Securing Agent-based e-Banking Services

Juan Jim Tan, Leonid Titkov, Stefan Poslad

Department of Electronic Engineering, Queen Mary, University of London,
Mile End Road, London E1 4NS
{juanjim.tan, leonid.titkov, stefan.poslad}@elec.qmul.ac.uk

Abstract. Services being developed by the EU Agentcities project require, and would benefit from having security. This paper focuses on analysing and specifying agent security services for an Agentcities e-Banking service. In this version of the service we have defined a set of requirements for core, general and Multi-Multi Agent Systems (MMAS) scenarios to support confidentiality, integrity, and authentication interactions. We have applied an Abstract Security Model for mapping relationships between Assets, Safeguards, and Threats. Profiles define the mapping between these entities in order to meet the system requirements and policies represent collaboration rules to achieve the goal of meeting the requirements. Consequently, the Security Model implementation in this scenario has demonstrated that agent based commerce services are plausible in MMAS environments.

1 Introduction

The Agentcities network[1] represents the first attempt to build an open global standard for agent-based infrastructure intended for research and future commerce. Therefore an Open Service Model to support MMAS interoperability is essential. For this reason, we have designed a flexible high-level security model and applied it to an e-Banking scenario.

In order for an e-banking institution to support payment mechanisms for the purchase and sale of goods, an appropriate level of security is needed to secure e-commerce transactions within the Agentcities network [1]. The security service has initially focussed on providing security support for the e-Banking service but there is already ongoing work to extend this security support to other services and transactions within in Agentcities.

1.1 e-Banking Services

The expression “e-Banking services” refers to the set of processes and mechanisms that a virtual agent-based banking institution offers to agents intending to engage in

[1] http://www.agentcities.org/

commerce activities in an agent environment [1]. The e-Banking service consists of two sub-services:

- Electronic payment service for enabling agents to make and receive payments;
- Account Management service for creating, maintaining and closing bank accounts.

In the following sections we analyse use-cases for secure versions of each of these with respect to the e-Banking service. These scenarios provide two examples of use cases from the perspective of security service users. In section 2, we specify an abstract security model for supporting security in MMAS environments and a reification of the model into the e-Banking scenario. In section 3 we discuss use cases from the perspectives of security service providers. The security provider use-cases are more complex because the provider must specify the management and distribution of security credentials for authentication and authorisation and the establishment and closure of protected communication channels. Finally, we have implementations and results, and a short conclusion to sum up our work.

1.2 Agent-based e-Banking services

The security service is not highly integrated with the e-Banking service. It is loosely coupled for reusability where it could offer security with other services and to explore opportunities for offering security to open services. In some cases the need for securing combinations of services are necessary. The e-Banking service provides a sufficient indication for capturing fundamental requirements to support the design and instantiation of our modular security service.

1.3 Security Requirements for e-Banking

In order to generate the security requirements for the e-Banking service we have modelled a series of use-cases. A portion of one of the use-cases for a credit card transaction that is conducted after goods have been selected from a merchant is shown in Figure 1.

To justify the design of any security system it is necessary to define elements in the system that requires protection, to analyse the possible threats to these elements, and to model a security mechanisms that will provide protection for these elements.

In the security requirements for the e-Banking services, we do not aim to satisfy all the complex security requirements from the above use-cases, but instead aim to satisfy only the core requirements. Based on the use-cases for secure payment and secure account management we derive the following core security requirements: Authentication, Authorisation, Message Confidentiality, and Message Integrity.

1. Both the buyer and seller usually require authentication to establish identity. The vendor is authenticated based on physical location and name, the customer supplies credentials specific to this (payment) service such as a credit card. The customer's credentials may be verified against other credentials such as driving-license or by contacting the issuer of the credentials.
2. The customer or merchant initiates a private communication channel to cover the payment transaction;
3. The customer gives the payment credential, for example credit card details, to the merchant, expecting confidentiality (i.e., non-disclosure of the number to a third party);
4. The customer also expects integrity (i.e., no change in transaction amount) from the merchant;
5. To do this the bank must authenticate the merchant and merchant must authenticate the bank, hence we have the payment body, which is the customer's bank that is designated or associated with the client authenticating the merchant. The merchant also authenticates the payment body (using credentials based upon location, telephone number, etc). Authentication occurs over a second secure channel between the merchant and the customer's bank.
6. Once the customer's bank authenticates the merchant's ID and role, it gains the authority (this is linked via some policy to the authorisation and the role) to check that the payment amount, i.e., getting approval (authority) for the purchase amount by a credit card clearinghouse.

Figure 1: Part of a use-case for the payment part of an e-banking service

This version does not offer support for non-repudiation of messages, it is assumed that the agents are trusted not to repudiate.

1.4 MMAS Infrastructure Requirements

The domain specific security service requirements, in this case the e-banking domain, needs to be considered in conjunction with the security requirements of the underlying agent based infrastructure.

MMAS requirements can be divided into intra and inter platform security requirements. For the former case, agents are usually contained in a homogeneous domain and single MAS. In the latter case, it means that the participating agents are distributed over various agent platforms and a possibly ad hoc domain cluster.

1.4.1 Intra platform security requirements

For inter platform security, some of the requirements gathered are the following [13]:

- Authentication of agents by facilitators when writing to directories accessed via facilitator agents such as the FIPA AMS (Agent Management System) and the DF (Directory Facilitator). This helps prevent one agent masquerading as another agent and changing directory information it doesn't own.

- Authentication of facilitators by agents so that agents are able to trust that information and requests sent to them by facilitator agents is valid.
- The use of a private channel for transferring messages between agents when required. This helps prevent malevolent agents stealing private information belonging to others.

1.4.2 Inter platform security requirements

For the inter platform security it is necessary that:

- Message interoperability is achieved, i.e. two platforms, positioned in different geographical locations (either part of the same domain or different ones) should either share the common ontology or some sort of conversion mechanism exists, so the message sent from one platform will be understood by another.
- Single sign-on on multiple domains must exist so the agent authenticated and authorised on one platform should be able to use their privileges without the need to repeat authorisation and authentication processes.

2. Abstract Security Model

Most MAS (Multi Agent Systems) have been used and tested in homogeneous rather than heterogeneous environments. These homogeneous configurations span from single to multi domains, but lack the infrastructure or support for dynamic and flexible heterogeneous systems. In terms of developing security, various models have been investigated but they are restricted because they:

- Are oriented towards low-level, homogeneous, static security models;
- Offer no holistic security model: there are many security specifications but it is not clear how they are combined;
- Are impartial match for open services, in which there is a mismatch of security instantiation between heterogeneous systems.

Therefore, there is a need for a heterogeneous security model that offers a holistic multi faceted approach to support agent interactions within Multi Agents Multi Domains (MAMD).

2.1 Security Model Definition

In this section, we define an abstract security model that maps to general agent infrastructures where a variety of domains exist to support various arbitrary entities in various societies or communities. The model expresses four groups of entities:

- **Assets:** Includes items of value for instance service accessibility and operation of service resources, critical information stored in a system or domain, contents and relevant data included in communication messages.

- **Safeguards:** Protection can be divided into two categories, one that is non-technical and another is technical. In the former, it includes safeguards that relates to threats rising from human error, insiders, or social threats that aren't specific classes or known vulnerabilities. As for the latter, these safeguards includes known technical issues for example cryptography mechanisms, access controls, and password systems. These safeguards closely relates to profiles and threats that it enforces and protects for a given asset. Hence, in this security model we focus largely on technical to non-technical safeguards that are identifiable as specific classes of vulnerabilities and also be able to scope our focus.
- **Threats:** Elements that causes disruption to the service or assets in a system. Threats closely relates to what safeguards tries to protect. In terms of MAMD Systems, we define threats as technical attacks to a system such as denial of service, password cracking, or man in the middle. Due to the various varieties of threats, one cannot prevent all threats from occurring; hence in our security model we use reiterative steps in improving safeguards against threats (will be discussed later). Lastly, to scope the threats we are trying to protect, our model has asymmetrical relations between each entity where the profile specifies various threats it tries to protect.
- **Profiles:** The profile contains entities that express what are the threats and safeguards it wishes to prevent and enforce, in other words it also identify the sets of mappings between threats, assets and safeguards. These profiles also identify the various assets one system or domain may wish to protect at various level of granularity. The granularity may include hierarchical approached where multi systems can be protected at various steps for example at the service level, domain level, and multi domains level, and so on. The profiles may also contain manageable policies by an event-condition-action paradigm where aspects of security behaviour may be controlled by a set of rules to achieve its objectives.

2.2 Asset Security Model

In this section, we further elaborate our definitions from the previous section into an architecture model and elucidate the relationships, and collaboration of entities in our model within the context of MAMD Systems. In the following figure, we ground the system consisting various arbitrary entities with entities included in our model for supporting heterogeneous systems security.

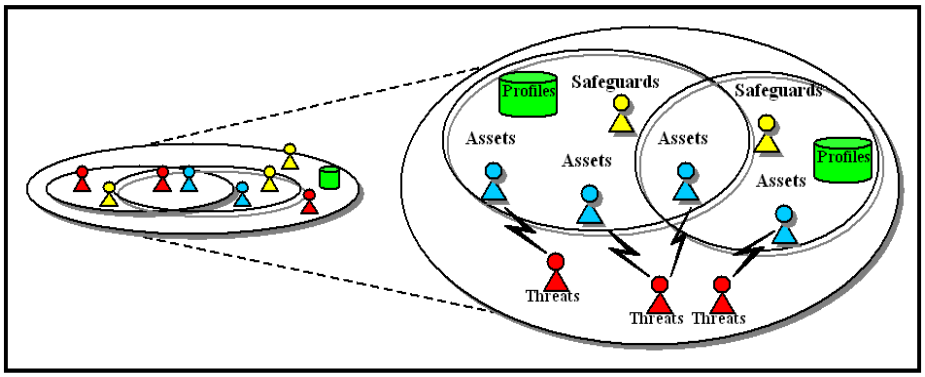

Figure 2: Abstract Security Model

In the above figure, we described our model in a typical domain scenario where various agents exist. The model contains various assets denoted as agents representing entities such as, Message Transport Service, Component Services, Platform Management Systems, Directory Facilitators and other valued assets within the system for instance information. Within the same model, Assets may define profiles or policies that map the relative security requirements, threats that it wants to prevent, safeguards it wishes to enforce and behaviour rules it would like to implement.

The profiles support a ground up relationship between assets open to threats, and safeguards against threats. Therefore the profile provides a global description of mappings, followed by threats attacking assets, and finally safeguards protecting assets against threats. The relations between the various entities are ad-hoc and can most frequently support various type of network topology in any given agent network architecture.

Within the MAMD infrastructure, we further elaborate the protection, threats, profiles and the scope of the model in which it supports. The assets in a domain specify formal profiles containing policies and mappings of items it would like to protect. The profile can be divided into two main configurations: the mappings and the service behaviour rules. We investigate the former of the two configurations in greater detail, but would leave the latter due to the scope of this paper. In the mappings we represent the security profile in the following formula, where Profile *(P)* equals Safeguards *(S) or* Threats *(T)*, where *S* equals a collection of safeguards *(s)* that maps to a variety of assets *(a)*, where *T* equals a collection of *a* that maps to a variety of threats *(t)*.

$$S = \{s_1 (a_1, a_2, \ldots, a_M), s_2 (a_1, a_2, \ldots, a_M), \ldots, s_n(a_1, a_2, \ldots, a_M)\}$$

$$T = \{a_1 (t_1, t_2, \ldots, t_M), a_2 (t_1, t_2, \ldots, t_M), \ldots, a_n(t_1, t_2, \ldots, t_M)\}$$

$$P = S \cup T$$

A profile in most circumstances could either contain both *S* and *T*, or just *S* or just *T*. These safeguards or better known as technical safeguards would guard against threats defined in the previous section. The technical safeguards mustn't only meet the requirements of addressing the vulnerabilities they are intended to mitigate, but must also be properly implemented, installed, operated, and maintained. They are also subject to abuse in cases where they are inadequate to the actual threats or where they create undue burdens on users. Technical safeguards are typically implemented by domain administrators to ward against malicious threats, but to support truly decentralised security management, assets such as services can also specify safeguards for their assets against threats they wish to protect and vice versa by service owners.

According to Cohen [10] technical safeguards are designed to cover attacks, where it ranges from special-purpose techniques that detect a specific attack to generic techniques that cover large classes of accidental and intentional disruptions. Technical safeguards also range from well thought out, well designed, carefully implemented customized solutions to poorly conceived, poorly designed, and poorly implemented off-the-shelf products. Unfortunately, most technical safeguards have not been designed to be easily and effectively managed by an average person. In fact, most of them cannot be effectively managed even by a well trained protection specialist without special training and custom made tools.

The lack of adequate protection management tools in support of technical safeguards is slowly being addressed, and over time, operational management of technical safeguards may reach a level where average users can protect themselves. But for now, this is a major shortcoming of technical safeguards that limits their effectiveness. The lack of effective tools to support protection management using technical safeguards has been a major factor in distributed systems, where no security management model is defined or implemented to support user-friendly enforcement of a security model like the one defined in this paper. Therefore in correlation with this abstract security model, a policy based management framework may be necessary in such environments where a suitable mechanism is required.

The abstract security model follows an iterative security analysis step where domains or assets' profiles can be re-examined to include additional security requirements or to ensure critical security measures are met within the system. These iterative steps called *iterations* in our model can be viewed in a dynamic perspective where assets or domains could either have single profile iteration to multiple profile iteration based on case-to-case basis. These *iterations* are orthogonal to risk analysis where *iterations* only occur when a particular risk becomes dominant. Based on recent event, one can say before the September 11 attack on the World Trade Centre, the risk of a plane crashing into a skyscraper was very low risk, but in recent times the claims of such occurrences are considered high risk. Therefore, these *iterations* supports the model based on past events that can be collectively monitored for potential security risk and becomes a provision for security improvements in within the system.

2.3 Reification of Model for e-Banking

This reification maps the abstract security model presented in previous sections of this paper into the e-Banking environment populated by FIPA Agents. This reification step contains several expressions of *Assets*, *Threats* and *Safeguards* mechanisms.

2.3.1 Assets

Assets of the e-Banking scenario can be represented as FIPA based elements [11] representing a variety of concepts, for instance the following:

Table 1: Assets Reification

Element	Description
Bank/User Agent	Entities that mutually communicate with one another to achieve e-Banking goals: transfer of funds or account management.
Message-transport-service	A **service** that supports the sending and receiving of **transport-messages** between **agents**.
Security and Service Ontologies	Security ontology includes relationships of vocabulary symbols referring to security aspects for a subject domain, as well as symbols referring to relationships that enhance upon the service when utilising security.
Accounts or Payment Services	The Account Management and Payment service provided for **agents** by the **Bank Agent**.

2.3.2 Safeguards

We can generally define safeguards as mechanisms, applications, or models that provides some of the following functionality:

- Cryptography mechanisms – encryption/decryption
- Authentication
- Authorisation
- Public Key Services – Certificate Authority.

2.3.3 Threats

Threats are orthogonal to safeguards and could share similar definitions as safeguards (see Section 2.3.2) but on the opposite scale. Therefore, threats would include some of the following attacks, for example denial of service, integrity attacks, surreptitious forwarding and many others. Threats can therefore exists as tangible or intangible forms, where applications like viruses or repudiations during various negotiations. Threats can thwart security at various levels within the FIPA Agent Communication Language (ACL) communication stack of the e-Banking scenario, for instance the following: Message Transport Level, Communicative Act Level, Ontology Level, and Interaction Protocol Level [12].

Due to the variety of security threats, combinations of safeguards against threats at the above levels are vital in supporting end-to-end security. All though strong level of security is sometimes necessary, the issue between openness vs. security in a heterogeneous distributed service environment still arises. Therefore, the use of security analysis *iterations* can be helpful in progressively improving the security measures upon a case-to-case basis.

2.3.4 Profiles

The profiles provide the run time implementation details of system behaviour rules and its objectives. It also supports the basis of the abstract security model in defining assets, safeguards and threats mappings and relationships in context with e-Banking service. The following defines the abstract model's association and interfaces in a particular profile:

- Assets mapped from e-Banking scenario elements (mappings from service elements)
- Safeguard associations with various Assets from single to multi domains
- Threats that maps under Assets, where each Asset can defined numerous threats it wishes to protect against

The e-Banking service defines these profiles, and it is verified by the security service application to reacts against threats based on definitions in the profiles. For instance, a profile within the system requires a particular content (asset) to be confidential, therefore the security service would enforce this requirement of communicating using 'encrypt and signed' policy.

2.3.5 e-Banking Service Asset Model

In the following figure, we hypostatize our abstract model as an e-Banking service asset model by initially identifying the following set of assets: accounts service, payment service, Bank Agent, User Agent, Secure Channel, Market, and Ontology.

Following which, we specify Profiles mapping relationships of safeguards protecting assets, for example message integrity for account services but message confidentiality for payment services. In addition, we could also define what each asset should protect against, for instance Bank Agent must protect against insertion replays within the interaction protocol to maintain consistency of messages. The safeguards mechanisms are enforced by the Security Service where profile requirements are met by instantiating the specific policies within the service to ensure the requirements within the profiles are met.

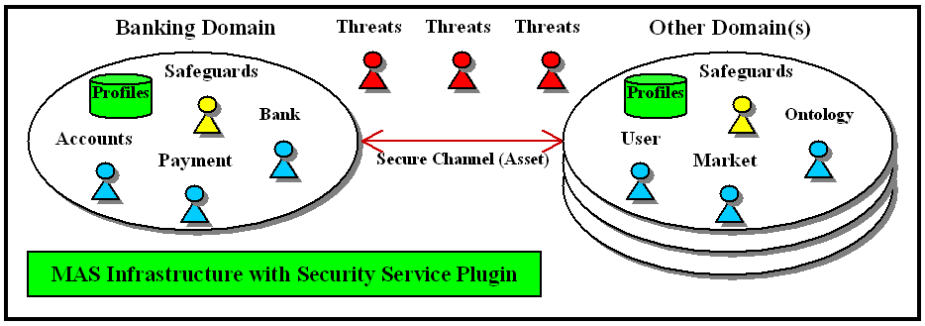

Figure 3: e-Banking Asset Model Deployed Using Security Services

Multiple banking domains may exist and domains are independent component services. There is also a different notion between an independent service domain compared to a directory service domain, where the former are domains of individual services which only host assets belonging to one administrator or company. Hence, threats do not normally occur within this boundary except in the case of the latter domain that host a cluster of smaller or overlapping domains. The possibility of threats within an independent domain is still possible but not inherent in the scope of this paper, but could still be address in our model using security services in a local boundary scenario at the expense of larger overheads.

3 e-Banking Security Service Design Issues

In this section, we further elaborate the models from our analysis by discussing the following key design issues:

- The trade-offs in modelling security services as agents versus non-agent services.
- The trade-offs in modelling security protocols at different levels e.g. transport level encryption versus application level encryption.
- The architecture of the system and how the security services are interlinked to a security service user such as the e-Banking services.
- The management of the security service such as the management of credentials to support authentication, authorisation and secure channels. This is modelled using service provider use-cases.
- Definition of security policies, e.g. when do we request certificates.

These points are interrelated.

3.1 The e-Banking Architecture

In this section, we derived a diagram of the security service architecture. The diagram includes a few actors, namely the customer, merchant, bank and CA agent (we can

also assume that there might be more than one CA). Based on figure 4 below, we explain the role of each actor based on one or mix occasions:

3.1.2 Customer Agent

This agent is the user of the service offered by a merchant, it may negotiate with a merchant during the buying and selling process. After which, authentication, secure tunnelling or encrypted messaging is used to transfer sensitive information to either the merchant or the bank. The buying process is over when the merchant receives the payment and hands over the goods.

3.1.3 Merchant Agent

The merchant agent is involved in negotiation between itself and the customer agent during a buying-selling process. The Merchant Agent may also contact the bank agent to check the receipt of payment from the customer agent before handing the goods to the customer.

3.1.4 Bank Agent

The bank agent provides account management and payment service for agents. To access services offered by the bank agent, all agents are required to undergo the authentication process, followed by a secure communication link. After, which the bank agent allows its customers to transfer-funds, create or close an account and check the account.

3.2 The Security Service Architecture for e-Banking

The security service is designed to be integrated with multiple services not just the e-Banking service. For the e-Banking, the principle requirement is to provide authentication and message integrity and message privacy. Authentication is achieved by using a Certification Authority Agent. The support for message integrity and message privacy is supported using a non-agent sub-service that is called a secure channel. All agents can locally access one end of this sub-service in order to set up a secure channel.

3.2.1 Certification Authority

Agents that wish to interact securely should always register themselves with the CA Agent. The registration process between an agent and the CA is always encrypted to provide confidentiality of public-key transfer from the agent to the CA or vice versa, this also includes the process of certificate chaining during a particular registration of a certificate. Secondly the CA also provides the service for requesting certificates of a particular agent. In this service, the certificate is not encrypted because a certificate is normally tamper-proof and classed as unclassified information.

This agent interacts with the CA Agent using certifcate request for registration, it may also initiate a secure communication channel between itself and the bank or merchant if the need for secure transmission arises.

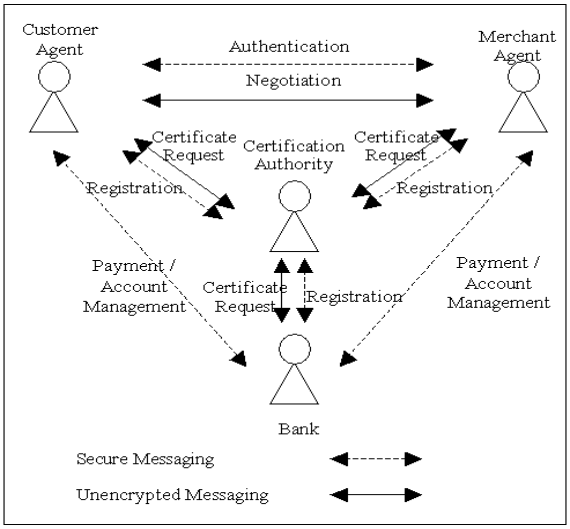

Figure 4: Security Service Architecture

3.3 Authentication Service Design

The authentication service is modelled as an agent; it behaves as a trusted third party for all agents. This service is better known as the CA in short for Certificate Authority in the security and e-commerce community. In our simple authentication model we deploy one CA per service domain (each CA would focus in providing credentials services for specific domains) that is trusted by all agents. It supports three key functions:

- Credentials registration
- Credentials verification
- Credentials request.

Our simple CA does not currently deal with credential revocation and credential leasing. We use standard credentials such as X.509 certificates. The assumptions we have made for this service are:

- All agents starts by owning the credentials of the CA, i.e. the CA credentials are distributed to each agent out-of-bounds;
- The CA is always treated as a trusted third party;
- Agents use a local non-agent service to generate their own credentials. The agent then sends these to the CA for central registration.

There are two main options for credential generation: distributed generation or central generation (in this model we support option 1).

1. When an agent is registered on a particular platform, it will need to generate its private and public keys and embed the certificate of the CA onto its trust store repository. After doing so, the agent may search for authentication services provided by the CA via service discovery methods within the agent shell. The agent would then contact the CA to register its key before granted a certificate. The CA stores the public key of the agent in its key management store or repository and following which provides future authentication service by distributing agents' keys when requested. The security bootstrap for the local key generation and certificate storage on the local trust repository will be discussed in later parts of the document.

2. A second solution to the registration process does not require the agent to generate its own keys, but it sends a request to the CA for a key pair. The CA would then generate the private and public keys on behalf of the agent. The private-key will need to be distributed to the agent using a secure channel.

During the modelling of the service, some considerations were made between making the CA as an agent or a web service. Hence, the decision of modelling the CA as an agent was driven by the following reasons:

- To provide key management within the agent environment
- Facilitate registration and distribution of certificates automatically within a single environment rather than having user registration with the CA via the web
- Provide a root of trust in the agent space

3.4 Secure Channel

The secure channel provides message confidentiality against message eavesdropping and message integrity where transmitted information isn't tampered. The secure channel uses a symmetric key or also known as a session key for faster processing of encrypted message at each end of a two-way communication configuration.

The secure channel can either be delegated to the transport service performing transport level encryption, e.g. SSL or performing encryption at the application level where information is encrypted before transmission. In the design of our system, we included secure tunnelling after the process of authentication to provide efficient end-to-end transmission between the bank and customer agents. The channel is constructed using a session key generated at the conversation initiator's side. The key is exchanged using a mix of public and private keys of the two parties to verify integrity and validity of the key.

During the secure session, the customer agent may query the bank service for the transfer of funds, account description, account creation, and account deletion. Apart from e-Banking services, the secure channel is also scalable and dynamically applica-

ble to either generic or specific services in connection with secure agent-to-agent communication. In some scenario, the secure channel can also be initiated without the authentication process if two communicating agents have the certificate of either side, and finally any initiating party using an “end-session” request can end the secure conversation.

3.5 Use Cases

In this section, we will use Hierarchical Task Analysis (HTA) [5] to break our tasks into sub tasks for expressing the relationships and high-level view of the security service. Our efforts of expressing the service in a HTA diagram would support comprehensible understanding of the system as a whole and aid in creating use cases for the system in later parts of this section. The following are level 1 HTA diagrams expressing the service and please take note that Figure 6 is decomposed from one main task in Figure 5 as noted:

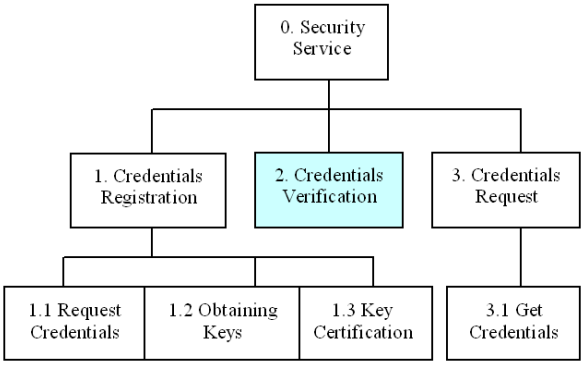

Figure 5: HTA Diagram of Security Service

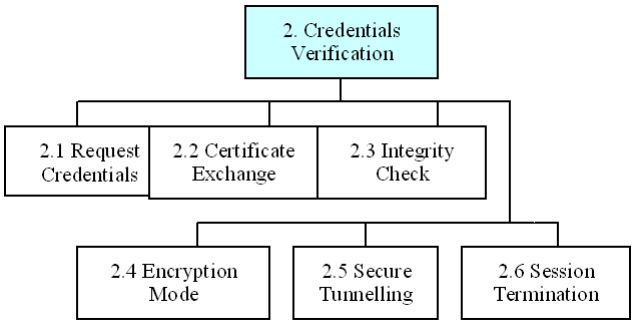

Figure 6: HTA Diagram of Credentials Verification

Task 0: Security Service
Task 1: Credentials Registration
Plan 1:
1.1.Agent wishes to obtain a recognised certificate, (register with the system)
1.2 CA authenticates agents
1.3 CA receives agent's public key and information
1.4 CA certifies key and returns it to the agent
Task 2: Credentials Verification
Plan 2:
2.1 Agent request for authentication
2.2 Agent authenticates each other
2.3 Agent's certificate is checked for integrity
2.4 Agent request for encryption mode
2.5 Secure Session is made between agents
2.6 Agent requesting for secure session termination
Task 3: Credentials Request
Plan 3:
3.1 Agent request for certificate of another agent from CA

Figure 7: HTA Task Model with Plans

Use cases for the security service can be sub-divided into three sections: *registration, verification* and *credential request* use cases:

1. **Credential Registration Use Case:** The registration use cases are mostly derived from inputs received from the users to perform a key registration process. On the other hand, the certificate request use case provides interactions between an agent and a CA during the request for a particular agent's certificate.
2. **Credential Verification (Authentication) Use Case:** As for the *authentication and secure tunnelling* use cases, it features more about the post bootstrapping process of the security support for credentials registration. The initial bootstrap requires all agents to register themselves with the CA based on the assumptions in Section 3.3 and the authentication process during an agent-to-agent communication.
3. **Credential Request:** In this section, we offer a description and use case of the process involved when acquiring a certificate from the CA by agents.

4. Implementation and Results

Our modular agent security software is implemented as a hierarchy of packages. Each package encapsulates the functionality relevant to a specific area. These packages have been developed as agent platform independent entities; figure 8 presents the package hierarchy of the system:

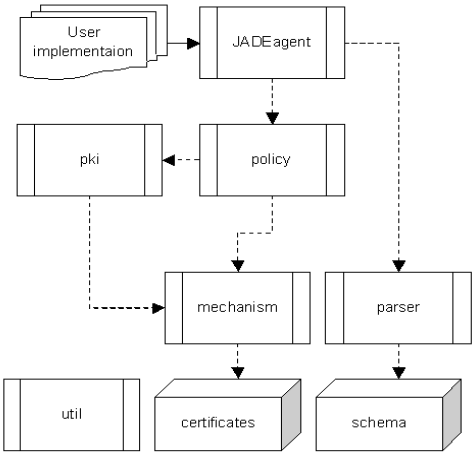

Figure 8: Package Hierarchy

The Agent can get an access to secure services via implementing a specific abstract class (JADEagent), which extends an interface to these secure packages. The principle behind this structure is to allow agents to achieve secure communication without knowing the details of the implementation itself. Additionally, we have also developed a security ontological syntax for key information and key registration protocol. The Security Ontology aims at standardising the interface between PKI services by using standard messages that are formatted according to a specified schema definition. The application has been tested using JADE and has been deployed for live trials and demo[2] over the Agentcities network. The live trials and demo have successfully demonstrated secure communication between agents at the ACL level and its simplicity of deploying security enhanced agents using our model.

5. Conclusions

The introduction of a security service model provides a significant contribution towards future electronic commerce research and development within the Agentcities network and provides input into the FIPA agent interoperability specification process.

6. Acknowledgements

The research described in this paper is partly supported by the EU Agentcities.RTD (IST-2000-28385) project and the EU CRUMPET (IST-1999-20147) project. The opinions expressed in this paper are those of the authors and do not necessarily reflect

[2] http://www.agentcities.org/EURTD/DemoZoneI/index.php?target=demo_services

those of the project partners. The agent security software development was also supported in part with funding from Motorola.

7. References

[1] M. Calisti, D. Deluca, A. Ladd, An Agent-Based Framework for Financial Transactions. Autonomous Agents 2001 Workshop on Agent-Based Approaches to B2B, May 2001, Montreal, Canada.

[2] FIPA Communicative Act Library Specification. Foundation for Intelligent Physical Agents, 2000. http://www.fipa.org/specs/fipa00037/

[3] FIPA SL Content Language Specification. Foundation for Intelligent Physical Agents, 2000. http://www.fipa.org/specs/fipa00003/

[4] Seels, B. and Glasgow, Z. (1990). Exercises in instructional design. Columbus, OH: Prentice Merrill.

[5] SEMPER, Secure Electronic Marketplace for Europe, ACTS Project AC026, http://www.semper.org/

[6] Q. He and K. Sycara ACM, Towards A Secure Agent Society, AA'98 Workshop on "DECEPTION, FRAUD and TRUST in AGENT SOCIETIES", 1998

[7] Rober S. Gray, A flexible and secure mobile-agent system, Thesis, Dartmouth College, Hanover, New Hampshire, 1997

[8] Lalana Kagal, Jeffrey Undercoffer, Filip Perich, Anupam Joshi, and Tim Finin, A Security Architecture Based on Trust Management for Pervasive Computing Systems, Paper, Grace Hopper Celebration of Women in Computing, 2002.

[9] Lalana Kagal, Timothy Finin, and Yun Peng, A Framework for Distributed Trust, Paper, Workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents, IJCAI-2001, Seattle, August, 2001

[10] Frederick B. Cohen, Protection and Security on the Information Superhighway, Wiley, 1995.

[11] FIPA Abstract Architecture Specification. Foundation for Intelligent Physical Agents, 2000. http://www.fipa.org/specs/fipa00001/

[12] Juan Jim Tan, Stefan Poslad, Open Service Vision of Agentcities. 25th German Conference on Artificial Intelligence (KI 2002) Workshop on Multi Agent Interoperability (MAI), September 2002, Aachen, Germany.

[13] S. Poslad, M. Calisti - Towards improved trust and security in FIPA agent platforms published at AA2000, Barcelona, Spain, June 2000.