Developing web services security systems: a case study

Eduardo Fernández-Medina

University of Castilla-La Mancha
Paseo de la Universidad
4–13071 Ciudad Real, Spain
E-mail: eduardo.fdezmedina@uclm.es

Carlos A. Gutiérrez*

STL
C\Xaudaró, 15–28034, Madrid, Spain
E-mail: carlos.gutierrez@stl.es
*Corresponding author

Mario Piattini

University of Castilla-La Mancha
Paseo de la Universidad
4–13071 Ciudad Real, Spain
E-mail: Mario.Piattini@uclm.es

Abstract: This article presents a case study in which the PWSSec (Process for Web Services Security) process was applied in the development of a web services-based system. It deals with the migration of an existing system whose objective is the execution of a workflow that enables web customers to carry out payments from their account to the account of a web-based organisation. Throughout this article, we present the difficulties encountered when applying the PWSSec, as well as the solutions that were employed. Some of these solutions were developed from scratch and others were adapted from existing proposals found in current literature.

Keywords: elicitation methods; domain-specific architectures; life cycle; process; quality analysis and evaluation; risk management; security and privacy protection; software engineering for internet projects; distributed/internet-based software engineering tools and techniques; standards.

Reference to this paper should be made as follows: Fernández-Medina, E., Gutiérrez, C.A. and Piattini, M. (2006) 'Developing web services security systems: a case study', *Int. J. Web Engineering and Technology*, Vol. 2, No. 4, pp.292–306.

Biographical notes: Eduardo Fernández-Medina is PhD and MSc in Computer Science. He is Assistant Professor at the Escuela Superior de Informática of the Universidad de Castilla-La Mancha at Ciudad Real (Spain). His research activity is security in databases, datawarehouses, web services and information systems, and also in security metrics. He is co-editor of several books and chapter books on these subjects, and has several dozens of papers in national

and international conferences. He participates at the ALARCOS research group of the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. He belongs to various professional and research associations (ATI, AEC, AENOR, IFIP WG11.3 *etc.*).

Carlos Gutiérrez is MSc by the Autonomous University of Madrid (Spain) and currently, he is an Assistant Professor and PhD candidate at the University of Castilla-La Mancha, and he is working as Internet Analyst at Sistemas Técnicos de Loterías del Estado (State Lotteries' Technical Systems). He has developed his professional activity in national and international companies making consultancy activities. His research activity is focused on web services security and secure software architectures. He has several papers in international conferences and he has published diverse articles in national and international magazines on these subjects. He is participating at the ALARCOS research group.

Mario Piattini is MSc and PhD in Computer Science by the Politechnical University of Madrid, certified Information System Auditor by Information System Audit and Control Association (ISACA), Associate Professor at the Escuela Superior de Informática of the Castilla-La Mancha University (Spain) and author of several books and papers on databases, software engineering and information systems. He leads the ALARCOS research group of the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. His research interests are advanced database design, database quality, software metrics, object oriented metrics, software maintenance.

1 Introduction and motivation

The Web Services (WS, hereafter) paradigm has attained such a relevance in both the academic and the industrial world, such that the vision of the internet is evolving, passing from being considered as a mere repository of data to becoming the underlying infrastructure on which complex business processes and alliances are being deployed (Zhang, 2005).

WS-based specifications, standards, technologies and tools have showed a spectacular growth lately. This evolution has been mainly motivated by its wide adoption and promotion by the major vendors in the industry (Gutiérrez, 2004). Despite its growth, there is no development process that facilitates the systematic integration of security within all stages of the WS-based software development life cycle. To tackle this matter, we have defined the PWSSec (Process for Web Services Security) process. PWSSec has as its main purpose to facilitate and orientate the development of WS-based security systems so that in each one of the traditional stages for the construction of this kind of systems (Endrei *et al.*, 2004a), a complementary stage comprising security (Endrei *et al.*, 2004b) can be easily integrated. Therefore, this process can be used once the functional architecture of the system has been built or during the development stages of this architecture. In both cases, the result will be a security architecture formed by a set of coordinated security mechanisms that use the WS security standards to fulfil the system security requirements.

In this paper, we present how this process has been applied to a real case study. It mainly focuses on how the WS-specific security requirements have been elicited.

The rest of the article is organised as follows: in Section 2, the current case study is described; in Section 3, the application of the PWSSec's WSSecReq (Web Services Security Requirements) stage to the case study is developed; and finally, in Section 4, conclusions are presented.

2 Case study of a bank transfer system

In this article we present a case study that was applied to a WS-based system known as WS-BTS (Web Services-based Bank Transfer System). The objective is the sale of certain products, which were selected by the purchasers through a web application. Payments are made from the purchaser's bank account, which is associated with the bank account of the sales organisation.

In Figure 1, a general diagram is presented of the present system. This is a WS-based system and consists of at least two different agents:

1 a WS consumer agent, belonging to the sales organisation, which will be referred to as WS-BTSConsumer (Web Services-based Bank Transfer System Consumer)

2 the WS provider agent of the bank service that will be referred to as WS-BTSProvider (Web Services-based Bank Transfer System Provider). These agents interact in order to fulfil a business workflow that aims to assist the final customer during payment and to facilitate the purchase.

The current workflow is represented as a use case entitled Performing Bank Transfer (PBT), as seen in Figure 2.

This use case is achieved by following a three-step protocol:

1 When the web application receives the sales order from the purchaser, including the bank's identity through which he wishes to make the payment, it requests the WS-BTSConsumer to create a security token and send it through the internet to the WS-BTSProvider agent located at the solicited bank organisation. The user's browser is then redirected to the bank's website. The aforementioned security token, created primarily from product-related information, including information identifying the purchaser in the bank system (there is a network identity federation based on pseudonyms between the web sales domain and those of different banks (Liberty Alliance Project, 2003)), is computationally impossible to be repeated in more than one transaction. The objective of this interaction is to signal to the WS-BTSProvider agent the user's desire to complete a transfer in order to pay for a certain product. Hereafter, we will refer to this interaction as 'Token for New Transference'.

2 The banking entity's system, which received the transfer request from the WS-BTSProvider agent, will then proceed to authenticate the buyer's identity. Once authorised to make the transfer, it will request the WS-BTSProvider agent to re-solicit the token from the WS-BTSConsumer service, which eventually will return it. This interaction is named 'Transfer Token Request'.

3 Once the token is received and verified, bit by bit, to be the original from the web application, the WS-BTSProvider agent finalises the transfer and generates a unique reference number, not to be duplicated in other transfers (including transfers between different bank organisations), which is then sent to the WS-BTSConsumer.
The client checks this number and sends a confirmation message of the completion of the transfer. This step is referred to as the ‘Reference Number Processing’ interaction.

The development of this workflow was designed in the beginning of 2001, when security WS-based standards and technologies were still somewhat undeveloped. Because of this lack of experience and maturity, a tailor-made solution was opted for, based on the exchange of XML documents concerning HTTPS (architectural style known as REST (Fielding, 2000)).

Figure 1 Overview of the WS-BTS system

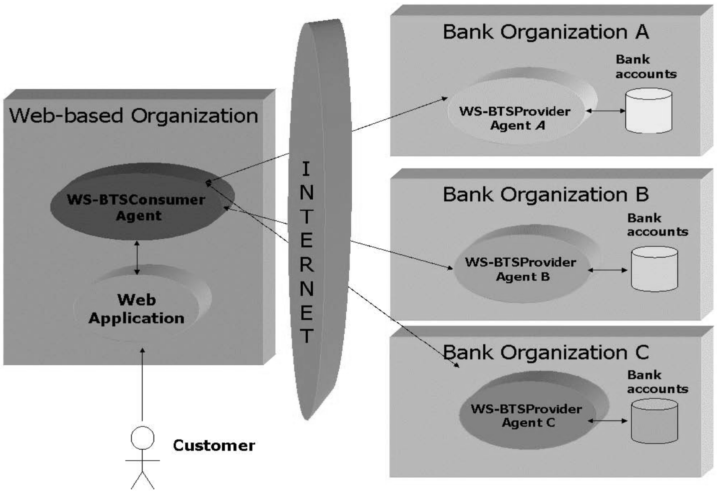
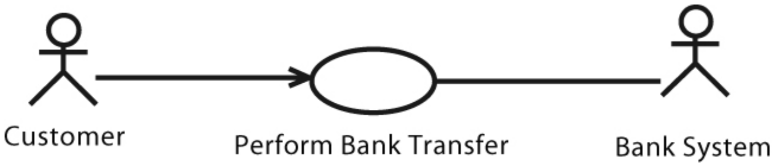


Figure 2 Use case ‘Perform Bank Transfer’



In brief, the requirements for basic security used for the said systems were:

- mutual authentication, permitting identity verification between partners
- integrity, allowing the detection of interference in the exchanged messages
- confidentiality, guaranteeing that the exchanged information would only be visible to the receiving partner.

The biggest problem facing this system is its low interoperability and integrity, owing to its being a tailor-made system without a standardised design. The costs of integrating with bank systems are very high and require considerable (human) interaction. Consequently, it becomes necessary to outfit a standard character requiring minimal interaction between the sales organisation and the banking institution. Then, with a list of WS-based (security) standards and a specified workflow using a standardised language (*e.g.*, BPEL), the banking organisation is capable of independently developing and testing its own WS-BTSProvider agent.

The scope of the application of the PWSSec process to this system is limited to secure every interaction between the web services agents, regardless of whether the whole protocol is secure or not (we assumed it is). The main objective of applying PWSSec to this case study was to verify that we could systematically come up with a set of WS-based security requirements, a WS-based security architecture and a set of WS-based security standards that could discover and address the main security concerns, define a coordinated security mechanism-based solution, and 'standardise' the aforementioned interactions (that is, the system). Thus, there is a clear opportunity to apply this process in a relatively small, real system serving as a proof-of-concept for the process in question.

One of the main advantages of the PWSSec process in this case study is that developers possess a thorough understanding of the domain of the problem, and can therefore apply the process in a detailed manner. The PWSSec process has various stages, and in this article we will apply the WSSecReq stage concerning the case study to clearly show how the security system requisites were elicited and specified.

3 Application of WSSecReq to the BTS system

3.1 Elements of WSSecReq

In this system we will apply the WSSecReq stage concerning the case study to clearly show how the security requirements to the aforementioned system are elicited and specified.

Eliciting the security requirements in the WSSecReq stage includes the following aspects:

- identification of the functional WS to be protected
- identification of potential types of attackers
- identification of possible threats and attacks
- evaluation of the impact of the attack
- estimation and prioritisation of security risks
- selection of security subfactors
- specification of security requirements by selecting the most suitable templates from the repository of reusable requirement templates specific to WS
- determination of the criterion and metrics for appropriate security and determination of the minimum level of acceptable for the selected metrics and the identified risks
- specification of the security requirements.

Our process of eliciting is based on the security artefacts depicted in Figure 3:

- IBM Web Services-based business and application patterns (Endrei *et al.*, 2004c)

  We use these patterns as guides to systematically locate potential threats being structured in the form of Threat/Attack Trees. First, the appropriate WS-based Business Pattern is identified based on the proposed problem and then, the possible WS-based Application Patterns, which correspond to the interaction under analysis, are chosen. The idea is that, by beginning with the identification of the problem with the pattern, we will be able to systematically obtain a set of potential threats and attacks to the system.

- Adaptation of the attack trees (Schneier, 1999)

  For every IBM WS-based Business and Application pattern, we have defined an associated Abstract Tree of Threats and Attacks (ATTA). The ATTA associated with the IBM WS-based Business Pattern is used to establish the base of our tree. This business ATTA will be instanced with a tree-like structure to obtain the set of business level-threats, which should be considered in the system under analysis. This first security artefact will facilitate developers during the analysis of the security of the business problem and the context within the computer system to be constructed is going to be deployed. At the application level, we should determine what IBM WS-based Application Pattern is followed by the functional architecture. We have defined an ATTA for every IBM WS-based Application Pattern that states, in a tree-like form, the set of possible threats to the elements contained within the pattern. This application-level ATTA will be instanced and will refine the business-level tree.

- Table of threats, attacks, and risks

  This table will be constructed little by little as we continue refining the tree of threats and attacks. It will allow us to reflect the impact of the identified threats and the associate risk for each type of attack that fulfils those threats.

- Attack profiles (Moore *et al.*, 2001)

  Define an architectural design, offer a series of variants, gather a series of Abstract Misuse Cases, and have a glossary of terminology. In our case, we have defined an Attack Profile called *Unidirectional WS Internet-based Interaction.* We have defined a set of Abstract Misuse Cases associated with this profile and should be instanced, establishing the value of their variants (as defined by the Attack profile to which they belong). The instances of those Abstract Misuse Cases will refine the leaves (that at this point are threats) of the TTA, defining the possible attack alternatives.

- Table of threats, attacks and risks

  Each threat should have an assigned value that quantifies the damage represented by its realisation. Each attack, defined by the instances of the Abstract Misuse Cases should have an assigned probability so as to understand its threat and recognise with which threat it is most critical and most probable. The order in which we will continue to analyse each Abstract Misuse Case instance will be based on this value.

- Misuse cases (Alexander, 2003; Sindre and Opdahl, 2000)

  Define the possible attack paths of the associated threats. We have Abstract Misuse Cases and Concrete Misuse Cases. The former are grouped under Attack Profiles, while the latter are instances of the former, which define the sequence used to perform certain types of attacks. Every Abstract Misuse Case is associated with one or more Security Use Cases.

- Security use cases (Firesmith, 2003)

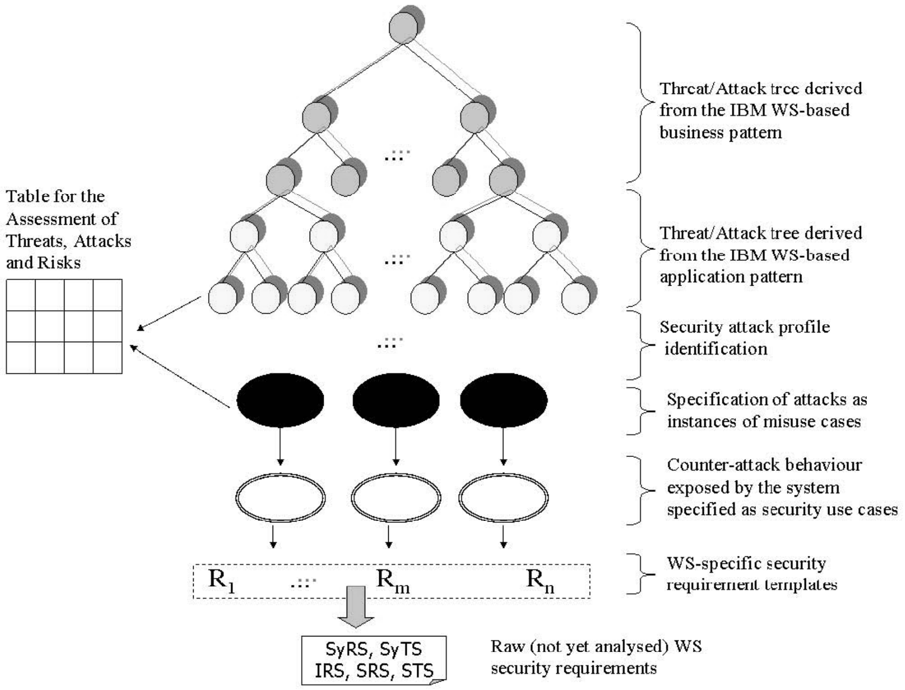
  Define a sequence of steps that allow the system to detect, prevent or survive each of the attacks formalised as instances of Misuse Cases. Each Abstract Security Use Case will have one or more associated templates of WS-based security requirements that should be instanced in order to obtain the final security requirements.

- WS-based security requirement templates (Firesmith, 2004) and documentation of WS-based reusable security requirements (Toval, 2001)

- WS security requirements specification

  These are the final products of this stage and will allow us to make the appropriate architectural decisions with regard to security.

Figure 3 Security artefacts used during the elicitation of the security requirements


Threat/Attack tree derived from the IBM WS-based business pattern
Table for the Assessment of Threats, Attacks and Risks
Threat/Attack tree derived from the IBM WS-based application pattern
Security attack profile identification
Specification of attacks as instances of misuse cases
Counter-attack behaviour exposed by the system specified as security use cases
R_1 .::· R_m R_n
WS-specific security requirement templates
SyRS, SyTS IRS, SRS, STS
Raw (not yet analysed) WS security requirements

3.2 Traceability in WSSecReq

Through this set of elements, we are able to establish a relationship of traceability that allows us to associate WS-based Business and/or Application services with the set of related WS-based security requirements. This traceability allows for future projects having the same business or application pattern, to re-use the security set requirement templates, as well as the intermediate set of security abstract artefacts. The basic plan to reach the specification of the security requirements is shown in Figure 3, where it is seen that, beginning with the specification of the service to be protected, the goal is to find a WS-based pattern from which we can derive a tree of potential threats. This tree will be refined through the WS-based pattern and shall have, as leaves, the potential set of threats to the service being analysed. From this set of threats, and with the appropriate Attack Profile, the misuse cases are instanced as black, oval-shaped figures, automatically derived from the security use cases represented by the thickly-bordered, oval-shaped figures. Each Security Use Case has one or more associated WS-based security requirements template, which will be instanced in order to obtain the appropriate set of security requirements.

In this manner, a direct and logical relationship is established between the elicited security requirements and the functional WS they are derived from. To our understanding, this approximation, which is directed at the elicitation and analysis of WS security requirements, is unique in how it treats the combination of all these security elements.

3.3 Case study application

Subsequently, it will be shown how this set of elements was applied to the case study presented in the first section. The IBM WS-based Business Pattern associated with our system is known as Extended Enterprise and has the associated ATTA shown in Figure 4.

Figure 4 ATTA associated with the IBM WS-based business pattern Extended Enterprise

Objective: 1. To cause harm during the Execution of [Business Process, Work Flow, Service, Use Case] [Name].

1. To attack business entity [name]
2. To attack the network [name]
3. To attack business rules [name]
4. To attack interaction [name]

In our case, we instance this tree in order to obtain the result shown in Figure 5.

As previously stated, each WS-based Business pattern has a related set of lower abstraction level application patterns. In our case, the IBM WS-based Application pattern used is known as *Exposed Direct Connection*. This application pattern represents the simplest form of interaction based on a one-to-one topology.

This application pattern has two possible variations: ‘Variation based on Message’ or ‘Variation based on Invocation’. The ‘Token for New Transference’ interaction fits perfectly with the ‘Variation based on Message’ given that the WS-BTSConsumer sends a one-way message and continues with its execution (see Figure 6).

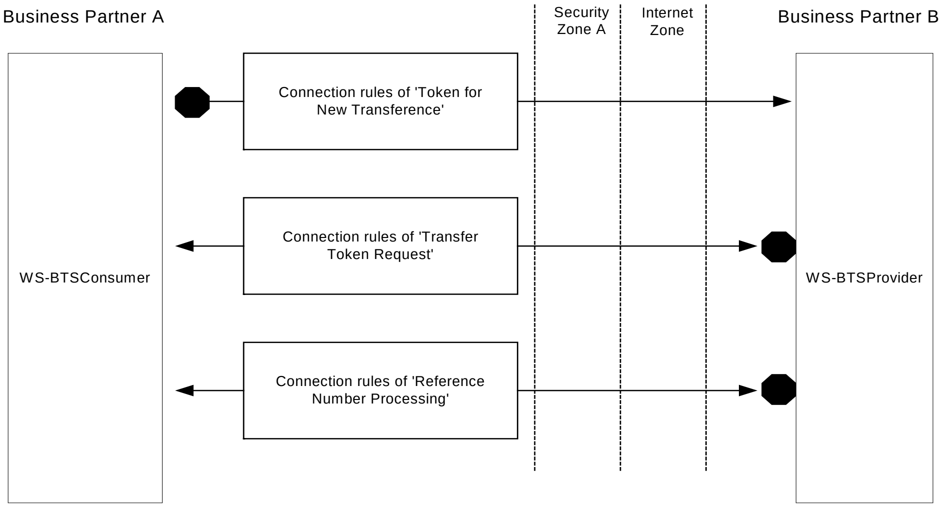
Figure 5 Resulting business-level ATT for the ‘Perform Bank Transfer’ use case

Objective: To cause damage during the execution of the use case ‘Perform Bank Transfer’.

1. To attack business entity
    1. To attack banking organisation
    2. To attack web sales organisation
2. To attack the network
    1. To attack internet
        1. To attack DNS
3. Attack business rules.
    1. Attack business rules of ‘Perform Bank Transfer’ use case

       …
4. Attack interaction.
    1. Attack Interaction ‘Token for New Transference’
    2. Attack Interaction ‘Transfer Token Request’
    3. Attack Interaction ‘Reference Number Processing’

Figure 6 Characterisation of the IBM WS-based application pattern known as Direct Exposed Connection in our work-flow


Business Partner A
Security Zone A
Internet Zone
Business Partner B
Connection rules of 'Token for New Transference'
Connection rules of 'Transfer Token Request'
Connection rules of 'Reference Number Processing'
WS-BTSConsumer
WS-BTSProvider

In our case, the elements that define the application pattern *Direct Exposed Connection in its Variation based on Message* are:

- Origin system: the WS-BTSConsumer agent
- Destination system: WS-BTSProvider agent
- Connection rules that are defined by the protocol specification in each interaction. In the ‘Token for New Transference’ case, we have called the Rules of Connection ‘Token for New Transference’. Basically, these connection rules define the current representation of the data, including its security, through the application of encryption and integrity.

- Regarding zones, it is necessary to indicate that the zone found between two businesses or organisations is the internet, so that one must consider that the interactions will take place in an environment accessible to the public, and where neither of the two organisations has control.

We have defined an ATTA associated with this application pattern which, due to space limitations, will not be shown. This tree allows us to refine the associated TTA Business pattern of the Extended Enterprise, which contains the WS-based Application pattern Direct Exposed Connection. The leaves of the instance of the application-level TTA will be a set of threats that will be refined with one or more attack scenarios (WS-I, 2005).

For the 'Token for New Transference' interaction, which consists of a one-way message between the WS-BTSConsumer and the WS-BTSProvider, the following threats were identified (only three are shown in the example):

1 'Alteration of the Message *Token for NewTransference*'

2 'Falsification of the Message Token for NewTransference'

3 'Confidentiality of Message Token for NewTransference'.

Once the threats were identified, their impacts were evaluated. For the time being, we will assess the impact with one of three possible values: HIGH, MEDIUM, and LOW. The message alteration threat would have a LOW impact on the system if the 'Token for New Transference' is requested again during the 'Transfer Token Request' interaction to detect any attack form of this threat.

Likewise, this would happen with the Falsification Message only if it occured in the 'Token for New Transference' interaction. Finally, the 'Confidentiality of Message Token for New Transference' gets a HIGH mark when the capability of accessing the critical message data means not being able to follow one of the established objectives: to safeguard the reputation of the participating organisations. For example, the attacker would be able to read the information and make it public on the internet. Once we have initiated the table construction, we continue refining our tree of threats.

The threats alone are insignificant if there is no existing attack that can manifest. Therefore, the set of possible attacks that could result from each threat was identified. To accomplish this, the Attack Profile was used, as described in Moore *et al.* (2001). The attack patterns described in this work prove to be less than efficient if compared with the misuse cases defined in Sindre and Opdahl (2000). As both devices have the same objective, we chose to use the second type when defining our attack profile, to define the sequence of steps that lead to the successful completion of a system attack. In our case, and given that it was the first time to apply the PWSSec process, we did not have the predefined attack profiles, therefore we created the attack profile: *One-way WS-based Internet Message.*

As indicated in Figure 3, each WS-based application pattern is related to one or more attack profiles (abstract), which define the attacks that may threaten it. The IBM WS-based application pattern Direct Exposed Connection has been assigned as its attack profiles the *One-way WS-based Internet Message* (for the 'Variation based on Message') and *Request-Reply WS-based Internet Message* (for the 'Variation based on Invocation'). From the set of misuse cases defined for these profiles, we will explain the misuse case related to the integrity, which we have named *Misuse Case Attack on the Semantics of the*

Message SOAP [name] which, in turn, refines the A3Ap leaves of the application pattern *Directly Exposed Connection,* referring to the threats of message alteration and forgery (see Figure 8).

Figure 7 Abstract TA tree associated with the WS-based application pattern named Exposed Direct Connection

```
Abstract TA Tree for WS-based Application Pattern Extended Enterprise::Exposed Direct Connection

              ID: A3Application-EDC.

              Objective 1. Attack Exposed Direct Connection WS-based Application Pattern
                     AND/OR   1. Attack System

                              AND/OR 1. Attack Source Agent
                                     2. Attack Target Agent

                              2. Attack Zone

                              AND/OR 1. Attack Secure Zone
                                     2. Attack Demilitarised Zone
                                     3. Attack Inter-business Zone

                              3. Attack Connection Rules
                              AND/OR 1. Attack Connection Rules Type A
                                        …
                                     N. Attack Connection Rules Type N

                              4. Attack System Interactions
                              AND/OR 1. Attack Interaction A

                                   AND/OR   1. Attack Message Channel

                                      AND/OR  1.   Alteration of Message
                                              2.   Confidentiality of Message
                                              3.   Falsified Messages
                                              4.   Principal Spoofing
                                              5    Forged Statements
                                              6    Replay of Part of the Message
                                              7    Replay of the Whole Message
                                              8    Denial-of-Service

                                            2. Attack WS Actions

                                             AND/OR 1. Attack Underlying Infrastructure
                                     2. Attack Interaction B
                                     AND/OR   1. Attack Message Channel

                                                AND/OR 1. Alteration of message

                                                       2. Confidentiality of Message

                                                     …

                                         …

                                     N. Attack Interaction N
```

Figure 8 Abstract misuse case ‘Attack on the Semantic Content of the SOAP’

<table>
<tr><th colspan="4">Name of Misuse case: Attack on the Semantic Content of the SOAP [message | interaction] [message | interaction name]</th></tr>
<tr><td colspan="2"><i>ID: AMUC-1-1-1</i></td><td colspan="2"></td></tr>
<tr><td colspan="2">PROBABILITY [HIGH] [MEDIUM] [LOW]</td><td colspan="2"></td></tr>
<tr><td colspan="4">Summary: the attacker type [attacker type] gains access to the [message | interaction] [name] exchanged by the [consumer | provider | discovery] agent [agent name] and the [consumer | provider | discovery] agent [agent name] and [modifies | deletes | inserts [part]*] of the message at the [transport | SOAP]-level situated in the [header | body | attachment] with the object of [objective].</td></tr>
<tr><td colspan="4">Preconditions:
1) The attacker has physical access to the message.
2) The attacker has clear knowledge of the structure and meaning of the message.</td></tr>
<tr><td><b>Interactions of the Consumer Agent</b></td><td colspan="2"><b>Interactions of the Misuser</b></td><td><b>Interactions of the Provider Agent</b></td></tr>
<tr><td>The Consumer Agent sends the message [name of message]</td><td colspan="2"></td><td></td></tr>
<tr><td></td><td colspan="2">The attacker [type of attacker] [name of attacker] intercepts it</td><td></td></tr>
<tr><td></td><td colspan="2">The attacker [type of attacker] [name of attacker] identifies the part to modify and [deletes replaces adds] information</td><td></td></tr>
<tr><td></td><td colspan="2">The attacker forwards the message to the Provider Agent</td><td></td></tr>
<tr><td></td><td colspan="2"></td><td>The Provider Agent receives the message and processes it erroneously due to the altered semantic content.</td></tr>
<tr><td colspan="4">Postconditions
1) The system will remain in a state of error with respect to the original intentions of the Consumer Agent [name of consumer agent].
2) In the register of the system in which the Provider Agent [name of provider agent] was executed the request received with an altered semantic content will be reflected.</td></tr>
</table>

Additionally, and thanks to the information received in the defined attack profiles, we identified three potential attackers, primary performers in the misuse cases proposed:

1 Malicious Agent WS-BTSProvider

It is possible that the WS-BTSProvider agent will not behave as expected, performing illicit acts, such as revealing the identity of the buyers for his own benefit (by selling this information, creating buyer profiles to impersonate the buyer, *etc.*). The risk that the WS-BTSProvider agent will behave in this fraudulent manner is very low, given that there is a trust relationship between the participating organisations protected by legal contracts.

2 External Attacker

Attacker found in the internet capable of committing one of the aforementioned attacks. The risk existing with this type of attacker is very high given the unpredictability and uncontrollable nature of the internet.

3 Intermediate WS agent

In the SOAP architecture, upon which the web services are based, there is an intermediate SOAP node figure that is capable of processing the messages during their transmission. It is possible, and permitted, that a transmitting agent will disclaim the existence of this type of intermediary in the course of the sent messages.

Next, we defined the abstract misuse case CMUA-1-1-1 Attack on the Message SOAP's Semantic and it was instanced for the 'Token for New Transference' interaction. For the abstract misuse case mentioned, we specified and instanced the abstract security use case *To Guarantee the Semantic Integrity of the SOAP Message.* This abstract security use case has two assigned security requirement templates. One of them is shown in Figure 9.

Figure 9 Security requirements template

> The *[[consumer agent | provider agent discovery agent] [agent name]]* shall protect the message *[message name]* at *[transport <protocol> | SOAP message | both]* level that transmits from possible *[modifications removal | insertions]* on *[message parts]* altering its semantic due to *[non-sophisticated semi-sophisticated | sophisticated]* attacks during the *[[interaction type] [interaction | use case]]*+ execution.

A sample instance of this template would be:

> "The WS-BTSConsumer consumer agent should protect the SOAP message 'Token for New Transference' sent at both, the HTTP transport-level and the SOAP message-level, which transmits from possible modifications, eliminations, and insertions altering the identity of the recipient user due to sophisticated attacks to its integrity during the execution of the Token for New Transference interaction in the 99.9% of its executions."

The final security requirement specification is based on the method for requirements engineering based on requirements reused called SIREN (Toval *et al.*, 2001) and its coding will be derived from the codes of the intermediate security artefacts from which it was created.

With this example of security requirements, we have shown how, through an IBM WS-based business pattern associated with the service under analysis, we were able to define its security requirements. As a result of applying this process to each of the iterations, we obtained the following: a set of business and application-level threats and attacks structured in tree-like form, a set of misuse cases representing the potential attacks, a set of security use cases showing how the system could prevent, detect and react to the attacks and a set of WS-based security requirements. In addition, as it was the first time the PWSSec process was applied, we obtained the set of abstract security artifacts, from which the aforementioned products were derived. This set of abstract security artefacts were appropriately introduced into the WSSecReq stage's repository of reusable security artefacts.

4 Conclusions

In this article we have presented a case study in which we applied the WSSecReq stage, as part of the PWSSec process, defined by the authors. The process followed has been used in order to elicit the security requirements of a WS-based system from a set of intermediate artefacts offering complete traceability.

Certain factors remain pending in regard to this phase; for example, determining and expanding on the analysis and risk factors during the elicitation process, so as to follow an even more rigorous risk analysis method than the one described.

Likewise, we should decide upon the relationships between the alternatives to the threat tree (which possibly will generate relationships of inclusion/exclusion among the requirements that are derived (Toval *et al.*, 2001)), how to associate the work-flow with certain standards (in our case, we ultimately related 'intuitively' with the specification Security Assertion Markup Language (OASIS, 2003)), to model in this standard manner some of the developed artefacts, such as the misuse cases, using UML QoS (Quality of Service) profile (OMG, 2004).

Acknowledgements

This study was carried out within the framework of the CALIPO project (TIC2003-07804-C05-03) and the RETISTIC network (TIC2002-12487-E), through the Dirección General de Investigación del Ministerio de Ciencia y Tecnología.

References

Alexander, I. (2003) 'Misuse cases: use cases with hostile intent', *IEEE Computer Software*, Vol. 20, pp.58–66.

Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, P., Luo, M. and Newling, T. (2004a) 'Patterns: service-oriented architecture and web services', *IBM Redbook*, 1st ed., p.345.

Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, P., Luo, M. and Newling, T. (2004b) *Patterns: Services Oriented Architectures and Web Services.*

Endrei, M., Diotalevi, F., Dostal, W., Elich, K., Manginapudi, U., Neave, W. and Patrey, W. (2004) 'Patterns: direct connections for intra- and inter-enterprise', *IBM Redbooks, IBM*, Vol. 2004.

Fielding, R.T. (2000) 'Architectural styles and the design of network-based software architectures', *Software Research Group*, University of California, Irvine.

Firesmith, D.G. (2003) 'Security use cases', *Journal of Object Technology*, Vol. 2, pp.53–64.

Firesmith, D.G. (2004) 'Specifying reusable security requirements', *Journal of Object Technology*, Vol. 3, pp.61–75.

Gutiérrez, C., Fernández-Medina, E. and Piattini, M. (2004) 'A survey of web services security', *Presented at Workshop on Internet Communications Security 2004 (WICS 2004), in conjunction with the 2004 International Conference on Computational Science and Its Applications (ICCSA 2004)*, Assisi (PG), Italy

LibertyAllianceProject (2003) 'Liberty ID-FF Architecture', *Overview*, Vol.1.2

Moore, A.P., Ellison, R.J. and Linger, R.C. (2001) *Attack Modelling for Information Security and Survivability*, Software Engineering Institute.

OASIS (2003) *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)*,Vol. 1.1.

OMG (2004) *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms*.

Schneier, B. (1999) 'Attack trees: modeling security threats', *Dr. Dobb's Journal*.

Sindre, G. and Opdahl, A.L. (2000) 'Eliciting security requirements with misuse cases', *Presented at 37th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-37'00)*, Sydney, Australia.

Toval, A., Nicolás, J., Moros, B. and García, F. (2001) 'Requirements reuse for improving information systems security: a practitioner's approach', *Requirements Engineering Journal*, Vol. 6, pp.205–219.

WS-I (2005) *Security Challenges, Threats and Countermeasures Version 1.0*, Vol. 2005.

Zhang, J. (2005) 'Trustworthy web services: actions for now', *IEEE IT Pro*, Vol. 7, pp.32–36.