WORK IN PROGRESS

Security Architecture for Open Agent Systems

Demchenko Yu., <demch@nlnetlabs.nl>, NLnet Labs
Overeinder B., <bjo@cs.vu.nl>
Boonstra H.M. < hidde@cs.vu.nl>
Interactive Intelligent Distributed Systems, Faculty of Sciences, Vrije Universiteit Amsterdam

Abstract

This paper presents a Security Architecture for open Agent Systems based on recent developments in security technologies for service-oriented applications, particularly, XML and Web Services Security and OGSA Security.

The proposed security architecture is built as an additional middleware service of the AgentScape Operation System to provide secure agents operation in an open network environment with multiple hosts, locations, and security administration domains. The security architecture includes a communication layer, a messaging layer, a policy layer, and a security services layer. Newly developed security services provide the necessary functionality for secure access to and interaction with external Web Services using related security mechanisms and services.

The contribution of this paper is an analysis of security and trust issues in mobile agent systems, a feasibility study how existing XML-based security technologies can be deployed, and recommendations for building test and development environments using available Open Source software.

Keywords

Mobile agents, open agent systems, security architecture, XML security, Web Services security, OGSA security, security tools

1. Introduction

Over the last decade computer systems evolved from a centralised host based model to a distributed client/server model with host based resources and services. From there it evolved further to the Service Oriented Architecture (SOA) and its two implementations: Web Services Architecture (WSA) [1] and Open Grid Services Architecture (OGSA) [2]. WSA introduces a new abstraction layer to describe Web Services as stateful non-transient entities using Web Services Description Language (WSDL). OGSA addresses the specifics of Computer Grids as data-centric applications and defines a Grid Service as a stateful transient service.

WSA and OGSA provide a good basis for the further development of a new computing paradigm using mobile agent systems. Mobile software agents are programs that are goal-directed and capable of suspending their operation on one host system and moving to another location/platform where they can resume operation [3]. A mobile agent is defined as a program that can exercise an individual’s or organisation’s (or owner’s entity) authority, work autonomously towards a goal, and interact with other agents and external services [4].

Mobile agent technology will play an important role in extending the usability of service-oriented applications. The software agent model will enable service-oriented environments to be more distributed and heterogeneous: autonomous software agents interacting with other agents, entities and services to perform their tasks in Web Services environment that has well-defined description and discovery framework.

Interoperability between different agent platforms is defined by the Foundation for Intelligent Physical Agents (FIPA) specifications [5, 6]. Currently, these specifications define agent communication and interaction interoperability between agent platforms, but do not specify agent mobility. The MASIF standard [21] specifies an interface to support agent management, agent transfer, and agent tracking functions. MASIF also defines an interface for maintaining a dynamic name and location database of agents, places, and agent systems. The FIPA and MASIF work are somewhat complementary, in that FIPA has so far primarily worried about agent to agent communication (ACL, negotiation protocols, ontologies), while MASIF has primarily considered mobility and, to some extent, issues involved in interoperability among heterogeneous agent systems.

Mobile agents in open network systems put strict security requirements, both from agent and host side. Generic mobile agent security objectives are identified in the NIST Special Publication 800-19 [3]. The document also provides an analysis of the threats, it discusses possible countermeasures and it identifies the remaining problems and areas for future research.

The integration of existing security infrastructures used in Web Services Architectures, with agent platforms for mobile agents is addressed in this paper. Given the security requirement of mobile agents in open network systems, a feasibility study is presented showing the deployment of WSA security infrastructures to provide a security architecture for mobile agent systems.

The next section presents an overview of the AgentScape middleware, and threats and security requirements for mobile agent systems in general. Section 3 provides an overview and analysis of existing technologies that can create a basis for the proposed Security Architecture for Agent Systems (SAAS) in Section 4. This architecture makes use of XML/Web Services technologies to provide the secure operation of agents in an open distributed environment. Section 5 sketches the current prototype implementation of SAAS and the used Open Source components.

2. System Overview and Security requirements

2.1. AgentScape Operating System

The AgentScape Operating System (AOS) provides an open, scalable, extensible and secure operational environment for distributed interactive intelligent agents [7]. These goals are realised by separating the agent’s functionality from the basic operational services and providing these services to agents via a standard interface, which can be accessed through an Agent Server.

The AgentScape OS includes such components as the Agent Factory, location services, directory services, and location managers. The AOS supports active entities (mobile code or agents), passive entities (data entities or object which can be distributed) and services that can be dynamically deployable in the specific location. A location is a set of hosts connected by low-delay network, run by a single administration and running an AOS kernel on each host. Agents have access to the AOS middleware services/interface via its agent server.

Security services are a component of AgentScape middleware and provide a secure infrastructure for agents’ operation. Most security services can be implemented as additional services to the operational architecture, however some security services may require changes to the operational environment.

An important issue in agent's mobility is the preservation of the agent's state (or operational context, which may include security context (identity, authentication and authorisation), history or auditing records) when it is moving to another location or suspending/resuming its operation. The agent's code, the necessary data and the security contextual information is included into the Agent Container. This agent container may be sent to another host or location. The Agent Container is a data structure for storing and retrieving an agent's code, its data, and meta-information about the agent.

AgentScape is intended to provide a common middleware for running more specific application-oriented agent systems, one of which is Mansion. Mansion is an application-oriented extension of the AgentScape OS to such host specific virtual-world applications as shopping world, library world or game world. Mansion specific middleware is configurable according to the "world" specification and uses standard services provided by AgentScape.

The Mansion security framework for a mobile agent system is described in [8] and defines basic security requirements to underlying AgentScape middleware, in particular, the Mansion's security enforcement policy uses available security enforcement mechanisms in AgentScape.

2.2. Threats analysis and general security requirements

General security requirements that should be supported by any computer or network system include confidentiality, authentication, authorization, availability, integrity, and non-repudiation. The specifics of their realisation for a particular application area or system depend highly on the concrete security and threats model.

Jansen and Karygiannis [3] present a detailed analysis of security threats and describe countermeasures that are mostly targeted at the agent's operational environment and at the platform realisation. Although the described measures have general applicability, their use in an open multiplatform and multi-location/-domain environment require another approach. This approach separately addresses the operational environment, the middleware infrastructure, and the trust / role-based functional relations between major principals (e.g. authorisation, authentication and identity management).

Four threat categories are identified in [3]: threats originated from an agent attacking agent platform, an agent platform attacking an agent, an agent attacking another agent, and other (external) entities attacking the agent system. Depending on the target, security threats can be split into four main categories: disclosure of information, corruption of information, unauthorised use of resources and denial of service.

Because agent systems have specifics of impersonating or representing their owners in carrying or accessing sensitive information or carrying authorisation tokens to use valuable (or often monetary resources), they attract a wide selection of potential attackers. These attackers include internal actors (agent owners, location owners, service owners), external actors (hackers and script kiddies, professional thieves, terrorists, RIAA violators.

Most of the external attacks on the whole system (denial of service, unauthorised access to private information and resources) could be considered to fall into the general information technology security area and therefore should be prevented by applying related methods.

2.3 Specifics of the Security Architecture for Mobile Agent Systems

The ability of an agent to move between administrative and security domains imply new features and requirements applicable to the security architecture of its operational environment. As agents are mobile, they can operate over multiple trust and administrative domains and must preserve their state, their security context and their operational context when suspending/resuming or moving to another location.

In the context of currently existing and developing technologies that provide end-to-end (or client/server) security context, or are data and service oriented, agent systems have the following specifics (however not limited by this list):

1) Agents are mobile and therefore they cannot carry secure credentials with them, however they can carry the authenticated identity assertions, authorisation token, or other security assertions that should allow them to access their credentials.

Security services of an agent platform should have the possibility to request confirmation of an agent's identity or credentials based on carried security assertions. As an example, the Liberty Alliance Project (LAP) identity management framework allows an agent/entity to provide authentication assertions instead of credentials itself, if trust (or business) relations can be established between an agent's identity service and a local service provider (or local/trusted identity service).

2) Agents have limited credibility by definition. However it should allow an agent to act (possibly on temporary basis) even when no direct trust chain has been established between the agent's identity service and a local service:

- The agent system should provide the possibility for limited credibility of an agent based on anonymous access/association in Virtual Organization (VO): an agent may carry tokens of limited credibility that normally should allow an agent to act on an anonymous basis (like money cash in real life).
- The level of credibility and trust should depend on type of an agent's credentials and the local policies.

3) Agents may have a changing and/or multifaceted identity when they move from one location to another (or from one administrative and trust domain to another), and therefore there should be a provision for tracking an agent's history/evolution.

4) An agent should be capable to enter dynamically created association, and in particularly a VO, and act on anonymous basis, however there should be a possibility of tracking an agent's identity depending on local policies.

3. XML based Security technologies overview

3.1 New paradigm of application security

Recent developments in security architecture for XML Web Services and Computer Grids have been resulted in adopting a new paradigm of the document-centric security architecture that builds security services and mechanisms around semantic object, in contrast to connection/host based security approach in client /server architecture.

Document centric approach is a necessary paradigm shift for providing security functions specific for stateful transient services with distributed resources, in particular:

- cross-domain identity management and identity federations/associations

- binding security context to the authenticated identity and enabling role-based access control (RBAC)
- binding security context to documents or data crossing multiple administrative and security domains

The current application security framework is based on two key technologies, namely Public Key Infrastructure (PKI) and XML Security. These technologies include the following components:

- A PKI and a PKI based trust management framework.
- An Authentication framework and Identity management.
- An Authorisation framework and a RBAC framework.
- XML Security as underlying technology for security assertion and context management.
- The Web Services Security framework and the OGSA Security Architecture as technologies for managing stateful transient service oriented applications.

The Public Key Infrastructure is a currently well-developed technology with a number of standards and wide deployment. PKI defines a framework for binding an identity with credentials (e.g. an identity's public key) using Public Key Certificates (PKC, RFC 3280), or for binding an identity with attributes (e.g. roles or privileges) using Attribute Certificates (AC, RFC 3281). Proxy Certificates are used for delegation and impersonation. PKC and AC create a basis for the Authentication and Authorisation framework, the Privilege Management Infrastructure (PMI) [9] and for Role Based Access Control (RBAC) [10].

PMI and RBAC separate Authentication and Authorisation (AA) functions giving the home organisation (or identity provider in other architectures) to authentication user and resources or target to make authorisation decision. Main components of distributed AA architecture include Authentication (AuthN) function, Access Enforcement Function (AEF) or Policy Enforcement Point (PEP), Access Decision Function (ADF) or Policy Decision (PDP), and Access Control Information (ACI) or Policy module.

Authorisation or access control in general are provided by the target or requested resource, and decision is taken according to the access policy based on provided by user or entity credentials which may include identity credentials, privileges or roles or other information provided by Authentication service. All or part of AA information can be provided by the user (push model) or requested by Authorisation service (pull model). Accordingly, ADF/PDP modules can also work in push or pull modes.

XML Security technologies provide a solid base for document oriented security services and mechanisms and include:

- Basic XML Security suite comprising of W3C standards XML Signature [11], XML Encryption [12], and XML Key Management Specification (XKMS);
- XML Security profiles for Authentication, Authorisation and Access Control including OASIS standards SAML (Security Assertion Markup Language) [13], XACML (eXtensible Access Control Markup Language) [14];
- Web Services Security (WSS) foundation defining security profiles for SOAP messages exchange and binding [15].

The powerful feature of XML Signature is that it can both sign only specified parts, or a set of parts of a XML document. A XML Signature can be attached to the document or stored separately. Other signers can also selectively sign the whole document or its parts including or excluding already existing signatures. Analogous XML Encryption can encrypt selectively any part of XML document. XML Encryption and XML Signature can interoperate.

Application-oriented security technologies and profiles include the WS-Security Framework developed by major industry partners such as IBM, Microsoft, VeriSign, RSA Security and BEA Systems [16], the Liberty Alliance Project Network Identity Management Framework (LAP) developed by wide industry Liberty Alliance [17], and the OGSA Security Framework under development by the Global Grid Forum [18].

3.2 Platforms for service-oriented applications

Two complementary platforms are currently available for developing service-oriented and semantic based applications: Web Services Architecture (WSA) developed by the industry and OGSA (Open Grid Services Architecture) developed by the Global Grid Forum (GGF).

WSA includes the core specifications SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) from W3C and UDDI (Universal Description, Discovery, and Integration), which together provide a service description, discovery and messaging framework for Web Services applications.

Extended WSA includes specifications such as: WS-Policy, WS-Coordination, WS-Transaction, WS-Inspection, WS-Addressing, and WS-Security framework [16]. WS-Security describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. These mechanisms can be used to accommodate a wide variety of security models and encryption technologies. WS-Security also provides a general-purpose mechanism for associating security tokens with messages and it describes how to encode binary security tokens, in particular, X.509 certificates, Kerberos tickets, and encrypted keys. The WS-Security Profile for XML-based tokens describes how to use XML-based tokens such as the SAML or the eXtensible rights Markup Language (XrML) with the WS-Security specification. It also includes extensibility mechanisms that can be used to further describe the characteristics of the credentials that are included within a message.

Other specifications from the WS-Security stack include WS-SecurityPolicy that specifies a format for the policy assertions, and WS-Trust that enables Web Services to request and issue security tokens and to manage trust relationships. WS-SecureConversation defines extensions for secure communication by establishing and sharing security contexts, and deriving session keys from security contexts.

Recently published WS-Federation defines mechanisms for federated identity management. These mechanisms are used to enable identity, attribute, authentication, and authorization federation across different trust realms [19]. The federation model extends the WS-Trust model with descriptions of how identity providers act as security token services and how attributes and pseudonyms can be integrated in security token mechanisms to provide federated identity. Tokens can represent the principal's primary identity or some pseudonym. Services can request attribute/identity service based on a provided token/pseudonym to obtain authorised information about the identity. WS-Federation active requestor and passive requestor profiles define how the cross trust realm identity, authentication and authorization federation mechanisms can be used by active requestors (e.g. SOAP-enabled applications), or by passive requestors (e.g. Web browsers) to provide Identity Services. The functionality provided by WS-Federation is similar to identity federation provided by Liberty Alliance Project (LAP).

WSA defines service according to Service Oriented Architecture (SOA) concept as a well-defined set of actions, it is self-contained, stateless, and does not depend on the state of other services. "The description of a service in a SOA is essentially a description of the messages that are exchanged. This architecture adds the constraint of stateless connections, that is where the all the data for a given request must be in the request” [1, 20].

Grid Service, defined in OGSA (Open Grid Services Architecture), extends the definition of a service to describe the stateful and dynamic character of the Grid Service. WSA is build around request/response while OGSA is built around the data or a semantic object. OGSA extends WSA in order to allow working with transient stateful services, like most of the data-centric tasks in Computer Grids that may span for a long period of time and between multiple locations.

The above described technologies, together with other related security technologies, can provide a good basis for developing the Agent System Operational environment and ensures easy integration with rapidly developing applications based on WSA and OGSA. In this context, agent codes can be considered to be semantic and data objects moving between locations - administrative and trust domains. The major issue of building semantic based infrastructures is to provide a reliable/secure binding between the object/data and their identity/meaning in the context of specific environment or subject area. WSA and OGSA can provide the necessary functionality for this.

4. Security Architecture for open Agent Systems (SAAS)

4.1 Components of a proposed Security Architecture

The proposed Security architecture for distributed open Agent Systems (SAAS) is built on top of existing and emerging WS-Security and intended to be compatible with emerging Web Services and OGSA platforms. WSA and OGSA define a generic framework for which specific profiles must be defined which describe the message formats and interfaces needed to implement application oriented services and protocols.

The Security Architecture for open Agent Systems includes the following layers and components (see Fig. 1, from the bottom up):

1) The Communication/Transport Security Layer which defines network infrastructure security and uses network security services such as SSL/TLS, IPSec, VPN, SASL, and others.

2) The Messaging Security Layer which is based on the currently well defined, and supported by different Web Services platforms, SOAP/WS-Security. This layer also uses relevant XML Security mechanisms like XML Signature, XML Encryption, SAML and XACML security token exchange format.

3) The Policy Expression and Exchange Layer which defines a set of policies applicable to Agents and the Agent operation environment which are required to ensure multi-domain and multiplatform compatibility. The Policy layer provides the necessary policy information for the Service/Operational Security layer.

4) The Services/Operational Layer defining security services/mechanisms for secure operation of Agent Systems in an open environment. This layer includes:

- Secure Context Management
- Identity and Credential Translation
- Authorisation and Access Control
- Auditing and Non-repudiation

Some of layers and components are described below in more detail.

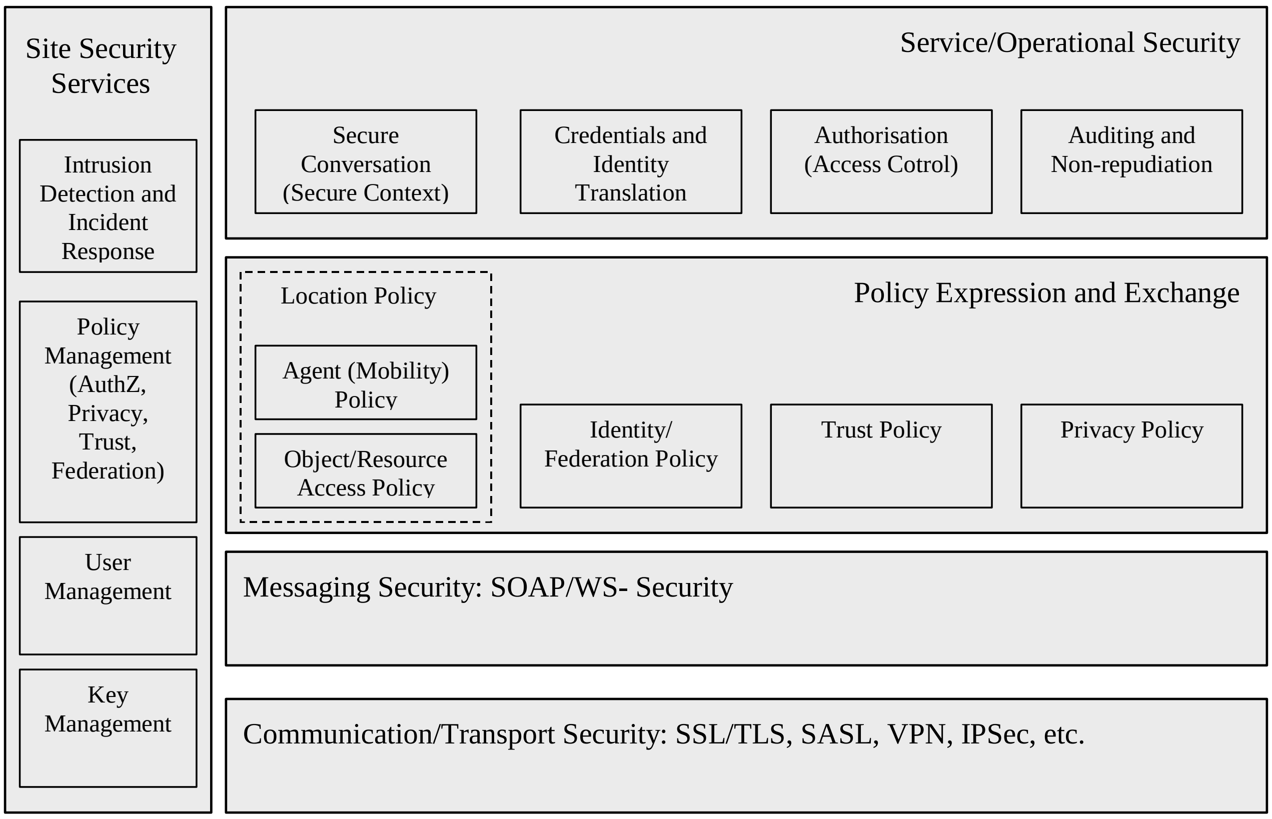
Site Security Services
Intrusion Detection and Incident Response
Policy Management (AuthZ, Privacy, Trust, Federation)
User Management
Key Management
Service/Operational Security
Secure Conversation (Secure Context)
Credentials and Identity Translation
Authorisation (Access Cotrol)
Auditing and Non-repudiation
Policy Expression and Exchange
Location Policy
Agent (Mobility) Policy
Object/Resource Access Policy
Identity/ Federation Policy
Trust Policy
Privacy Policy
Messaging Security: SOAP/WS- Security
Communication/Transport Security: SSL/TLS, SASL, VPN, IPSec, etc.

Fig. 1. Security Architecture for open Agent Systems.

a) The Policy Expression and Exchange Layer

Communicating agents/principals need to conform to certain requirements in order to securely interact. It is important that service or resource requestors have access to, and understand policies associated with, the target service. As a result, both the service requestor and the service provider select the acceptable security profile. It is also important to mention that the privilege to acquire security policy is given by the hosting environment to authenticated and authorised entities only.

The policy expression and exchange suite includes (but is not limited to) the following policies:

- agent system location security policies comprising in general of an agent processing and mobility policy and a resource access policy
- a identity association and federation policy
- a trust policy, and
- a privacy policy.

The policy layer provides necessary information for policy enforcement modules of the Service/Operational Security layer. It is suggested that policy expressions should confirm to WS-Policy that provides an extensible framework which is configurable for specific applications based on several common attributes. These attributes include privacy token requirements, security token requirements, and information encoding and supported algorithms.

b) Secure Context Management

The Secure Conversation service will adopt and leverage WS-SecureConversation specification to maintain consequent message exchange between the components of agent systems that may span over open network environment. Secure Conversation will maintain the secure context, established during initial mutual authentication, throughout an active communication session between interacting application endpoints. Secure Conversation will operate at the SOAP message layer also providing a binding with the policies associated with the end points.

c) Identity and Credential Translation

The agent operational environment typically consists of multiple locations and security domains that maintain their independent security services and policies. Operations between entities in different domains will require mutual authentication. Different security domains may incorporate different formats and semantics for requestor/provider identities and credentials. Interoperation will require federation of the involved domains and translation or mapping of identities and credentials. The federation may also be accomplished through trusted proxies or broker services.

Identity and Credential translation service can be built on two currently available identity management specifications: WS-Federation and Liberty Alliance Project [17, 19].

d) Authorisation and Access Control

The Authorisation and Access Control security service is a key part of the managed security in an open service oriented environment. Authorisation is typically associated with a service provider or resource owner, who controls access to a resource based on credentials or attributes provided by the requestor that define the requestor's privileges or roles bound to requestor's identity. Separation of Authentication and Authorisation services allows dynamic role based access control management and virtual association between interacting entities. The separation provides a basis for privacy in an open environment.

The Authorisation and Access Control service for the Agent System will re-use models proposed in WS-Authorisation and in the OGSA Authentication and Authorisation framework. It will be based on security token definition and exchange protocols defined in SAML and XACML [13, 14].

e) Auditing and Non-repudiation

Auditing and non-repudiation are components necessary for security services assurance and policies enforcement. They provide the secure logging functionality that is necessary for many higher level audit related functionalities. Some limited auditing functionality may be required for other services at the Service/Operational Security level, in particular, timestamping.

f) Security Services Management

Effective and reliable operation of the security services requires underlying management of security services that includes:

- key management for cryptographic functions;
- user management, including user registry and related role or privilege management;
- policies management which includes local operational security policies, services security policies and trust management;
- intrusion detection and incident response capability.

4.2 Example Scenarios

Three examples are described below to illustrate how the main components of the proposed Security Architecture interoperate to provide basic security services for Agent Systems. They also provide a basis for modelling SAAS operation using the test and development environment described in the next section.

a) Communication between two agent hosting locations over the Internet

This is a very basic example, however it covers most of the communications between different Agent hosting locations or platforms that belongs to one administrative or trust domain. The trust infrastructure is established and provided by local services based on mutual/pairwise business or trust agreements. This means that authentication and identification services at different locations trust each other.

The agent or host opens a network connection to the target host or service using a service from the Communication layer (e.g., SSL/TLS) that provides a secure communication channel, and a secure messaging service (e.g., SOAP over HTTP) from the Messaging layer. The requestor/sender includes a security token into the message that contains the authentication assertions together with the credentials or authorisation attributes. Based on the provided information the target system or the remote agent hosting platform makes an authorisation decision and provides the requested service or action. If a requested action is the relocation of an agent, the new host will check integrity and authenticity of the agent container (AC), verify the mobile agent's credibility, check the embedded policy and impose local policies to the agent's functionality or mission. Differences may be in authorisation or policy enforcement model using push or pull model.

The AC may be sent as a body of or as an attachment to a SOAP message. Security tokens may be exchanged using the SAML request/response protocol. SOAP and SAML, which are based on XML, also allow using XML Signature and XML Encryption to provide message integrity and confidentiality.

b) Using third party to establish trust relations

This scenario addresses a case where an agent is to be sent to, or a resource or service is to be requested from, a location with which there is no established trust or business relations. But these relations can be established via a third party trusted by both home and target locations. This may be an Identity Provider or a Service Broker, or other kind of trusted introducer service.

In this scenario the home/source location asks the trusted third party to provide authentication assertion and confirm the agent's identity. The agent or host will provide all the necessary information to the third party. The third party replies with the authentication token and authorisation attributes that may contain confirmation of the agent's public key (in a form of Public Key Certificate) and of the agent's roles or privileges (in a form of Attribute Certificate). Now the agent or source host can request a service from the target location providing the obtained security tokens that will be trusted by the remote location.

WS-Federation, together with other WS-Security components or Liberty Alliance Identity management, provides the framework for establishing indirect trust relations and exchanging security context necessary for this scenario. SOAP and SAML provide standard semantics and a standard protocol for exchanging the required information.

In Situations where there is no possibility to establish trust relations directly or via third parties one may use such technologies as a "leap-of-faith" or a model similar to credit card clearinghouse. However these technologies don't impose new requirements to the proposed Security Architecture.

c) Accessing external service

An agent typically will need to access some external services to request an authorised or public service. It is foreseen that in the future more and more services will be available as Web Services accessible through the standard WSDL interface. As an example, Google is already providing specialised search services as a Web Service with a WSDL interface. If a Web Service is provided as a public service, its WSDL description can be retrieved from UDDI registry and used by the requestor to send an appropriately formatted request using a SOAP message. If a Web Service requires authentication and authorisation, all necessary communications for establishing and exchanging a security context will be conducted using WS-Security or the Liberty Alliance framework. These communications can be based totally on SAML protocol and semantics.

5. Building a test environment for SAAS components

The proposed SAAS is built as an open architecture based on XML and Web Services security. It works at the middleware level and can be deployed as an additional security service for existing AgentScape middleware providing a standard platform for communication over open network environments and for accessing external Web Services.

There is a complete suite of Open Source software written in Java for the deployment of Web Services supporting SOAP and WSDL as web applications. The most popular and widely used web application server is Jakarta Tomcat created by Apache Foundation. Tomcat is the servlet container that is used in the official Reference Implementation of the Java Servlet and JavaServer Pages specifications. However, it requires a number of supporting packages and libraries. Some other software will be necessary for key management, SOAP messaging, XML Security and XML Schema development.

The following software components are necessary to enable services at the first two SAAS layers (i.e. the Communication/Transport Layer and the Messaging Layer):
1) A Tomcat web application server version 4.1 or later - http://jakarta.apache.org/tomcat/
2) Apache Axis as an Open Source SOAP server and client – http://ws.apache.org/axis/
3) Key management tools to generate, export or import server and client keys and certificates; available possibilities are: the native Java key and certificate management tool, the OpenPGP key tool and JCE Provider; IBM KeyMan;
4) XML Security packages: the Apache XML Security library and the xss4j Security suite;
5) tools and libraries for working with XML documents: the Xerces XML parser and the Xalan XSLT processor from Apache;
6) the OpenSSL library distributed with The Apache HTTP Server or available as a separate Java JSSE library.

The Libraries and implementations of the SAML based Authorisation and Authentication services and of the Identity management services are distributed with the Interoperability Prototype for Liberty (IPL, http://developer.java.sun.com/developer/codesamples/liberty.html) and Shibboleth (http://marsalis.internet2.edu/cgi-bin/viewcvs.cgi/).

It may be also useful to have integrated XML development tools like XMLSpy (http://www.xmlspy.com/) that in its full Enterprise Edition supports XML Schema design, XSLT, WSDL and SOAP. XMLSpy also allows to generate program code in Java, C++, C# for a library from a created schema. It also contains a WSDL editor and a SOAP debugger.

Models made by authors showed complete feasibility of the proposed approach of building SAAS based on Web Services and XML Security technologies. The wide availability of tools for XML and WS and their high compatibility made it possible to quickly create a test environment that provides the full functionality needed for the first two SAAS layers and a necessary platform for the higher level services from the Policy and Security Services layers. Using a layered structure and well-defined standard interfaces allows to test all SAAS components in non-production test environment and the gradual deployment of additional security components.

Summary

The Proposed Security Architecture for open Agent Systems (SAAS) provides a framework for designing security services in a structured way. It incorporates existing and emerging security technologies from the XML Security and the Web Services Security world and addresses specific the requirements of mobile agent systems. The security architecture includes communication and messaging layers, a policy layer and a security services layer. New security services provide the necessary functionality for secure communication between agent locations over an open network environment and for securely accessing and interacting with external services which may use WSDL interface and related security mechanisms. The proposed architecture can operate in an environment consisting of multiple administrations and trust domains.

From the design point of view, the proposed architecture provides an open extendable framework for developing security services that can be added as additional security services to existing middleware and can be deployed gradually. This approach also allows to benefit from a variety of existing development tools for XML and Web Services based applications.

The test and development environment created by the authors using Open Source software demonstrated feasibility of the proposed approach and created a basis for further development of application specific components including policy definitions, trust models, identity management and access control.

Policy and trust related issues that define different aspects of mobile agents' operation and functionality can be expressed using standard semantics and mapped to related semantics of basic security services and message format and exchange protocol. However, this task requires specialist knowledge and will remain for the future work.

Acknowledgements

This work in general was supported by NLnet Foundation (http://www.nlnet.nl/). Technology research and development related to XML Security issues is a part of the Fonkey project developed by NLnet Labs (http://www.nlnetlabs.nl/donkey/). The authors acknowledge the contribution made by Erik Rozendaal.

Reference

1. Web Services Architecture, W3C Working Draft 8 August 2003 - <URL: http://www.w3.org/TR/ws-arch/ >
2. The Open Grid Services Architecture Platform. <URL: http://www.gridforum.org/Meetings/ggf7/drafts/draft-ggf-ogsa-platform-2.pdf >
3. NIST Special Publication 800-19. Mobile Agents Security. October 1999. - <URL: http://csrc.nist.gov/publications/nistpubs/800-19/sp800-19.pdf>

4. Picco, G.P., Mobile Agents: An Introduction, *Microprocessors and Microsystems*, vol. 25, no. 2, pp. 65–74, April 2001.
5. FIPA Abstract Architecture Specification. Published December 3, 2002. - <URL: http://www.fipa.org/specs/fipa00001/>
6. FIPA Agent Management Specification, Published December 6, 2002. - <URL: http://www.fipa.org/specs/fipa00023/>
7. AgentScape Operating System. <URL: http://www.iids.org/projectfolder/agentscape>
8. Noordende, G. van 't, Brazier, F.M.T. and Tanenbaum, A.S. (2002), A Security Framework for a Mobile Agent System - <URL: http://www.iids.org/publicationdata/db/111/pubdetail>
9. ITU-T Rec. X.812(1995) | ISO/IEC 10181-3:1996, Information technology - Open systems interconnection - Security frameworks in open systems: Access control framework.
10. Role Based Access Control (RBAC) – NIST, April 2003. - http://csrc.nist.gov/rbac/
11. XML-Signature Syntax and Processing. W3C Recommendation. 12 February 2002 - http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/
12. XML Encryption Syntax and Processing W3C Recommendation 10 December 2002 - http://www.w3.org/TR/xmlenc-core/
13. Security Assertion Markup Language (SAML) v1.0 - OASIS Standard, 5 November 2002 - http://www.oasis-open.org/committees/documents.php?wg_abbrev=security
14. eXtensible Access Control Markup Language (XACML) Version 1.0 - OASIS Standard, 18 February 2003 - http://www.oasis-open.org/committees/documents.php?wg_abbrev=xacml
15. Web Services Security Framework by OASIS - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
16. Security in a Web Services World: A Proposed Architecture and Roadmap - http://www-106.ibm.com/developerworks/webservices/library/ws-secmap/
17. Liberty Alliance Specification suite - http://www.projectliberty.org/specs/index.html
18. The Security Architecture for Open Grid Services - http://www.globus.org/ogsa/Security/
19. Web Services Federation Language (WS-Federation) Version 1.0 - July 8 2003 – http://msdn.microsoft.com/ws/2003/07/ws-federation/
20. A Grid Application Framework based on Web Services Specifications and Practises. - http://www.neresc.ac.uk/projects/gaf/
21. Mobile Agent System Interoperability Facility. – OMG Standard. - http://www.fokus.fraunhofer.de/research/cc/ecco/masif/