Secure Communications in Multi-Agent Systems Protecting KQML

Sierra J. M., Hernández J.C., Izquierdo A. and Ribagorda A.

IT security Group. University Carlos III of Madrid. Spain

Abstract. When multiagent systems use insecure networks their communications must be protected in the same way that any other applications that run over this type of channels. There is no doubt that multiagent systems expansion will be joined to the Internet technology, and for that reason our work tries to protect agents communications by a new security architecture and an extension of the KQML. Our security architecture has been designed to be installed over the RETSINA framework, which was specifically designed for an open system, such is the Internet. The core of our proposal is a SEcurity SubAgent Module, called SESAMO, which was expressly designed to easily interact with the RETSINA components. The protection is based a public key infrastructure that, in addition to an extension of KQML, will supply authentication, non-repudiation, integrity and confidentiality services to agent communications.

1 Introduction

KQML, *Knowledge Query and Manipulation Language*, permits autonomous and asynchronous agents share their knowledge and work cooperatively for solving problems. The possibilities of Multi-Agent Systems (MAS) increase considerably if they use the Internet. But it is necessary to adapt the KQML to this open environment, supplying to the agents security services such are confidentiality, integrity, authentication and non-repudiation.

First aim of this security architecture is to effortlessly coexist with other multiagent systems. Our proposal is designed to work over the RETSINA framework. The core of our architecture is the SESAMO module (SEcure SubAgent MOdule). This module supplies cryptographic capabilities to RETSINA Task Agents, permitting them to establish secure communications with others. The SESAMO module can be installed into a Host Agent or also allows that several agents (agents connected by a private network or installed into the same machine) to share a single SESAMO, we called that option Shared SESAMO. We also describe some other functions that can be developed by SESAMO because its design can be used as a communications security gateway between groups of agents.

Agents that want to interact directly with their parties can bypass our architecture. A common situation could be that Task Agents just use the SESAMO when the remote agent is asking for a secure connection, or when they want to establish this type of connections with others. In the rest of situations they will

J. M. S., J.C. H., A. I. and A. R. (2004).
Secure Communications in Multi-Agent Systems Protecting KQML.
In *Proceedings of the 2nd International Workshop on Security in Information Systems*, pages 305-313
DOI: 10.5220/0002684803050313

communicate openly with KQML. The SESAMO modules will communicate using a security extension of KQML that we have designed. This extension is called KQML-SE and is composed by three new performatives:

1. Cryptographic Capabilities Negotiation.
2. Both Parties Authentication messages.
3. Encapsulated KQML messages.

Our scheme is based on public key cryptography, and obviously this environment needs the existence of a Public Key Infrastructure. This PKI will be used for the authentication of agents and hence the architecture security relies on it. We have designed another extension of KQML that provides the performatives needed for the creation, renewal and cancellation of certificates, and also for the maintenance of Certificate Revocation Lists (usually noted as CRL's). All these topics will not be treated in this paper because we would need some more extra space to describe its behaviour and management. However, we are aware about the essential role that PKI plays in our architecture.

2 Security Architecture

When we start this work we tried to develop architecture easy to place in a Multi-Agent System. This architecture should introduce the minimum number of changes, enabling an agent to integrate security services with no modifications on its basic architecture. In this paper we concentrate our work in the well-know architecture called RETSINA. In our proposal each agent (from now on, Host Agent) has another sub-agent associated, called SESAMO (SEcurity Sub-Agent MOdule). SESAMO module is in charge of dealing with all the KQML messages sent or received by the Host Agent.

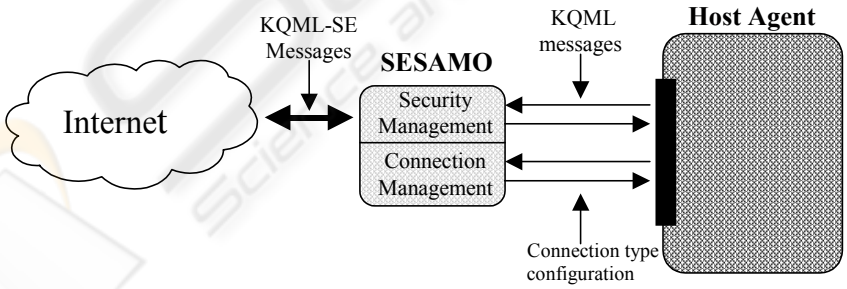

Fig. 1. : KQML-SE architecture based on the SESAMO

The SESAMO only manages performatives defined by our new ontology PKCertificate. SESAMO has implemented all the cryptographic capabilities that the Host Agent does not have; in this way the SESAMO will provide all the required

security services (authentication, non-repudiation, confidentiality and integrity). On the other hand the agent will be working with no change, security services will be applied by the SESAMO transparently to the Host Agent. The Host Agent only must indicate to the SESAMO which messages will need protection.

The SESAMO will be divided into two parts, *Security Management* part, which will be in charge of protecting the KQML messages exchanged between the Host Agent and the other agent. And the second part, *Connection Management*, which will be use by the Host Agent to specify to the SESAMO the connection features. The Host Agent must indicate to the SESAMO what to do in case that other remote agent does not have his own SESAMO. For example a Host Agent could block any message that is not authenticated, in this case the SESAMO will be protecting his agent from non-authenticated communications just keeping away from messages that does not contains the PKCertificate ontology. The SESAMO could filter communications, for example rejecting messages with certain source address, size, etc..

Another functionality that can be included into the SESAMO is a Connection Features Database. Into this database will be stored the connection features of a pair of agents (the Host Agent and the remote Agent). Using this database will be skipped the main work of the Connection Management part of the SESAMO. This database will be particularly useful when a Shared SESAMO is used (see Sharing a SESAMO Agent section)

An important advantage of our design is that the communication is possible in any case. If one of the parties does not have a SESAMO module, the other SESAMO can bypass the common KQML messages (without the PKCertificate ontology) and permits to establish the communication.

Table 1. : Optional functionalities of SESAMO module

Function	Type
Encapsulating (KQML ↔ KQML PKCertificate).	Obligatory
Strong Authentication of both agents	Obligatory
Digital signature of KQML messages	Obligatory
Shared SESAMO	Optional
KQML messages filtering	Optional
Connection features Database	Optional

3 KQML Security Extension. KQML-SE

The KQML security extension consists on four new performatives. These performatives will be used in three steps. The first one is the negotiation of the Cryptographic capabilities. The new performative designed is ***negotiation,*** which enables to negotiate: Certification Authorities, Digital Signature Algorithm, Cipher Algorithm and Digest Algorithm.

Once agents know the cryptographic capabilities one each other, it begins the second step where is accomplished the authentication of the parties and the establishment of a shared secret key (this key will be based on information supplied

by both parties). The performatives that support this second step are Auth-link and Auth-challenge.

Finally, once the parties are properly authenticated and a ciphering key is established, it is possible to set up a secure channel. The performative is called ***Auth-private***.

Next subsection describes the new parameters involved.

3.1 New parameters

3.1.1 :certificateCA(<Certification Authority 1><Certification Authority 2>,...)

Where the argument <Certification Authority X> is the identification associated to one Certification authority (Thawte, VeriSign). This parameter is an enumeration of the different certification authorities supported by the agent. It is a parameter of the Negotiation performative.

3.1.2 :certificate (<Certification Authority><the certificate>)

The second argument is the certificate, which is a public key signed by a certification authority (indicated in the first argument).

3.1.3 :connection-id (<NONCE_X><NONCE_Y >)

This parameter is used in the Auth-private performative and identifies a previous negotiation.

3.1.4 :auth-key(<boole><key-type><SEED>)

First argument is a Boolean value. If this value is TRUE the session key must be changed. The new key to use will be the result of a hash function calculated over the concatenation of SEED, NONCE_X and NONCE_Y (the resulting digest could need to be adapted to the encryption algorithm key size. This operation is done to ensure that the new key depends from both parties.

If the first argument is FALSE, next arguments will be ignored because the agent is signifying that, for the moment, the session key is valid. Any party of the communication could indicate a key change when it considers appropriate. This parameter is included into Auth-private and Auth-challenge performatives.

3.1.5 :signature(<<Key_ID><information signed>)

The second argument represents certain information digitally signed by the agent. The first argument identifies the public key that must be used to check the signature. It is included into Auth-link, Auth-challenge and Auth-private performatives and provides Authentication and Non-Repudiation of the messages.

3.1.6 :algSecretKey (<alg1><alg2>...<algN>)

It indicates the different symmetric cipher algorithms supported by the agent that sends this parameter into a Negotiation performative. Also it is used in Auth-link performative, in this case an agent is indicating to its party the selection of certain algorithm.

3.1.7 :algSignType (<alg1><alg2>...<algN>)

This parameter indicates the different digital signature algorithms supported by the agent that sends the message. Can be used into a Negotiation performative and in Auth-link performative but in this case it is signifying the selected algorithm

3.1.8 :algDigestType (<alg1><alg2>...<algN>)

This parameter indicates the different digest algorithms supported by the agent that sends the message. Can be used into a Negotiation performative and also it can appears in a Auth-link performative but in this case just with the algorithm selected

3.1.9 :mySESAMO (<SESAMO_ID><protocol><address>)

It is used into the Negotiation performative and indicates what SESAMO agent is being used for protecting messages.

3.2 New Performatives

Into this section we will present the new performatives defined into KQML-SE. Those performatives should be added to those included into RETSINA.

Table 2. New performatives for KQML-SE

Performatives	Meaning
Auth-link	Request of secure communication
Auth-challenge	Acknowledge for Auth-link request
Negotiation	Cryptographic capabilities negotiation
Auth-private	KQML messages ciphered and encapsulated

In the following lines we describe the content of all these new performatives:

Name: Negotiation.
Description: Cryptographic capabilities negotiation between two SESAMO's
Additional paramenters:
:mySESAMO (Only has to be used when a Shared SESAMO is used)
Ontology: PKCertificate
KQML Description:
Negotiation:
: sender <A>
: receiver <B>

: certificateCA<VeriSign><Thawte><Internal Domain >.....>
: mySESAMO <<X<tcpip><X@domain.com>>
: algDigestType<<MD5><MD4><SHA>...>
: algSecretKey<<DES><RC2>...>
: algSignType<<RSA><DSA>>
: ontology <PKCertificate>

Name: Auth-link
Description: Solicitud de comunicación segura.
Optional parameters:
:peer-address, only used when a shared SESAMO is used.
:algDigestType, list of digest algorithms supported.
:algSecretKey, list of symmetric algorithms supported.
Ontology: PKCertificate
KQML Description:
Auth-link
:sender <A>
:receiver<B>
:reply-with<expresion>
:algDigestTypeType<MD5>
:algSecretKey<DES>
:algSignType<RSA>
:certificate <<VeriSign><Certificate_A>>
:signature<<A_KEY> <Signature of (NONCE_A & A & B)>>
:content: <NONCE_A>

Name: Auth-challenge
Description: Acknowledge for Auth-link request
Ontology: PKCertificate
KQML Description:
Auth-challenge:
:sender <B>
:receiver<A>
:in-reply-to<expresion1>
:reply-with<expresion2>
:certificate <<VeriSign><CertificateB>>>
:auth-key<<T><DES><SEED ciphered by the public key of A>>
:signature<<KEY_B><Signature of (NONCE_A & NONCE_B & SESSION_KEY)>>
:content: <NONCE_B>

Name: Auth-private
Description: KQML message ciphered and encapsulated.
Ontology: PKCertificate
KQML description:

Auth-private:
:sender <A>
:receiver<B>
:in-reply-to<expresion1>
:reply-with<expresion2>
:connection-id<<NONCE_A><NONCE_B>>
:auth-key< <FALSE> <><> >
:signature<<A_KEY><Signature of(KQML message)>>
:content <KQML message ciphered with SESSION_KEY1>

4 Sharing a SESAMO Agent

The SESAMO module can be shared among more than one Host Agent. In this way all the KQML-SE messages must be sent to the SESAMO and the rest of non-protected KQML messages can be sent by each agent itself. The advantages of this approach are very interesting, because using a shared SESAMO the agents can be working with KQML in the same way that they are already working. And, in case they need a protected communication with other agent, they will use the SESAMO for establishing a secure channel.

The SESAMO needs to implement extra software for the management of several connections at the same time. This software will be able of storing the connection features of every agent into the commented Connection Features Database. The shared SESAMO is a distributed scheme. In this way any agent with authorization can use a shared SESAMO. However this system has been designed for being used in a community of agents connected by a private network (or agents that are running into the same machine). The utilization of Shared SESAMO through insecure networks needs additional protections that probably imply an excessive cost and more complicated management.

[1] SESSION_KEY is the digest of NONCE_A, NONCE_B AND SEED. The digest funcition used is the agreed in the negotiation process.

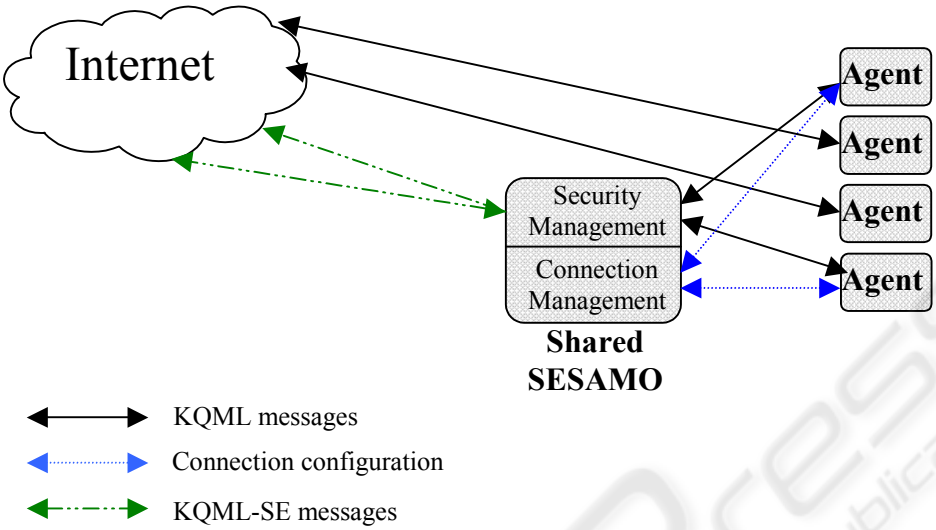

Fig. 2. KQML-SE architecture based on a Shared SESAMO

5 Conclusions

The expansion of the Internet has important implications for the MultiAgent Systems. However the Internet features must be taken into account because all of them will be inherit by the applications that run over it. With this document we tried to outline the security architecture for protecting KQML communications. We are aware that the implementation of our proposal is not completely described here, but this topic is already open in our research group and we will report new contributions.

KQML-SE and SESAMO are part of a complete system with the objective of providing secure communications. Foundations of this system are the public key cryptography and the implementation of a Public Key Infrastructure. We have developed an architecture for the management of PKI in a multiagent system, we have called this module SPA –Security Proxy Agent-. Our work is based on the contributions of Sycara about Security Agents.

Further research on this topic can be associated to the utilization of IPsec framework for the communication of the agents (the new IP Security Protocol that supplies authentication, integrity and confidentiality of IP packets). In this way the SESAMO module can be substituted by an IPv6 implementation on the agents. However, if we use IP security, it does not replace the task of the mentioned SPA module. The combination of the SPA and IPsec protocol will be a next step for our future research works.

6 References

[1] W. Timothy Polk, Donna F. Dodson, etc, Public *Key* Infrastructure: From *Theory to* Implementation, http://csrc.ncsl.nist.gov/pki/panel/overview.html, NIST

[2] Tim Finin, Yannis Labrou, and James Mayfield, *KQML as* an agent communicatíon language, in Jeff Bradshaw (Ed.), "Software Agents", MIT Press, Cambridge (1997).

[3] Carl M. Ellison, Bill Frantz, Butler Lampson, Ron Rivest, Brian M. Thomas, Tatu Ylonen, Simple *Public Key Certificate,* http://www.clark.net/pub/cme/spki.txt

[4] Ronald L. Rivest, Butler Lampson, *SDSI* - A Simple *Distributed Security* Infrastructu,re, http://theory.lcs.mit.edu/ cis/sdsi.html

[5] Bruce Schneier,Applied Cryptography, Second Edition, John Wiley and Sons, Inc., 1996.

[6] Matt Blaze, Joan Feigenbaum, Jack Lacy, *Decentralized Trust* management, In Proceedings 1996 IEEE Symposium on Security and Privacy, May, 1996.

[7] Sycara, K., Decker, K, Pannu, A., Williamson, M and Zeng, D., Distributed Intelligent Agents. IEEE Expert, pp.36-45, December 1996.

[8] Tim Finin, James Mayfield, Chelliah Thirunavukkarasu, Secret Agents - A Security Architecture for the KAML Agent Communication Language, CIKM'95 Intelligent Information Agents Workshop, Baltimore, December 1995.

[9] Qi He, Katia P.Sycara. Personal Security Agent: KQML-Based PKI. October 1997.