FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS

Document title	FIPA Agent Message Security Object Proposal		
Document number	`f-in-00095`	**Document source**	Giosue Vitaglione (TILAB) Nicolas Lhuillier (Motorola) Dominic Greenwood (Whitestein)
Document status	Draft	**Date of this status**	2003/11/13
Change history	2003/11/13 Initial Draft		

http://www.fipa.org/
Geneva, Switzerland

Foreword

Contents

1. Definitions....3
  1. Algorithm 3
  2. Authentication 3
  3. Encryption 3
  4. Hash 3
  5. Integrity 3
  6. Message Authentication Code (MAC).... 3
  7. Message payload 3
  8. Public key cryptography 3
  9. Signature 3
2. Per message security 4
  1. FIPA SecurityObject 4
  2. Format of SecurityObject 4
3. Signature Scenarios 5
  1. RSA-like Signature 5
  2. DSA 6
  3. MAC 6
  4. Kerberos signature 6
4. Encryption scenario 7
  1. Asymmetric algorithm 7
  2. Symmetric algorithm with secret key wrapped 7
  3. Symmetric algorithm with known secret-key 7
5. Acknowledgement 8
6. References 9

Introduction

This note includes a proposal for a ***SecurityObject*** to be used in the envelope of FIPA ACL messages for providing interoperable per message security. First, the structure of this object is described, then some scenarios are described to provide data integrity and data origin authentication by using commonly used technologies. Finally, it is also described how the SecurityObject can be effectively used as placeholder for the information to ensure message-level encryption.

1.Definitions

This section defines some of the most important terms used in this document.

1.Algorithm

This represents any standard cryptographic algorithm (such as AES, RSA, DSA, ECDSA, SHA-1, etc.) usually used to cipher, sign or hash data. Encryption algorithms can be symmetric (e.g. AES) or asymmetric (e.g. RSA).

2.Authentication

Also called “data origin authentication” this includes any mechanism to ensure that data received actually originates from the claimed sending entity. This is a usual safeguard against masquerading, spoofing, etc.

3.Encryption

This includes mechanisms used to protect the confidentiality of transmitted data by preventing anyone but the intended receiver to access it. This is a usual safeguard against eavesdropping.

4.Hash

A hash algorithm is a one-way function that produces from the original data, a data segment of specific length in such a way that there is a high probability that any change to the original data will result in a change to the digest.

5.Integrity

This includes all mechanisms to ensure that the data received by the recipient is exactly the one sent by the sender. This is a usual safeguard against tampering.

6.Message Authentication Code (MAC)

This is a hash produced using a secret key (usually appended to the original data) to provide authentication in addition to integrity.

7.Message payload

The ACL message, encoded according to the “payload-encoding” slot of the envelope and which is transported by FIPA MTS.

8.Public key cryptography

A security model in which all entities own a key-pair, composed of a public-key known by everyone and a private-key, which is secretly kept. Data encrypted with the private-key can only be decrypted with the corresponding public-key and vice-versa. Key-pairs are used with asymmetric cryptographic algorithm (e.g. RSA) and are often linked to Public-key infrastructures (PKI).

9.Signature

Also called “digital signature”, this represents a possible mechanism to ensure both authentication and integrity of transmitted data. The signature mechanism consists in the sender creating a hash of the data to be sent and encrypting this hash using an asymmetric algorithm.

2.Per message security

This proposal introduces the concept of “per message security” which means that each individual ACL message contains the security information required to process the embedded security safeguards. This proposal is intended to be generic and extensible enough to support a plurality of different security safeguards (sections 3 and 4 give some example scenarios). Agents are intended to process the security mechanisms themselves where appropriate so as to provided end-to-end (or peer-to-peer) security. However this does not exclude the provision of platform services that provide additional security mechanisms such as authentication services and secured MTP (not yet defined by FIPA).

1.FIPA SecurityObject

The security information required to process security safeguards needs to be transmitted within the FIPA TransportMessage. The *SecurityObject* this proposal introduces is the generic placeholder for such information, just like the *ReceivedObject* already represents stamps placed by the MTS. For instance, if the message is signed, the SecurityObject will contain the signature of the message.

In order to ensure integrity or confidentiality of an entire ACL Message, most safeguards need to apply only to the message payload, therefore this proposal attaches the SecurityObject to the message Envelope. The SecurityObject can be included as user-defined slot into the envelope (e.g. “X-Security”), or, if standardized by FIPA, as an optional slot (e.g. “Security”)1. Furthermore, the slot containing the SecurityObject can contain a set of SecurityObjects, according to the different safeguards applied to a message2, just as the envelope can already contain several ReceivedObjects.

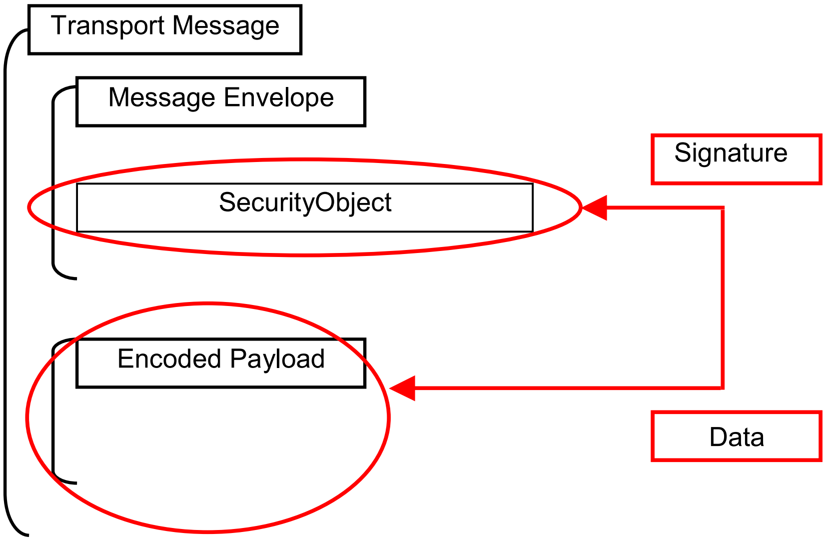

Figure 1: Example of SecurityObject used to contain the signature of a message

2.Format of SecurityObject

This section presents a proposed format for the SecurityObject to be discussed, refined and standardized by FIPA. For instance, the SecurityObject must include all the information required by the message receiver to perform message authentication and decrypt the payload. Sections 3 and 4 give more some technology specific scenarios of how this is achieved.

---

1 Note this will create backwards incompatibility with the IDL definition of FIPA envelope unless SecurityObject is a "X-" user-defined slot."

2 Note that this would require FIPA to define a precedence mechanism for processing of safeguards.

Frame Ontology	`security-object`

Parameter	**Description**	**Presence**	**Type**	**Reserved values**

type	Indicates the specific usage of the generic SecurityObject	Mandatory	`String`	`fipa-security-signature`
				`fipa-security-encryption`
				`fipa-security-kerberos`
				etc

algorithm	The algorithm used to process data. This shall be explicit enough, e.g. including the mode, CBC, CFB for block ciphers.	Optional	`String`	e.g. `RSAwithSHA1`
key	Key data (e.g. public-key) encoded with Base 64	Optional	`String`	
certificate	Certificate data (DER Base 64 encoded)			
key-ref	Reference of the key if key is not included for efficiency purpose	Optional	`String`	
data	Generic placeholder for cryptography-related data (e.g. signature, MAC, ticket, wrapped symmetric secret key). The actual content will depend on the "type" slot. Base 64 encoded.	Optional	Set of `string`	
parameters	Specific parameters that may be required by the safeguard (e.g. IV). Base 64 encoded	Optional	Sequence of `string`	

Note 1: FIPA will have to define the encoding of the SecurityObject in all standard envelope encoding format: `fipa.mts.env.rep.xml.std`, `fipa.mts.env.rep.bitefficient.std`, `fipa.mts.env.rep.idl.std`.

Note 2: the SecurityObject could also include customisable user-defined slots to allow usage of safeguards that may not supported by this specification. However, this may add unnecessary complexity, as custom (non-standard) security can also be placed in other user-defined envelope slots.

3.Signature Scenarios

Signatures can be used in order to provide data integrity and data origin authentication. When communication relies on message exchange, each message can be signed by the sender in such a way that the receiver can verify the validity of the signature. The verification results allow the receiver of the message to securely evaluate the information integrity and the identity of the sender. The payload shall not been modified between signature calculation and verification, therefore signatures can be applied to a FIPA ACL message by calculating the signature over the encoded payload. This allows protecting the information included into all ACL slots.

In this section we provide some examples of the kind of information included into the SecurityObject by taking into consideration some common scenarios.

1.RSA-like Signature

The sender owns a cryptographic public/private key pair. He calculates the *signature = f(payload). f* is a non invertible function, calculated by the following steps:

1. A hash function (example: MD5, SHA-1) is calculated over the payload;
2. The result of the hash is asymmetrically encrypted (example: RSA) by using the sender's private key.

The *signature* consists in the result of this encryption, plus all the needed information required by the receiver in order to verify the message integrity and authenticity.

The verification process consists in asymmetrically decrypt (using the sender’s public key) the message signature to have the hash of the payload at the source. Then the hash is calculated over the received payload. The two hashes are compared, if they are equals the integrity of payload is ensured as well as the origin of the message.

In this case, the SecurityObject shall for instance include the following information:

```
Type = ”fipa-security-signature”
Algorithm = ”RSAwithSHA1”
Data = ” xA7SEU+e0yQH5rm9kb...”     // the calculated signature
Key = “80EF45632...”               // encoded public key
```

From the encoded public key, if needed, the receiver can easily calculate modulus and public exponent. The verification is performed by using the algorithm indicated, over the signature value and the sender’s public key.

2.DSA

With DSA, the SecurityObject shall contain the following information:

```
Type=”fipa-security-signature”
Algorithm   = ”DSA”
Data = ” xA7SEU+e0yQH5rm9kb...”     // the calculated signature
Parameters = “Key_P” “Key_Q” “Key_G” “Key_Y” “Key_J”     // encoded keys (optional)
```

3.MAC

The sender calculates the signature as hash of the payload concatenated with other data. Such data is a concatenation of:

1. a string derived by a secret key (shared with the receiver); and optionally
2. an arbitrary string that can be chosen (example: randomly) by the sender.

In this case, the SecurityObject shall include the following information:

```
Type=”fipa-security-mac”
Algorithm   = ”MD5”       // the hash algorithm
Data = ”xA7SEU+e0ywJrm9kb...”      // the calculated signature
Parameters = “fr5jvddvr6...”      // random text (optional)
Key-ref = “...”         // reference to the secret symmetric key (optional)
```

In order to perform the signature verification, the receiver will recalculate again the hash of the concatenation of: payload+data, and compare it with the received signature.

4.Kerberos signature

The Sender first requests a TGT (Ticket-Granting Ticket) from a AS (Authentication Service), which will be subsequently used by the Sender to request individual session tickets from the TGS (Ticket Granting Service). Normally, the AS and TGS whilst logically distinct may be physically co-located and collectively termed as a KDC (Key Distribution Centre). The TGT's validity can time-period limited by the TGS and contains a SessionKey signed using the TGS's own private key.

When this Sender now creates a message to be sent to Receiver, it contacts the TGS using the established TGT. The TGS returns two new SessionKeys; the first (known as the Validator) signed using the SessionKey contained in the TGT and the second (known as the Ticket) signed with the Receiver key (which must be pre-registered with the TGS). The Sender then unlocks the Validator using the TGT SessionKey and then uses the TGS SessionKey contained within to sign an Authenticator token, which will typically be a timestamp or checksum. The Ticket (returned from the TGS) and Authenticator are then attached to the outgoing message as a signature and the message is sent to Receiver.

The Receiver extracts the TGS SessionKey from the Ticket using its own key and then uses this to extract the token from the Authenticator. If these steps are successful then the message is deemed to be authentic (i.e. sent by Sender).

With Kerberos, the SecurityObject shall contain the following information:

```
Type = "fipa-security-kerberos"
Algorithm = "Kerberos"
Data = (validator, ticket, authenticator)  // validator and authenticator not simultaneuously present
```

4.Encryption scenario

In the encryption scenario, the SecurityObject shall contain the information required by the receiver to decrypt the payload. There are different sub-scenarios, depending if the payload has been encrypted with a symmetric or asymmetric algorithm, if the symmetric secret-key is known by the receiver, etc.

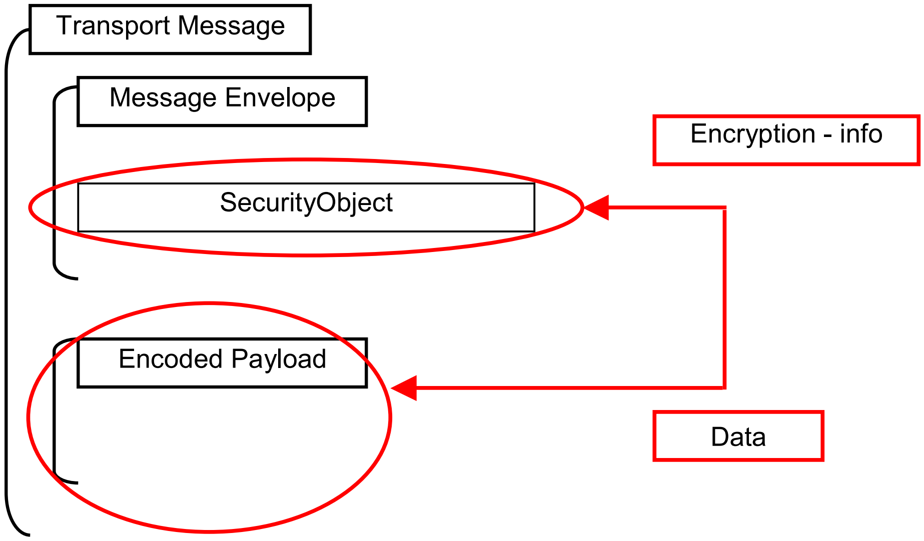

Figure 2: Example of SecurityObject used to contain the encryption information

1.Asymmetric algorithm

```
Type = "fipa-security-encryption"
Algorithm   = "RSA"
KeyRef = "..."     // (or key)
```

2.Symmetric algorithm with secret key wrapped

```
Type = "fipa-security-encryption"
Algorithm   = "AES"
Data = "xA7SEU+e0ywJrm9kb..."     // the wrapped symmetric key
Parameters = "Algo" "PubKey"     // Algorithm and public-key used to wrap the secret key
```

3.Symmetric algorithm with known secret-key

Here the symmetric key has been somehow agreed in advanced by the agents:

```
Type = ”fipa-security-encryption”
Algorithm   = ”AES”
Key-ref = “..”    // reference to the symmetric key
```

5.Acknowledgement

This proposal and the SecurityObject concept has been developed by the Security group working in the JADE Board, mainly by Giosue Vitaglione (TILAB), Nicolas Lhuillier (Motorola) and Dominic Greenwood (Whitestein).

The authors grateful acknowledge the help and support of people who contributed into the discussions or reviewed this document, in particular: Sergi Robles and Joan Ametller (MARISMA research team), Fabio Bellifemine and Giovanni Caire (TILAB).

6.References

[FIPA00075] FIPA Agent Message Transport Protocol for IIOP Specification, 2002. http://www.fipa.org/specs/fipa00075/

[FIPA00085] FIPA Agent Message Transport Envelope Representation in XML Specification, 2002. http://www.fipa.org/specs/fipa00085/

[FIPA00088] FIPA Agent Message Transport Envelope Representation in Bit Efficient Specification, 2002. http://www.fipa.org/specs/fipa00088/

[FIPA00067] Transport Service Specification. Foundation for Intelligent Physical Agents, 2000. http://www.fipa.org/specs/fipa00067/

[FIPA00067] Transport Service Specification. Foundation for Intelligent Physical Agents, 2000. http://www.fipa.org/specs/fipa00067/

[XMLSignWG] XML Signature Working Group - http://www.w3.org/Signature/

[Kerberos] Kerberos: The Definitive Guide by Jason Garman. O'Reilly & Associates 2003. ISBN: 0596004036