# Towards a Process for Web Services Security

Carlos Gutiérrez[1], Eduardo Fernández-Medina[2] and Mario Piattini[2]

(1) STL, Madrid (SPAIN)

(2) Alarcos Research Group. Universidad de Castilla-La Mancha.
Paseo de la Universidad 4, 13071, Ciudad Real. (SPAIN)

**Abstract.** Web Services (WS) security has been enormously developed by the major organizations and consortiums of the industry during the last few years. This has carried out the appearance of a huge number of WS security standards. This fact has caused that organizations remain reticent about adopting technologies based on this paradigm due to the learning curve necessary to integrate security into their practical deployments. In this paper, we present PWSSec (Process for Web Services Security) that enables the integration of a set of specific security stages into the traditional phases of WS-based systems development. PWSSec defines three stages, WSSecReq, WSSecArch and WSSecTech that facilitate the definition of WS-specific security requirements, the development of a WS-based security architecture and the identification of the WS security standards that the security architecture must articulate to implement the security services, respectively.

## 1 Introduction

The open Internet nature is promoting the development of Web Services (WS) as a paradigm that enables complex business workflow integration scenarios and provides the so-demanded and so-called hyper-connectivity inter- and intra-enterprises [1].

This standard-based quality-centered paradigm is evolving rapidly due to its ability of handling and addressing the heterogeneous information systems integration challenge. In fact, according to the most recent reports from IDC, over the next years, the market for WS-based solutions will grow steadily reaching $11 billion in 2008 [2]. Due to this fact, an enormous quantity of WS standards is being produced. This diversity, also found in the context of WS [3] security has made us consider its application, from a global perspective, as a very complex and hard process to understand with a very difficult learning curve. Following, some questions that are hard to answer for someone who is new to WS-based systems are stated:

- Given a complete set of functional requirements, what WS security standards should I choose?
- What are the WS security requirements that should be taken into account in my WS-based system?
- What WS security standard from those addressing similar aspects should I integrate in my WS-based system?

Farkas C. (2005).
Tree Automata for Schema-level Filtering of XML Associations.
In *Proceedings of the 3rd International Workshop on Security in Information Systems*, pages 298-308
DOI: 10.5220/0002575502980308
Copyright © SciTePress

At present, there is still a lack of a global approach that offers a methodical development to construct security architectures for WS-based systems. For this purpose, the main objective of this paper is to present the process PWSSec (Process for Web Services Security). PWSSec has been created to facilitate and orientate the development of security for WS-based systems in a way that in each one of the traditional stages for the development of this kind of systems [4], a complementary stage comprising security [5] can be integrated. Therefore, this process can be used once the functional architecture of the system has been built or during the stages used to elaborate this architecture. In both cases, the result will be a WS-based security architecture formed by a set of coordinated security mechanisms that use the WS security standards to fulfill the WS-based system security requirements.

In this paper we will provide a brief overview of the complete process and we will present in a more detailed form the reference security architecture defined in the WSSecArch stage.

The rest of the paper is organized as follows. In section 2, the PWSSec process is introduced. In section 3, an in-depth study of the stage related to the specification of the WS-based security architecture is presented and an example is shown. In section 4, related research works are stated, and, finally, in section 5 conclusions and points that need to be developed in the future are enumerated.

## 2 PWSSec - Process for Web Services Security

In this section, we will provide an overview of the process PWSSec.

This process specifies how to define security requirements for WS-based systems, describes a WS reference security architecture that guarantees and demonstrates its development and provides us with facilities to obtain concrete security architectures based on the current WS security standards. In general, the main features of the process presented in this section are as follows:

- Iterative and incremental process: For each iteration that comprises the development of all stages, a part (increase) of system security is analyzed, characterized and developed.
- The two basic principles are process traceability and reusability and product interoperability and reusability. Process reusability will allow us to apply it to different domains in which it is necessary to develop a WS-based security architecture whilst product reusability will guarantee shorter development cycles based on proved solutions. Product interoperability, mainly applied in WSSecArch and WSSecTech stages, will guarantee that WS-based security solutions agreed by the most important industry consortiums will be taken so that systems developed with PWSSec will present a high degree of integration and interoperability.
- Process managed by the elements and basic procedures defined for an Architecture based on WS [6].: the basic actors are the services provider agents, the services consumer agents and the discovery agents whilst the basic processes are publishing, discovery, binding and invocation.
- Based on the concept and techniques developed within the scope of Security Requirement Engineering and Risk analysis and management [7-9].

- Developed from the concept and techniques that allow us to implement security into software architecture [10-12].
-

  Figure 1 shows us the stages in which PWSSec process is structured.

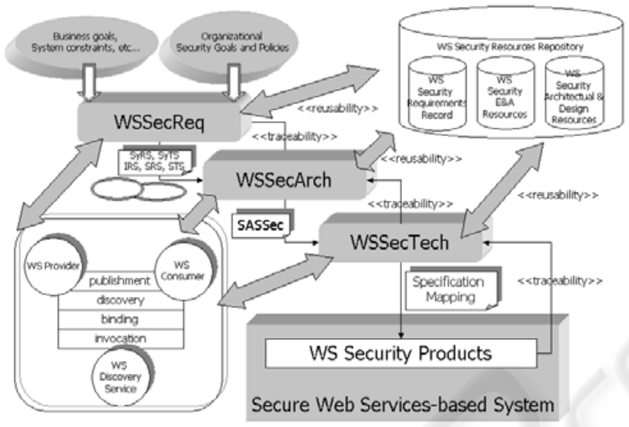

**Fig. 1.** PWSSec process layout.

Each one of the stages defined by PWSSec describes its inputs, outputs, activities, actors and sometimes, guides, good practices, tools and techniques that complement, improve and facilitate the set of activities developed within these stages.

The PWSSec stages depicted in Figure 1 are briefly described as follows:

- WSSecReq (Web Services Security Requirements): The main purpose of this stage is to produce a specification (or a part of it) of the security requirements of the target WS-based system. Its input is composed by a specification of the scope that we want to comprise during iteration (e.g.: if we have a definition of the Use Cases available, we can select those that we want to cover and use them as an input for iteration), the business and security goals defined for the system as well as the part of the organizational security policy that we estimate that can impact on the system design. The output is basically formed by the set of attack scenarios, defined according to [13, 14] and represented according to the UML profile [9], by the set of use cases of security according to Donald G. Firesmith [15] and by a formal specification of the security requirement for the scope of the system based on SIREN [16]. This stage is supported by a repository that contains attack scenarios patterns that are grouped into attack profiles and security use cases, by a set of reusable security requirements templates and by a basic guide for the definition of scenarios and security requirements within systems based on WS.
- WSSecArch (Web Services Security Architecture) has as its main objective to allocate and integrate the security requirements specified in WSSecReq stage through the identification of the appropriate WS-based security architectural patterns and its integration in a WS-based security architecture. We skip a brief description of this stage because a detailed revision is developed in section 3.
- WSSecTech (Web Services Security Technologies): The main purpose of this stage is to identify a set of WS security standards that will implement the security

services identified in the previous stage. Output will be a description of the set of standards identified for each security service together with the reasoning framework that made us select it and a concrete security architecture design. The activities carried out in this stage are the following: i) WS Security Standards Identification; ii) Security Policies Specification to define the Security Policy of the Abstract Security Service Instance that implements certain standard.

## 3 WSSecArch – Web Services Security Architecture

WSSecArch comprises the security-related Architecture Design providing a clear distribution of the security requirements into WS-based security mechanisms. These WS-based security mechanisms help to mitigate those risks identified for every business Web Service in the WSSecReq stage and complete the WS-based security architecture.

The two principles that characterize this stage are the same than in WSSecReq: products reusability and traceability. Architecture security solutions are taken from architectural patterns [17] that have been successfully proved in other projects and belonging to the most diverse domains. Thus, it has been created a repository of WS security architectural patterns related to each one of the security factors, including the reasoning framework that relates and justifies them [18, 19]. Nowadays, there are different WS-based security solutions for the different security requirements that we can identify. Up to this point we have focused our work in federation environments including authentication, Single Sign-On and domain trust federation topics.

The considered inputs in this stage are as follows: i)Business goals of the current iteration; ii) Organizational goals and security policies taken into account during the current iteration; iii) The set of attack and security scenarios developed in WSSecReq; iv) The set of security requirements specifications defined in WSSecReq stage; v) The WS Security Architectural & Design Repository which contains a set of WS-based architectural security patterns that will promote product reusability.

The main output of this stage is the Security Architecture Specification (SASSec in Figure 1) that accomplishes: i) The set of security requirements solved by the architecture; iii) The set of security architectural patterns that implement them documented as in Attribute-Based Architectural Styles [19]; ii) The catalog of security policies associated with the business WS and security WS; iii) A series of views [17, 20] whose type depends on the stakeholders that will read the SASSec. These views will demonstrate how the security architecture integrates with the functional architecture and how the attack scenarios are addressed. Between the security requirements, the identified security patterns and the WSSecKern (Web Services Security Kernel) components (we will get on to this concept later), a forward and backward navigable traceability relationship, justified by the set of design decisions or applied reasoning, will exist.

The main actors in this stage are the Requirements Engineering Team, the Architecture Design Team and the Security Team. The last two participate in the Security Architectural Patterns Identification and Security Architectural Patterns Integration stages. In the Security Architecture Validation stage the three actors participate.

## 3.1 Activities

### Security Architectural Patterns Identification

For each security requirement of each business WS belonging to the current iteration, we must identify the WS security architectural pattern(s) that solves it. This architectural pattern defines a set of abstract services types (since they do not define how they must be implemented in terms of algorithms or concrete data types) as well as a set of interactions that formally specify the security properties offered by the pattern. The WS architectural pattern, defining it as a set of coordinated design [17], adds a set of additional security services to which we must add a description of their interaction (interaction between security services and between security services and business services) in the functional services architecture. Therefore, these new services will offer us new security functionalities that must be reevaluated, analyzed and refined from the WSSecReq process.

As a complementary source for this stage, the WS Security Architectural & Design Repository (see Figure 1) should be used. New identified architectural patterns will be introduced within the repository allowing its future reuse.

### Definition of Security Policy associated with every Security Architectural Pattern

Furthermore, such architectural pattern defines, in an abstract way, the security policy possible parameters that can govern the identified security mechanisms capabilities and interactions.

The security policies allow Abstract Security Services and business WS to define their preferences, requirements and capabilities [21, 22]. Each Abstract Security Service, derived from one or more security requirements through the application of a certain architectural pattern(s), must indicate in its security policy the possible parameters for instancing it and the set of security requirements types it addresses (e.g.: authentication, availability, etc.).

### Integration of Security Architectural Patterns

With the purpose of obtaining a systematic method to be able to define the WS-based architecture security, we have elaborated a WS-based security reference architecture that shows the direct traceability of security requirements with their corresponding implementing software components. In our work, we have developed a WS security reference architecture which is shown in Figure 2.

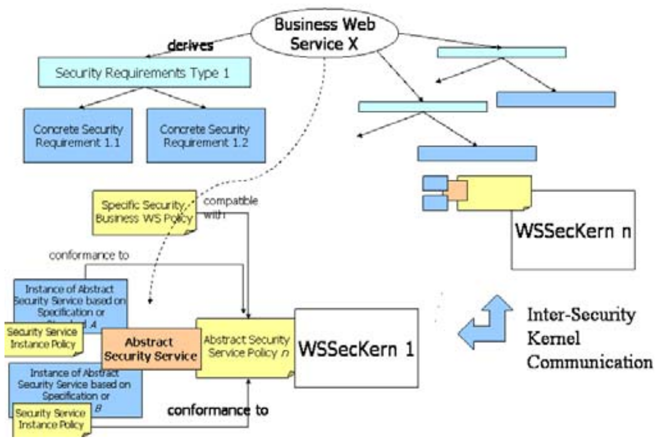
Business Web Service X
derives
Security Requirements Type 1
Concrete Security Requirement 1.1
Concrete Security Requirement 1.2
Specific Security Business WS Policy
compatible with
conformance to
Instance of Abstract Security Service based on Specification or Standard A
Security Service Instance Policy
Abstract Security Service
Abstract Security Service Policy n
WSSecKern 1
Instance of Abstract Security Service based on Specification or Standard B
Security Service Instance Policy
conformance to
WSSecKern n
Inter-Security Kernel Communication

**Fig. 2.** WS Abstract Security Reference Architecture

*Architecture Components*

The basic elements used in the WS security reference architecture are:

- WS Security Kernel (WSSecKern). This is the core of our WS-based reference security architecture. This component will manage a set of Abstract Security Services, derived from the application of a certain set of security architectural patterns, with the aim of supporting the security requirements of a potential set of business Web Services. Each WSSecKern will support one or more Abstract Security Services implemented through one or more concrete security mechanisms in form of WS security standards (identified in the stage WSSecTech). Thus, each WSSecKern will cover a set of security requirements by means of providing a set of WS-based security services to certain business WS.
- Abstract Security Service that comprises certain set of security requirements types (e.g.: security requirements related to authorization) and that can have several instances according to the number of implementations based on the WS security standards that will be identified in stage WSSecTech.
- Security Policy of an Abstract Security: it includes the possible parameters or attributes with which we can define the security policies of the Abstract Security Service potential instances as well as a description of the set of security requirements types that the Abstract Security Service handles.
- WS Security Standard that supports a certain Abstract Security Service Instance (indeed, this will not be defined until WSSecTech stage).
- Security Policy of an Abstract Security Service Instance in which the capabilities supported by the used specification or standard are defined.
- Business Service Security Policy: defined by each business WS. The business WS security policy will be registered in the WSSecKern when the business WS desires to use the security services provided by that WSSecKern. This way, the WSSecKern will know what security services, and how to use them, are demanded by certain WS. The business WS defines what set of security requirements it needs

and what mechanisms and how they will be used (e.g.: “I would like a simple message authentication based on X.509v3” certificates).
- Protocol of Intercommunication between WSSecKern: It allows the coordination and interaction of the different Security Services.

*Basic Interactions*

- Registration/cancellation of the business WS in WSSecKern. A business WS must register itself in a WSSecKern including the definition of its Business Service Security Policy. This way the WSSecKern will know what business WS should protect, what security services will have to apply and how.
- Execution of an operation of a Security Service Instance. When a request arrives at a business WS, depending on the way the system is configured, this could be intercepted by a certain WSSecKern or it could be forwarded by the business service to a WSSecKern in a way that the security service is effectively applied.

**Security Architecture Validation**

Mainly, this activity consists of verifying that the attack and security scenarios for the current iteration are covered. Besides, scenarios that have been identified so far must be reevaluated with the purpose of checking that a conflict situation has not started.

**Security Architecture Specification**

This activity states, in writing, a Security Architecture Specification document (SASSec in Figure 1) through the use of views [17, 20] that show us how the security scenarios display, through the architecture components interactions (WSSecKern and its Abstract Security Services, Agents WS Consumers and Agents WS Providers), as a countermeasure to the attack scenarios shown in the current iteration. Moreover, the specification must show the distribution of the security requirements given as input in a way that each WSSecKern must perform one or more security requirements.

**Case Study: XML Perimeter Security**

As an example, the following figure shows us a concrete security architecture covering the authentication requirements and the perimeter security [23]:

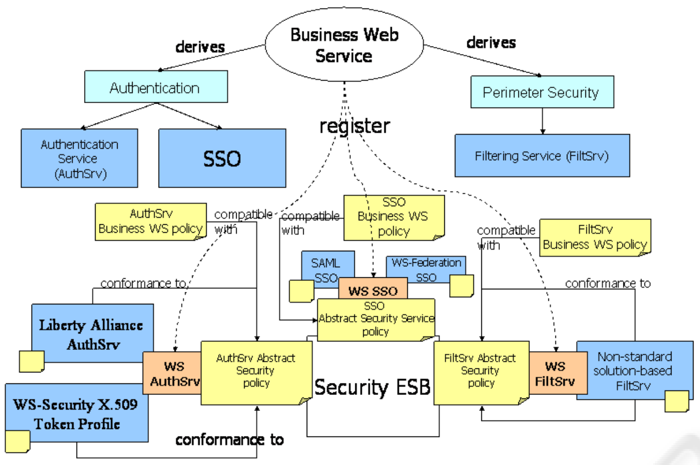
Business Web Service
derives
derives
Authentication
Perimeter Security
register
Authentication Service (AuthSrv)
SSO
Filtering Service (FiltSrv)
AuthSrv Business WS policy
compatible with
compatible with
SSO Business WS policy
compatible with
FiltSrv Business WS policy
SAML SSO
WS-Federation SSO
WS SSO
SSO Abstract Security Service policy
conformance to
conformance to
Liberty Alliance AuthSrv
WS AuthSrv
AuthSrv Abstract Security policy
Security ESB
FiltSrv Abstract Security policy
WS FiltSrv
Non-standard solution-based FiltSrv
WS-Security X.509 Token Profile
conformance to

**Fig. 3.** Concrete WS security Architecture

In Figure 3, we can see how the security requirement “Perimeter security” defined from the business WS analysis, can be specified in a Filtering Service whose aim is to filter messages according to some simple syntactic rules of them defined in their policy, and that are directed to that business service. The business service, with respect to this R1 requirement, will indicate, in its security policy, the following points: i) It requires Perimeter Security; ii) it requires Message Filtering; iii) it has R1 requirement associated; iv) filtering must be syntactic-based, applied to the message payload and based on a certain XML schema, defined in the business service policy.

In this example, the WSSecKern has been implemented as an Enterprise Security Service Bus [1] specialized in security (Security Enterprise Service Bus). The ESBSec must implement the defined semantics for a WSSecKern and has a WSFiltSrv Abstract Security Service associated derived from the Message Filtering security requirement discovered when applying WSSecReq to the given business service. WSFiltSrv defines as parameters of its policy, the definition of the formats in which the syntactic rules can be expressed (e.g.: XML Schema, DTD, RelaxNG, etc...). Instances of this Abstract Security Service must define what formats they support.

Given that there is not any standard or specification being prepared to be a standard; WSFiltSrv is implemented with priority defining in its Security Policy of the Abstract Security Service Instance, its capability to define the syntactic rules through the XML Schema. When the business WS is registered in ESBSec, it will indicate that it requires Perimeter Security of Filtering Message type and it will provide the syntactic rules in XMLSchema format as well. When ESBSec carries out the compatibility verification task, it will note that it has an Abstract Security Service that implements Message Filtering requirements type and that, also, and after having analyzed the Instances security policies (only one, in this particular case), there is at least one instance that accepts the XML Schema format for filtering syntactic rules. As verification is correct, R1 requirement, of Perimeter Security, Message Filtering” associated with such business service will be implemented by WSSecKern in the shape of ESBSec. We must note how it is possible to know at the time of execution what component or (sets of software components) implements a certain security requirement. This fact will provide us with a total requirement traceability from the security requirement to the architectural component that addresses it.

## 4 Related Work

At present, undoubtedly, the biggest effort is being carried out in the area of WS security standards definition. This effort has caused the existence of a vast number of drafts and standards that make it difficult to handle and to know them by the organizations that would like to use them. The lack of a global vision has caused that many organizations have been very reticent to use this method since they have thought it was full of acronyms. Our process allows us to face this problem in an orderly way enabling organizations to apply this method without having to know previously what draft or standard must be put to work.

With regard to the research area, EFSOC [24] is a event-driven framework for WS-based systems development that defines a security model that can be easily applied to systems in which the modifiability degree is high and therefore, they require a review and update of the authorization policies. In [25], a methodical and formal analysis based on "formal analysis of security-critical service-based software systems" is presented. None of these approximations puts forward a method such as PWSSec that, from the business and system security goals, can obtain a system based on secure WS to the standards level. Moreover, none of these methods offers us facilities for the reusability of the generated products in a way that their practical applicability is guaranteed.

## 5 Conclusion and future research

This paper has presented the process PWSSec, which allows us to provide a WS system with security through an own process. As far as authors know, there is not in the field of WS system research, a definition of a complete process comprising and taking into account all the stages of its life cycle.

Nowadays, we are applying PWSSec to two case studies carried out by two state-owned organizations. We hope that, as a result of this practical application, we could refine the stages of the process and enrich the products generated in it.

Some of the main aspects to be developed are the following: To complete the repository defined in WSSecReq stage with security requirements templates and specific attack patterns that comprise more security aspects; WS security requirements modeling and formal validation; to develop evaluation areas and cost/benefit analysis of WSSecArch; to complete the catalog of WSSecArch standards and specifications (nowadays, completed for authentication and perimeter security requirements); in the process of verification of policies compatibilities executed by WSSecKern, we still have to define if two policies are semantically equivalent.

## Acknowledgments

This research is part of the following projects: MESSENGER (PCC-03-003-1) financed by the "Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha" (Spain), CALIPO (TIC2003-07804-C05-03) and RETISTIC (TIC2002-12487-E) granted by the "Dirección General de Investigación del Ministerio de Ciencia y Tecnología" (Spain).

## References

1. Nott, C., *Patterns: Using Business Service Choreography In Conjuction With An Enterprise Service Bus*. IBM Redbooks Paper. 2004. 32.
2. IDC, *Cautious Web Services Software Adoption Continues; IDC Expects Spending to Reach $11 Billion by 2008*. 2004.
3. Gutiérrez, C., E. Fernández-Medina, and M. Piattini, *Web Services Security: is the problem solved?* Information Systems Security, 2004. **13**(3): p. 22-31.
4. Endrei, M., et al., *4. Service-oriented architecture approach*, in *Patterns: Service-Oriented Architecture and Web Services*. 2004. p. 345.
5. Endrei, M., et al., *Patterns: Services Oriented Architectures and Web Services*. IBM Redbook, ed. IBM. 2004.
6. Papazoglou, M.P. and D. Georgakopoulo, *Service-Oriented Computing.* Communications of the ACM, 2003. **46**(10): p. 25-28.
7. Alberts, C.J., et al., *Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) Framework, Version 1.0*, in *Networked Systems Survivability Program*. 1999, Carnegie Mellon. Software Engineering Institute. p. 84.
8. Smith, D.G., *Common Concepts Underlying Safety, Security, and Survivability Engineering*. 2003, SEI.
9. OMG, *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms*. 2004.
10. Bass, L. and R. Kazman, *Architecture Based Development*, in *Product Line Systems*. April 1999, Carnegie Mellon. Software Engineering Institute. p. 36.
11. Jürjens, J., *Secure Systems Development with UML*. 2005: Springer. 309.
12. Yu, H., et al. *Integrating Security Administration into Software Architecture Design*. in *International Conference on Software Engineering and Knowledge Engineering 2004*. 2004. Banff, Canada.
13. Sindre, G. and A.L. Opdahl. *Eliciting Security Requirements with Misuse Cases*. in *37th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-37'00)*. 2000. Sydney, Australia.
14. Alexander, I., *Misuse Cases: Use Cases with Hostile Intent.* IEEE Computer Software, 2003. **20**(1): p. 58-66.
15. Firesmith, D.G., *Security Use Cases.* Journal of Object Technology, 2003. **2**(3): p. 53-64.
16. Toval, A., et al., *Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach.* Requirements Engineering Journal, 2001. **6**(4): p. 205-219.
17. Bass, L., P. Clements, and R. Kazman, *Software Architecture in Practice.* 2nd, ed. 2003: Addison-Wesley. 560.
18. Ellison, R.J., et al., *Security and Survivability Reasoning Frameworks and Architectural Design Tactics*. 2004, SEI.
19. Klein, M. and R. Kazman, *Attribute-Based Architectural Styles*, in *Product Line Practice*. 1999, Software Engineering Institute. p. 90.

20. Krutchen, P., *The 4+1 View Model of Software Architecture.* IEEE Software, 1995: p. 42-50.
21. VeriSign, et al., *Web Services Policy Framework (WS-Policy)*. 2004.
22. Anderson, A., S. Proctor, and S. Godik, *OASIS XACML profile for Web-services*. 2004.
23. Cremonini, M., et al. *A XML-based Approach to Combine Firewalls and Web Services Security Specifications*. in *ACM Workshop on XML Security*. 2003. Fairfaz VA, USA.
24. Leune, K. and M. Papazaglou. *Specification and Querying of Security Constraints in the EFSOC Framework*. in *International Conference on Service Oriented Computing*. Willem-Jan van den Heuvel. New York City, USA.
25. Deubler, M., et al. *Sound Development of Secure Service-based Systems*. in *ICSOC'04*. 2004. New York, USA: ACM.