
SECURITY CONCERNS IN EGOVERNMENT AGENT-BASED INTEROPERABILITY

Fábio Marques¹², Gonçalo Paiva Dias¹³, André Zúquete²⁴

Abstract – The use of Information and Communication Technology in the delivery of public services has a number of advantages. In a first approach, it allows productivity and quality improvements in the delivered services and it allows reductions on the running costs. Furthermore, with interoperability between dispersed IT systems, it is possible to deliver enhanced, combined services. The security on eGovernment Information Systems is pivotal, even more when we are talking about security in interoperability eGovernment architectures. This paper presents a brief description of the functionalities of software agents and their important role for building flexible, agent-based interoperability architectures. Our contribution is the identification of new security issues inherent to agent-based interoperability architectures, some possible solutions to those issues and remaining open questions for future work.

1. Introduction

E-Government is defined by the Organization for Economic Co-operation and Development (OECD) as the use of Information and Communication Technologies (ICT) on government activities [1]. Nevertheless, there are other definitions; a short compilation can be found in [2]. In spite of the multitude of eGovernment definitions that coexist, all have a common baseline: the use of ICT. This means that ICT affects the way we interact with governments and the way governments interact with us.

The use of ICT in the delivery of services brought a number of advantages to the Public Administration. This is noted in the improvement of the quality of services providing low-latency service delivery around the clock, in the reduction of exploitation costs, in the increase of productivity. On the other hand, it also brought a set of concerns, mostly security related. Just to mention a few: privacy of personal, sensitive information; and trust, relatively to the information given by the systems and/or on the systems themselves.

1.1 Interoperability frameworks

The need to exchange information, either between government branches or to deliver services in a cooperative way, led to the creation of interoperability frameworks. These frameworks

¹ Escola Superior de Tecnologia e Gestão de Águeda, Universidade de Aveiro, 3750 – 127, Águeda, Rua Comandante Pinho e Freitas, nº28, {fabio, gpd}@ua.pt

² Instituto de Engenharia Electrónica e Telemática de Aveiro, 3810 – 193, Aveiro, Campus Universitário de Santiago, {fabio, andre.zuquete}@ua.pt

³ Unidade de Investigação em Governança, Competividade e Políticas Públicas, 3810 – 193, Aveiro, Campus Universitário de Santiago, gpd@ua.pt

⁴ Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro, 3810 – 193, Aveiro, Campus Universitário de Santiago, andre.zuquete@ua.pt

act as guides in the design and implementation of interoperability architectures, and in the design of new interoperability-aware information systems. Interoperability frameworks define standards, guidelines and rules. The Federal Enterprise Architecture Framework (FEAF) [3] defines nation-wide, mandatory interoperability rules in the United States of America. In Europe, a set of interoperability frameworks coexist. The European Interoperability Framework (EIF) [4] acts as a guide to achieve interoperability across the systems of the European Union countries and as a complement to the national frameworks. The electronic Government Interoperability Framework (eGIF) [5], in UK, and the Standards and Architectures for eGovernment Applications (SAGA) [6], in Germany, are two examples of national interoperability frameworks.

The advantages of using interoperability architectures as a way to deliver combined eGovernment services are far greater than its disadvantages. They may be used as a catalyst to rethink processes, therefore enabling a more efficient and effective service delivery. The exchange of information between government branches becomes faster, which reduces service delivery time, increases productivity and reduces costs. The clients would benefit from the enhanced service delivery: they can request services anytime, anywhere and through a single front-end, thus avoiding the endless searches and navigations throughout government branches. With interoperability, information can be aggregated by services, not by clients (citizens or businesses).

Several interoperability architectures were proposed over the years. To mention a few: Online Services Computer Interface (OSCI) [7]; Dias and Rafael (D&R) [8]; Secure Electronic Contracts (SeCO) Container [9]; Web Digital Government (WebDG) [10]; Common Services Framework (CSF) [11]; and Access-eGov [12]. These architectures use different strategies to achieve the same goal: public service delivery to clients (users, businesses, public institutions, etcetera) in a transparent way, any time and from any place. The approaches to the problem are also diverse: some privilege the Web services approach; others propose the use of a structured message approach.

1.2 Software agents

There are several types of software agents: their classification depends on their use. Their use is all-embracing, as they can be used for operating systems' interfaces, processing satellite imaging data, electricity distribution management, air-traffic control, business process management, electronic commerce and computer games [13].

An agent is characterized as being autonomous (free from external control), being able to communicate with other agents and being reactive and proactive [13]. Over the years, there were attempts to add new characteristics to this base characterization of an agent. A few examples of these characteristics are: the ability to learn, mobility, the ability to be benevolent, the ability to be honest and to be rational. An even stronger characterization is achieved by adding mental capacities to agents, as: beliefs, desires, and motivations [13]. In [14], Genesereth and Ketchpel, define a software agent as a software component that exchange messages with other software agents using a common language. Based on [13], we define *software agent* as a software component that has autonomy, is reactive and proactive and is able to communicate with other software agents.

In [14], Genesereth and Ketchpel state that one of the goals in the genesis of agent-based engineering was to facilitate the communication between constantly evolving, heterogeneous systems. This single justification, in the context of the Public Administration, would be strong

enough to consider software agents as the natural choice for building interoperability architectures for e-Government. Nevertheless, our choice of software agents is further supported in the basic characteristics of an agent:

- Autonomy – upon receiving a message (request for a service, a service delivery report, etc.), the agent knows how to deal with it without outside intervention;
- Reactive – upon reception of a message, it will promptly respond in an adequate manner;
- Proactive – a constant search for services and their availability is required to achieve a better performance;
- Communication between agents – a critical attribute to achieve interoperability.

1.3 Security issues in eGovernment

Security is one of the most important aspects of eGovernment systems. Typical security issues encompass trust, authentication and access control.

End customers, people and companies, want to be assured that their sensitive data is not accessed by unauthorized entities and that it is not used to other purposes besides those established by law or by specific contracts. Furthermore, customers require a trust relationship with the entity that delivers the service. Otherwise, all information obtained in the course of a service delivery would be useless. Finally, eGovernment services need to know to whom (customer) they are delivering the service, determine if the customer has access the service or the consequent information provided, and ensured that services will be paid by costumers if required.

All these security issues show how security is a mandatory concern in eGovernment services, both to clients and governments, and that it must be taken in consideration since the beginning of the specification of their ICT.

1.4 Contribution

As we have seen, interoperability is essential by many reasons and the use of software agents a natural choice to address the problem. Security is a main concern and, despite the security solutions for the generic security requirements that an interoperability architecture must respect, there are particular security concerns that depend of the used technology, such as software agents.

The objective of this paper is to identify specific security issues inherent to the usage of software agents for eGovernment interoperability, and to point out some of the possible solutions to those concerns. To achieve this objective, we briefly present service delivery processes, the main functionalities of software agents and their role in interoperability architectures.

This paper is structured as follows. In section 2 we present the general security requirements of an interoperability architecture. In Section 3 we present the security concerns inherent to agent-based interoperability architectures. In Section 4 we point out some possible solutions to those security concerns. Finally, in section 5 we conclude the paper and present some future work.

2. Generic Security Concerns

Security is one of the issues that can determine the success or failure of interoperability architectures in e-Government [8]. From the client point of view, this is mostly materialized in the fear of non authorized users being able to access their sensitive/private information. From the government point of view, the security is essential at many levels, ranging from knowing to whom they are doing business, to assuring that the client will pay the fees for the requested service.

In [8], the authors established different sets of security requirements for e-Government infrastructures. Although there are differences, the core security requirements are basically the same. These core security requirements are also mentioned in EIF [4] and FEAF [3]. In Table 1, we present a short compilation of the security requirements mentioned in the previous referred publications.

	Caloyannides et al.	Joshi et al.	Arcieri et al.	Dias and Rafael
Authentication	x		x	x
Authorization	x		x	x
Confidentiality		x	x	x
Integrity	x	x	x	x
Accountability/Traceability		x	x	
Non-Repudiation	x			x
Availability		x		x
Information Assurance		x		

Table 1: Requirements compilation table

From Table 1, we settle the following set of security requirements for eGovernment interoperability architectures:

- Authentication – as the ability to identify the system actors (users, services, etc.);
- Authorization – as the ability to give permissions to system actors to system resources;
- Confidentiality – as being capable to prevent the unauthorized access to information by eavesdroppers or non-authorized users;
- Integrity – as the capability to prevent, or at least, to detect corruption or tampering of the information;
- Traceability – as the ability to trace the actions to the actor who performed them;
- Non-Repudiation – as being capable to prevent the intervenients on a transaction from denying their action.

Information assurance and availability were not included. Information assurance was not considered since it is somehow included in the fulfillment of some of the other security requirements: Authorization, Confidentiality and Integrity. Availability requirement is an implementation concerned requirement, which is not aborded on this paper.

Several interoperability architectures for eGovernment have been presented over the years. OSCI, SeCO Container, WebDG, CSF, Access-eGov and D&R are some examples. We believe that the above architectures represent the technologies and methodologies used by the majority of the eGovernment interoperability architectures existing nowadays. The majority of the above architectures use a Public Key Cryptography (PKI), the only exception is WebDG. WebDG is the architecture that fulfills fewer requirements: Authentication and Authorization.

The only architectures that fulfill all security requirements are D&R and Access-eGov. Both use similar approaches to most of the requirements, the only exception is the Authorization requirement: D&R adopted a solution granting and denying access to resources using the information associated to the user or entity, in the case of Access-eGov this is done through the association of an attribute-based access control and XACML⁵ to each service.

The authentication requirement is generally fulfilled by the use of a PKI. As mentioned before WebDG is the exception. The authorization requirement has different approaches: WebDG gives the control to the user through the use of privacy profiles; the SeCO Container uses a PKI and the information contained on the structured message; the solutions of Access-eGov and D&R were already mentioned, D&R associates the authorization attributes to the service while Access-eGov associates the authorization attributes to the user or entity. OSCI does not treat this requirement and, in the used sources, there is no information on how this requirement is treated by CSF. The confidentiality requirement is fulfilled by OSCI, Access-eGov and D&R by an encryption based methodology. OSCI, SeCO, Access-eGov and D&R use digitally signed content to fulfill the integrity requirement. The traceability requirement is fulfilled by built-in solutions in all architectures, except in WebDG where is not treated. The non-repudiation is fulfilled in a standard way, by the use of digital signature, in all architectures besides WebDG. Despite verifying all requirements, Access-eGov and D&R have some limitations. On the used sources for Access-eGov there is no references to how the authorization is managed on combined service delivery, leaving it as an open question. D&R, for example, delegates user authentication to external entities.

3. Agent-based security concerns

As mentioned previously, we believe that software agents are the natural choice to base eGovernment interoperability architectures. This does not mean that other technologies should be put aside. For example, the usage of Web Services or Semantic Web Services can support the development of agent-base architectures. Other technologies may also be use to complement or expand the architecture functionalities.

The majority of public services offered by the Public Administration can be composed of simpler services delivered by specific branches of the Administration. We call the process of delivering a service (simple or composed) as a workflow. Therefore, to deliver an enhanced, combined service, the agent-based interoperability architecture must search the systems that provide all the necessary simpler services that compose it, until creating the workflow of the combined service (the one that is being requested). In other words, the architecture must find all the necessary workflows that compose the workflow of the combined service. This can be accomplished in two ways:

Static delivery – before starting the delivery of the service, the software agent to whom the service is requested decomposes the original request into simpler request, and searches for the agents that can deliver these services. Thereafter, the software agent composes a workflow and initiates the process of delivering the service (running the workflow);

Dynamic delivery – in this case the workflow construction takes place in run-time. This means that each software agent must choose the adequate software agent to proceed

⁵ http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

with the next step of the delivery process, without knowing the full extent of the complete workflow.

Messages are swapped between software agents on both types of service delivery. Although the functionality issues of the software agents are not an objective of this paper, it is important to address their basic functions to be able to point out the security issues that must be solved. Thus, a software agent must be able to: (a) receive a service request; (b) make the analysis of the request; (c) if necessary, decompose the request; (d) if able, deliver the service or, at least, part of the composed service; (e) identify the agents that can process the remaining parts of the service and forward sub-requests to them; (f) compose a service delivery from the answers it receives from those agents; (g) and certify the service it delivers.

The main difference of both types of service delivery, is the fact that in the case of static deliver the workflow path is constructed when the service is requested, while in the case of the dynamic delivery, this path is constructed in run-time. There are pros and cons in both cases. The most evident are performance and security related. In the case of the static delivery, the main advantage comes from the fact that all the software agents (public entities) are known from the start. This is an important advantage regarding security assurance. The main disadvantage is performance related, since the creation of the workflow path is in general a process that consumes a great deal of resources. In the case of dynamic delivery, the advantages and disadvantages reverse. While the resource consumption is divided by each of the software agents involved on the workflow path, the related security issues became harder to solve.

Given these functionalities, the following security issues must be solved:

- Confidence, or trust, in agents – Is a software agent competent to deliver the service that is publicized? Is it capable to choose wisely and securely the next agent on the workflow path (dynamic delivery) or to choose the agents that will composed the workflow path (static delivery)?
- Privacy and integrity preservation – the content of the messages that are being exchanged are, usually, of sensitive nature. Can the agent access the information? If it should not, how can it be prevented? How can it prevent the access of other agents to the information generated by itself and that should not be accessed by others?
- Confidentiality – How can the software agent assure that only the intended software agent can access a piece of information, when it may not know who it is (dynamic delivery)?
- User proxying – to deliver a service there are occasions where it is necessary to retrieve more information from other entities. This is done by soliciting that information to the competent public entity. What will be the privileges to access that information, the ones from the user requesting the service (the software agent acts as a proxy) or the ones from the software agent requesting the service? Is the information solely addressed to that software agent or user?

Although these security concerns are in a way embedded in the generic security requirements mentioned earlier, they have a direct impact in the security measures that have to be applied to agent-based interoperability architectures. There are other security concerns related to the implementation of software agents:

- Unknown agent behavior – agents may be very complex, thus making the testing incredibly hard. Although there exist techniques to minimize the problem, software agents may behave in a way that is not expected or even desirable.

- Agent compromise – software agents are key elements in the architecture structure, turning them into desirable attacking points to get access to the system. The risk of accessing or delivering a request to a compromised agent is real, however there are solutions that can minimize this risk.

On the next section we will mention some possible solutions to some of these questions.

4. Some possible solutions

Some of the security issues identified in the previous section have, already, some well-known, *de facto* standard solutions.

The trust on an entity, either a person or organization, is generally assured with the help of mutually trusted third party (TTP) entities. This is put in practice in the real world, e.g. a bank can act as a TTP in a business transaction between a person and a company. This TTP acts as a trust mediator, his job is to ensure that the remaining peers with no *a priori* trust relationships among themselves are trustworthy. In this specific case, a possible solution should be the insertion of a certificate authority (CA) or a set of CAs that could work together with the objective of building trust relationships between service delivery agents. The certificate issued by the CA should be added to the messages containing the information requested.

The privacy and integrity preservation problem can be solved by using a PKI. The integrity issue can be solved by digitally signing information, in the case of a person and, in the case of an entity, through the encryption of the hash of the information by the software agent. The privacy issue may be resolved in a similar manner, using the public key of the destination agent or person to assure that they are the only ones who can read and understand the information.

There are times when there is a need to request information to complete a request. Sometimes the information must be requested by another software agent, on behalf of the original requester. This raises the following question: What are the privileges that should be used to access that information? This question has no simple answer. There are some points that must be taken into consideration, such as: who is the addressee of the information? or, is it solely addressed to that entity?

There are situations when a certain kind of information must be gathered to be delivered somewhere in the workflow path. If the addressee is known this is not a problem (static delivery). However, in the dynamic delivery the workflow path is being created while the service is being fulfilled. This means that the software agent that will perform the service is decided by its predecessor, and is not known before that.

5. Conclusions and Future Work

In this paper we presented a set of general security requirements of agent-based, interoperability architectures for eGovernment. Besides the security concerns naturally rose by eGovernment services and by interoperation among eGovernment services, the use of software agents brings some additional security concerns: agent trust, privacy and integrity preservation, confidentiality, and user personification. These additional concerns derive from the specificity of software agents. In section 4 we presented some possible solutions to these concerns. The main technical solution to most of the issues is the use of a Public Key Infrastructure. It can

be used to create trust chains, and to assure data privacy and integrity. Although some solutions were presented, there are still some open issues remaining. Our future work will be towards finding solutions for the open issues, to implement a prototype and to validate our solution.

References

- [1] Public Management Service, E-Government: Analysis framework and methodology, December 2001. <[http://www.oelis.oecd.org/olis/2001doc.nsf/c5ce8ffa41835d64c125685d005300b0/0b677ed527d35bc0c1256b21004f4b6a/\\$FILE/JT00118445.PDF](http://www.oelis.oecd.org/olis/2001doc.nsf/c5ce8ffa41835d64c125685d005300b0/0b677ed527d35bc0c1256b21004f4b6a/$FILE/JT00118445.PDF)> (03/03/2009).
- [2] C. Codagnone and M. Wimmer (Eds), Roadmapping eGovernment Research – Visions and Measures towards Innovative Governments in 2020, My Print snc di Guerinoni Marco & C, Clusone, 2007. <<http://www.egovrtd2020.org/>> (14/02/2008).
- [3] The Chief Information Officers Council, Federal Enterprise Architecture Framework, version 1.1, September 1999. <<http://www.cio.gov/Documents/fedarch1.pdf>> (09/05/2008).
- [4] European Commission, European Interoperability Framework for Pan-European eGovernment Services, version 1.0, 2004. <<http://europa.eu.int/idabc/servlets/Doc?id=19529>> (10/05/2008).
- [5] Cabinet Office – e-Government Unit, e-Government Interoperability Framework, version 6.1, March 2005. <http://www.govtalk.gov.uk/schemasstandards/egif_document.asp?docnum=949> (14/05/2008).
- [6] KBSt at the Federal Ministry of the Interior, Standards and Architectures for eGovernment Applications, Version 3.0, October 2006. <http://www.cio.bund.de/cIn_093/sid_F20C96E5E0C318B4532F22C4D7350535/DE/Standards/SAGA/saga_node.html> (10/08/2008).
- [7] F. Steimke, M. Hagen, OSCI: a common communications standard for e-government, in: Proc. EGOV'03, Lecture Notes in Computer Science, vol. 2739, Springer, Prague, Czech Republic, September 1-5, 2003, pp.250-255.
- [8] Gonçalo Paiva Dias and José Alberto Rafael, A simple model and a distributed architecture for realizing one-stop e-government, Electronic Commerce Research and Applications 6 (1) (2007) 81-90.
- [9] M. Greunz et al., Supporting market transaction through XML contracting containers, in: Proc. Of the Sixth Americas Conference on Information Systems (AMCISS'00), Long Beach, CA, August 10-13, 2000.
- [10] A. Bouguettaya et al., WebDG – A platform for e-government web services, in: Proc. ER (Workshops), Lecture Notes in Computer Science, vol. 3289, Springer, 2004, pp. 553-565.
- [11] UMIC – Agência para a Sociedade do Conhecimento, Plataforma de Interoperabilidade. <http://www.unic.pt/index.php?option=com_content&task=view&id=2687&Itemid=112> (25/07/2008).
- [12] J. Kolter et al., An architecture integrating semantic e-government services, in: Communications Proceedings of the 5th International eGovernment Conference (EGOV'06), Krakow, Trauner Druck, 2006.
- [13] Michael Luck et al., Agent-Based Software Development. Artech House Publishers, 2004.
- [14] Michael R. Genesereth and Steven P. Ketchpel, Software Agents, in: Communications of the ACM, 37:48-53, 1994.
- [15] M. Caloyannides et al., US e-government authentication framework and programs, IT Professional 5 (3) (2003) 16-21.
- [16] J. Joshi et al., Digital government security infrastructure design challenges, Computer 34 (2) (2001) 66-72.
- [17] F. Arcieri et al., A layered IT Infrastructure for secure interoperability in Personal Data Registry digital government services, in: Proc. Of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications (RIDE'04), IEEE, New York, 2004, pp. 95-102.