in distributed, federated environments such as those built on SOA using SAML and SOAP-based Web Services. It does so by allowing relying parties to efficiently interact with validation responders. A *validation responder* is an authenticated dictionary widely distributed throughout Layer 7, and among partner organizations. Validation responders reply to queries about the validity of a public key in real time (tested to 54 micro-seconds) locally. An answer from a validation responder can be trusted as if it came from a source. Validation responders can even issue trust in untrusted environments. As illustrated in Figure 2, in the IAM Trust Authority architecture, validation responders lie between the IAM Trust Authority hardware appliance and the relying parties.

These features distinguish the IAMTrust Authority architecture:

- Trust domains can easily update their public key material through XKMS (XML Key Management Specification) messages to the IAM Trust Authority;
- The IAM Trust Authority can efficiently and securely push updates to validation responders throughout a service-oriented environment; and
- The validation responders can provide near-instantaneous, verifiable responses to public key requests from applications

The last two points are made possible through the Secure Transaction Management System (STMS) breakthrough technology developed by Brown University and IAM Technology. Figure 2 illustrates how the IAM Trust Authority enables real-time public key validation in an environment with multiple members and distributed relying parties (parties relying on the validity of the public key).

Figure 2: The IAM Trust Authority enabling effective and efficient key validation



In Figure 2, members’ keys are added to (registered), changed within (reissued), and deleted from (revoked) the IAM Trust Authority through XKMS messages [1] sent by the key’s owner/issuer to the IAM Trust Authority. Whenever a key’s status changes, the IAM Trust Authority recalculates (using an efficient cryptographic technique) a hash of the set of valid keys and signs that hash. The signed hash of the set is called a “Basis”; the Basis is sent to the validation responders along with an instruction reflecting how the validation responders are to update [2] their copies of the set of valid public keys – these “sets”, in cryptographic terminology, are referred to as “authenticated dictionaries”. In this way, the validation responders keep their authenticated dictionary of valid public keys in synchrony with that of the IAM Trust Authority.

When a relying party receives a SOAP or SAML message and needs to validate the digital signatures within, it queries the nearest validation responder as to whether the public key in question is contained within the validation responder’s authenticated dictionary [3]. The validation responder replies with an answer (“contained” or “not contained”), a cryptographic proof of the answer, and the Basis [4]. Using the efficient cryptographic technique, the client application verifies the answer against the proof and the Basis. Because the Basis is signed by the IAM Trust Authority, the trust in the answer lies with the IAM Trust Authority, not with the validation responder (which need not be secured).

### Dynamic Membership

In a dynamic, federated environment, new members enter and existing members may leave. As members come and go, so must the status of their public keys within the environment.

The X10 Trust Authority architecture makes management of a dynamic membership easy through its straightforward XKMS interface for registering, reissuing, and revoking the public keys securely stored within the X10 Trust Authority. What is unique about the Trust Authority is how it then updates the information base of the validation responders. Through the STMS update mechanism, the IAM Trust Authority can update multiple validation responders using very little bandwidth and requiring minimal processing power internally and within the validation responders. This makes it possible for the

organizations within a service-oriented environment to be assured that the relying parties in that environment have the best possible access to up-to-date trust information.

### **Historically Provable Validation of the Public Key**

Compliance requires evidence. In the area of signature validation, one needs evidence that the public key used to validate signature was valid at the time that the validation occurred. As mentioned in the previous section, participating organizations within the service-oriented environment join, leave, or simply update their keys. Yet how can one know whether a particular public key was valid at the time signature was verified?

To do this, the IAM Trust Authority incorporates each update into the existing authentication dictionary in accordance with IAM's Persistent Authenticated Dictionary (PAD) technology. With PAD, each update to the authenticated dictionary results in an adjunct to the data structure along with a corresponding Basis. To accommodate the time dimension of PAD, the STMS queries for historic validation include an additional parameter for the time instant of interest and the returned answer proofs are similarly cryptographically dependent on the time instants of the corresponding updates to the status of the public key. Consequently, the validation status of a public key can be proved for any time instant and, as such, that proof can be used as evidence for compliance with respect to signature processing and therefore also with respect to identity-related assertions and Web Services transactions.

### **Real-time Public Key Validation**

As mentioned earlier, dynamic, federated, service-oriented environments cannot be bottlenecked by security. Indeed, many of them are incorporating in-line hardware appliances to realize wire-speed processing of secured SOAP and SAML messages.

Whether signature validation is done in hardware or software, doing it efficiently requires real-time responses to public key validation queries. The IAM Trust Authority architecture makes that possible. The high efficiency of an IAM Trust Authority-based deployment is possible because the placing of the validation responders can be optimized for the particular resident infrastructure as the validation responders need not be secured. This is possible thanks to the use of a unique data structure (called a skip list) that enables efficient updating of distributed validation responders and a corresponding cryptographic mechanism that provides trusted proofs to relying parties about the answers received from a validation responder. With the IAM Trust Authority Architecture, system components within a service-oriented environment can now have up-to-date validation status for public keys, and access to those keys, in real time.

### **Conclusion**

Collaboration among enterprises, or within the constituent organizations of a single large enterprise, is being realized today through service-oriented architecture – much of which is being implemented with SOAP-based Web Services to enable advanced communication among devices and with SAML to ensure that participants in the transaction are identified, authenticated, and authorized.

SOAP and SAML, and their respective security specifications, contain many features that are invaluable to implementing multi-party business processes and ensuring that those processes are conducted in compliance with security policies, privacy policies, internal policies, and government policies. These features are designed to be best secured with message-level security; however, message-level security in a service-oriented environment requires real-time public key validation. Now, with the IAM Trust Authority, real-time public key validation is a reality thanks to its implementation of IAM's advanced cryptographic research. Furthermore, the revolutionary technology behind the IAM Trust Authority architecture supports dynamic membership among the participating enterprises and provides cryptographically provable evidence about the validity or not of a specific public key for future compliance audits.

For more information on IAM Technology, please contact us:

IAM Technology, Inc.  
1666 Massachusetts Avenue  
Lexington, Massachusetts USA  
866-585-6280  
[info@iamtech.com](mailto:info@iamtech.com)  
[www.IAMTECH.COM](http://www.IAMTECH.COM)