Service-Oriented Trust and Reputation Architecture [⋆]

Francisco Moyano

Department of Computer Science, University of Malaga, 29071, Málaga, Spain
{moyano,mcgago,jlm}@lcc.uma.es

Abstract. As the Future Internet arrives, more complex, service-based applications are spreading. These applications pose several challenges, including the huge amount of entities that must interact and their heterogeneity. The success of these applications depends on the collaboration and communication of these entities, that might belong to different organizations and administrative domains. Therefore, trust and reputation become two crucial issues. We propose the specification and design of a service-based security architecture that stresses the delivery of trust and reputation services to any application that might require them.

Supervisors: Carmen Fernandez-Gago and Javier Lopez

1 Introduction: Problem and Motivation

The context that frames this work is that of Service-Oriented Architectures, which is described in Section 1.1. The problem that we mean to address and the motivation are presented in Section 1.2.

1.1 A Brief Introduction to Service-Oriented Architectures

A rather complete definition of Service-Oriented Architecture (SOA) is "the policies, practices, frameworks that enable application functionality to be provided and consumed as sets of services published at a granularity relevant to the service consumer. Services can be invoked, published and discovered, and are abstracted away from the implementation using a single, standards-based form of interface" [17]. In general, a service implements a limited and specific functionality, and can be discovered and called by means of standard technologies.

SOA is also often understood as being an architectural style that defines several components with their relations and constraints from which many different

[⋆] The research leading to these results have received funding from the European Community's Seventh Framework Programme FP7/2007-2013 as part of the Network of Excellence NESSoS (www.nessos-project.eu) under grant agreement number 256980. The first author is funded by the Spanish Ministry of Education through the National F.P.U. Program.

architectures can be derived. However, architects of service-oriented systems often find conflicts when trying to reconcile the business goals, the non-functional requirements, and some principles that are a consequence of the SOA approach, such as standardization at multiple levels, loose coupling, reusability, composability, and discoverability [6].

A crucial component for many SOAs is the Enterprise Service Bus (ESB). An ESB is a complex piece of software that mediates between clients and services, and among services. It is an information bus that connects and allows the communication of heterogenous applications (e.g. Java Message Service (JMS) applications and Simple Object Access Protocol (SOAP) applications). It also caters for other commonly needed services, such as security, protocol conversion or exception handling.

1.2 Trust and Reputation in Service-Oriented Architectures for Future Internet Applications

SOA systems have been traditionally secured at the endpoints. This means that a service is hard-coded with security functions that realize security services. This approach has several shortcomings. One of them is the business and security coupling, which in turn makes manageability and interoperability more difficult. Reusability is also undermined, since although the service might fulfil a business goal interesting for the organization, the coupling with security would make it useless for the organization if the security policy of the service does not match that of the organization.

The Future Internet entails the arrival of new, complex service-based applications that span across multiple boundaries and administrative domains. This calls for trust and reputation services that assure the trustworthiness of the entities that take part of such heterogenous communications and collaborations, and which aid the decision-making processes. Examples of applications are those described in NESSoS [2]. These applications require the interaction of multiple entities and the management of personal and private information, therefore they are security-critical.

Trust and reputation services might assist traditional security services, such as encryption and authorization. The latter mechanisms can provide a trusted environment, where communication between service providers and clients is protected, and where access to business assets is limited. These mechanisms, if well leveraged, might provide certain degree of trust between clients and customers in general. However, the notions of trust and reputation focus on specific relationships between a given provider and a client, taking into account both local information (e.g. interaction records), and external information (e.g. recommendations from other clients or providers). Thus, traditional security mechanisms should provide the basis, the trusted medium onto which to build trust and reputation solutions.

Although there are some proposals towards the delivery of security as services, to the best of our knowledge none of them cope with trust and reputation services. Hafner et al. [10] propose an Enterprise Service Bus (ESB)-

based message-oriented Security as a Service architecture. Aurel [4] proposes SOSA (Service-Oriented Security Architecture), a security architecture for distributed web applications that mediates between a service requester and a service provider. WS-* security standards, such as WS-Security, WS-Trust, WS-Policy or WS-SecureConversation deal with confidentiality, integrity, non-repudation, and policy specification.

2 Aim: Trust and Reputation as a Service

As we mentioned in Section 1.2, the arrival of Future Internet demands trust and reputation solutions. In the complex, heterogeneous scenarios that arise in this context, trust mechanims must be leveraged since they play a crucial role in decision-making processes.

Trust is a complex concept for which a clear, standard definition has not been provided yet. A possible definition is the level of confidence that an entity of a system places on another entity of the same system for performing a given task. Thus, two features of trust are uncertainty and subjectivity. Reputation, on the other hand, is a more objective concept, and is based on information about or observations of the past behaviour of an entity. Both concepts are very related, and in fact, reputation can be used as a means to determine whether an entity can trust another entity [12].

Our aim is the specification and design of a functional, reusable service-based security architecture that emphasizes the delivery of trust and reputation services.

A key feature of this architecture must be its reusability, since it should serve as a framework for building multiple applications that require trust and reputation services, as it is the case of Future Internet applications. It is also important to note that we intend to analyze the relationship between trust, reputation, and other security services. The architecture might therefore deliver those security services that have an influence on the overall trust of any application.

In the literature there are important contributions in the field of trust and reputation that might assist our work.

Kiefhaber et al. [16] propose the Trust-Enabling Middleware, which provides generic funtionality for applications running on top of it that need to save, interpret, and query trust related information. However, the authors are oblivious of other security mechanisms that might have an important impact on the overall trust of the system.

It is also important for a reusable architecture to offer flexible mechanisms to accommodate or compose different trust models according to the needs of the application. In this direction, Huynh [7] proposed his Personalized Trust Framework (PTF), a rule-based system that makes use of semantic technologies for, given a domain, to apply the most suitable trust model. In a similar direction, Suryanarayana et al. [18] present PACE (Practical Architectural approach for Composing Egocentric trust), an architectural style for composing different trust models into the architecture of a decentralized peer in a P2P architecture.

3 Research Methodology

The methodology that we intend to follow to conduct the research consists of the following phases:

1. Exploratory phase: during this phase, a wide research will be done in order to gain a solid knowledge about the process and tools for architecting a service-oriented architecture.
2. Construction phase: the specification and design of the architecture will be done during this phase. In order to accomplish these tasks, we will accommodate the output of the previous phase, which provides us with a set of tools, methods and processes that currently exist to architect service-oriented solutions. This output could contain as well some found gaps that we will need to bridge during this phase.
3. Validation phase: NESSoS scenarios represent some of the most important Future Internet applications. Thus, the architecture will be validated for, at least, one of these scenarios, which are within the scope of e-Health or Smartgrids.

At the moment, we are at the exploratory phase, where we are surveying existing approaches towards the building of a service-based security architecture. Some of our findings are briefly described in the next section.

3.1 Processes and Tools for Architecting SOA Solutions

In general, the process for building any architecture is iterative and consists of several steps [9]. The first step is requirements elicitation, where the functional and non-functional requirements are identified and documented from the scenarios and the stakeholders. Next it is important to identify those requirements that drive the architecture building, namely the quality attributes (e.g. security, modifiability or performance) and constraints (e.g. reusability of legacy systems). These architectural requirements are prioritized. Trade-offs analysis might be required since requirements are often in conflict between each other. After choosing the set of architectural requirements, the architecture design takes place, when choosing a set of interrelated design patterns can help the architect. Finally, the architecture is validated. This validation consists of using scenarios to check by hand how the architecture responds to different stimulus. Another validation technique is early prototyping. In either ways, if a flaw is discovered, all the previous phases should be revised, leading to another iteration.

This generic process is also applicable to the case of SOA. Yet there are some more specific SOA-oriented processes, as the Rational Unified Process for Service-Oriented Modeling and Architecture (RUP SOMA) proposed by IBM [20], the Service-Oriented Modeling Framework, proposed by Michael Bell [5], and the SOA/TOGAF standard by The Open Group [3]. We will survey these processes to find one that better adapts to our needs.

The modeling of a service-oriented architecture requires designing many aspects, ranging from the business activities to the non service-oriented, existing assets. However, the most interesting for us is the service model, where both the atomic and composite services are defined in terms of their specifications, interfaces, inputs, outputs, and collaborations. As with traditional object or component-based applications, services can be graphically modeled with the Unified Modelling Language (UML). Several UML profiles exist in the literature that allow modeling service-oriented architectures, such as UML4SOA [13] from the SENSORIA project [1], UPSS (UML Profile for Software Services) [11] adopted by IBM, and OMG's Service oriented architecture Modeling Language (SoaML) [15].

For the implementation and deployment of services, there are several alternatives. One of them is Apache Tuscany [19], which provides a service-oriented architecture infrastructure to develop and run applications in a service-oriented approach. It implements the Service Component Architecture (SCA), an assembly model developed by major vendors for composing heterogenous applications. EclipseSOA [8] is another interesting alternative, since it provides a runtime and tool integration platform that assists in all the required steps for the development of a SOA solution. Regarding the Enterprise Service Bus (ESB), there exist several open source ESB implementations, such as Mule [14].

As in the case of the processes, we will research on these tools in order to choose the one that better fulfils our needs.

4 Conclusions

New Future Internet applications will need support from trust and reputation services for their successful adoption. Whereas traditional security requirements have been provided by architecture middlewares and frameworks, trust and reputation have often been laid aside.

On the other hand, the service-oriented paradigm provides many advantages, including its focus on reusability. We propose the building of a service-oriented architecture, capable of delivering security services in general, and emphasizing the delivery of trust and reputation services in particular. This architecture must be flexible and reusable, constituting a framework that can be used by different applications that might require their services.

The main contributions that we expect to do in the field of Engineering Secure Software and Systems are in different levels. At a process level, we will research on the high-level development processes for building service-oriented solutions. At the architectural level, a research will be conducted regarding the modeling issues for service-based architectures. At a security level, the relationships between security services will be analyzed and made explicit, which in turn might help us to gain a better understanding of how trust is related to more traditional security services. Finally, from an overall perspective, we believe that the process of building a service-based security solution bridges a gap between the service engineering and security research communities.

References

1. SENSORIA - Software Engineering for Service-Oriented Overlay Computers. `http://www.sensoria-ist.eu/`, 2005–2010.
2. Selection and Documentation of the Two Major Application Case Studies. NESSoS Deliverable 11.2, October 2011.
3. Using TOGAF to Define and Govern Service-Oriented Architectures. Technical report, The Open Group, November 2011.
4. Cristian Aurel. *Service Oriented Security Architecture applied to Spatial Data Infrastructures.* PhD thesis, University of the Federal Armed Forces, Munich, January 2008.
5. Michael Bell. *Service-Oriented Modeling (SOA): Service Analysis, Design, and Architecture.* Wiley, 2008.
6. Philip Bianco, Grace A. Lewis, Paulo Merson, and Soumya Simanta. Architecting service-oriented systems. Technical report, Software Engineering Institute, August 2011.
7. Trung Dong Huynh. A Personalized Framework for Trust Assessment. *ACM Symposioum on Applied Computing - Trust, Reputation, Evidence and other Collaboration Know-how Track*, 2:1302–1307, December 2008.
8. Eclipse. EclipseSOA. http://www.eclipse.org/eclipsesoa/.
9. Ian Gorton. *Essential Software Architecture.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
10. Michael Hafner, Mukhtiar Memon, and Ruth Breu. SeAAS - A Reference Architecture for Security Services in SOA. *Journal of Universal Computer Science*, 15:1–21, December 2009.
11. Simon Johnston. UML 2.0 Profile for Software Services. Technical report, IBM, April 2005. `http://www.ibm.com/developerworks/rational/library/05/419_soa/`.
12. Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, March 2007.
13. Philip Mayer, Nora Koch, Andreas Schroeder, and Alexander Knapp. The UML4SOA Profile. Technical report, LMU Muenchen, 2010.
14. MuleSoft. Mule ESB. http://www.mulesoft.org/.
15. OMG. Service oriented architecture Modeling Language (SoaML). Technical report, OMG, April 2009.
16. Florian Siefert Gerrit Anders Theo Ungerer Wolfgang Reif Rolf Kiefhaber. The Trust-Enabling Middleware: Introduction and Application. pages 1–18, March 2011.
17. David Sprott and Lawrence Wilke. Ewerare-cbdi forum service-oriented architecture. `http://msdn.microsoft.com/en-us/library/aa480021.aspx#aj1soa_topic3`, January 2004.
18. Girish Suryanarayana, Mamadou H. Diallo, Justin R. Erenkrantz, and Richard N. Taylor. Architectural Support for Trust Models in Decentralized Applications. In *Proceeding of the 28th international conference*, pages 52–61, New York, USA, 2006. ACM Press.
19. The Apache Software Foundation. Apache Tuscany. http://tuscany.apache.org/home.html.
20. Ueli Wahli, Lee Ackerman, Alessandro Di Bari, Gregory Hodgkinson, Anthony Kesterton, Laura Olson, and Bertrand Portier. Building SOA Solutions Using the Rational SDP. Technical report, IBM, April 2007.