THE UNIVERSITY OF HULL

# An Investigation of Interoperability Issues between Authorisation Systems within Web Services

being a Thesis submitted for the Degree of

**Doctor of Philosophy**
**in Internet Computing**

in the University of Hull

by

**Yunxi Zhang**

September 2014

# Table of Contents

Acknowledgements ........ I

Abstract ........ II

Chapter 1. Introduction ........ 1

1.1 Background and Motivation ........ 1

1.2 Problem Statement, Research Aims, Questions and Objectives ........ 4

1.2.1 Research problems ........ 4

1.2.2 Research questions ........ 5

1.2.3 Research aims ........ 5

1.2.4 Objectives ........ 5

1.3 Research Methodology and Methods ........ 6

1.4 Scope and Limitations of the Research Contributions ........ 8

1.5 Thesis Outline ........ 9

1.6 Chapter Summary ........ 11

Chapter 2. Review of Security-Related Standards, Authentication Services, RBAC, ABAC Approaches and the Relevant Authorisation Systems Within Web Services 12

2.1 Introduction ........ 12

2.2 Definition of Web Services ........ 13

2.3 Interoperability Issues and Protocols ........ 15

2.4 Security-related Standards, Authentication Services Within Web Services ........ 19

2.4.1 Official consortiums and protocol standardisation ........ 19

2.4.2 XML and Schema ........ 19

2.4.3 SOAP ........ 20

2.4.4 WSDL ........ 22

2.4.5 UDDI ........ 22

2.4.6 XML-Encryption and XML-Signature ........ 24

2.4.7 WS-Security ........ 25

2.4.8 WS-Trust ........ 26

2.4.9 WS-SecureConversation ........ 27

2.4.10 WS-Policy and WS-SecurityPolicy ........ 28

2.4.11 SAML and XACML ........ 29

2.5 Access Control Methods ........ 31

2.5.1 RBAC ........ 32

2.5.2 ABAC ........ 34
2.6 Existing RBAC/ABAC-based Authorisation Systems ........ 35
2.7. Existing Solutions for Providing Interoperability between ABAC-based Authorisation Systems within Web Services ........ 36
2.7.1 SAML Messages ........ 36
2.7.2 An object-oriented framework for adopting different policy languages ........ 39
2.8 Limitation of the Application of ABAC within Web Services ........ 40

Chapter 3. Trust Negotiation and Interoperability: State of the Art ........ 42
3.1 Introduction ........ 42
3.2 Concept of TN ........ 43
3.3 Review of TN ........ 45
3.3.1 Strategy ........ 46
3.3.2 Credential and policy ........ 54
3.3.3 Declaration ........ 56
3.3.4 Existing TN-based authorisation systems ........ 56
3.4 Interoperability Issues between Authorisation Systems in Web Services ........ 61
3.5 An Improved Multi-layered Interoperability Model ........ 67
3.6 Related Work ........ 73
3.7 Research Problem ........ 74
3.8 Discussion of Potential Solutions ........ 75
3.9 Chapter Summary ........ 76

Chapter 4. A Protocol-based Approach for Providing Interoperability between Authorisation Systems within Web Services ........ 77
4.1 Introduction ........ 77
4.2 Overview of A Protocol-based Solution Design ........ 79
4.3 Protocol Requirements Elicitation ........ 83
4.4 Overview of An Improved TN Protocol ........ 90
4.4.1 Scope and limitation of the protocol ........ 90
4.4.2 An improved TN protocol ........ 90
4.5 Preparation Stage ........ 92
4.5.1 Step one – Sends out a <TNPrepareRequest> message ........ 93
4.5.2 Step Two – Receives an incoming <TNPrepareRequest> and sends out an outgoing <TNPrepareResponse> message ........ 94
4.6 Negotiation Stage ........ 97
4.6.1 Step one – Receives an incoming <TNPrepareResponse> message and sends out an outgoing <AuthzDecisionQuery> message ........ 97

4.6.2 Step two – Receives an incoming <AuthzDecisionQuery> message and decides to sends a relevant outgoing message ........ 99
4.6.3 Possible intermediate steps – Receives an incoming <PolicySet> message and sends out a relevant outgoing message ........ 100
4.6.4 Possible intermediate steps – Receives an incoming <CredentialSet> message and sends out a relevant outgoing message ........ 105
4.6.5 Possible second last step – Receives a <Response> message from a Web Service Requester and sends out an <AuthzDecisionStatement> message ........ 111
4.6.6 Last step – Receives an <AuthzDecisionStatement> message from a Web Service Provider ........ 111
4.7 Chapter Summary ........ 112

Chapter 5. Protocol Verification ........ 114
5.1 Introduction ........ 114
5.2 Discussion of Methods for the Completeness Proof ........ 116
5.3 Introduction of An Example FSM ........ 117
5.4 FSM-based Completeness Proof of the Protocol ........ 120
5.4.1 Overview of the states of the protocol ........ 120
5.4.2 FSM-based completeness proof of the preparation stage ........ 124
5.4.3 FSM-based completeness proof of the negotiation stage with the use of a parsimonious strategy ........ 125
5.4.4 FSM-based completeness proof of the negotiation stage with the use of an eager strategy ........ 130
5.5 Identified Innate Vulnerabilities ........ 133
5.6 Impact of the FSM Approach ........ 133
5.7 Chapter Summary ........ 134

Chapter 6. Remembrance of Local Information Status for Enforcing Robustness of Policy-Exchanged Strategies for Trust Negotiation ........ 135
6.1 Introduction ........ 135
6.2 Innate Vulnerability Issues in TN ........ 136
6.3 A Proposed Solution Design ........ 138
6.3.1 Discussion of one potential solution ........ 138
6.3.2 Conceptual idea of a proposed solution ........ 139
6.3.3 Realisation technique ........ 140
6.4 Evaluation Result Analysis ........ 143
6.5 Impact of the Proposed Solution ........ 145
6.6 Chapter Summary ........ 147

Chapter 7. Protocol Evaluation ........ 148
7.1 Introduction ........ 148
7.2 Evaluation Research Method ........ 148
7.3 Use of Case Studies ........ 150
7.3.1 A general procedure of conducting case studies ........ 150
7.3.2 Types of case studies ........ 151
7.3.3 Construction of case studies ........ 154
7.3.4 Methods and process for data collection ........ 156
7.3.5 Model-based testing of protocol application within embedded single-case study for data collection ........ 159
7.4 Discussion on Qualitative Data Analysis ........ 186
7.4.1 Data analysis strategy and technique ........ 186
7.4.2 Data analysis results for interoperability between authorisation systems and correctness of the protocol ........ 187
7.5 Limitations of the Protocol or of the Interoperability-Solution Design ........ 191
7.5.1 Conceptual interoperability issue ........ 191
7.5.2 Extra required functionalities ........ 192
7.5.3 No common strategy or capability between two entities ........ 192
7.5.4 Performance tests challenge ........ 193
7.6 Related Work ........ 193
7.6.1 Differences between the protocol and state-of-the-art TN-based authorisation systems ........ 193
7.6.2 Differences between the protocol and state-of-the-art TN-based authorisation systems in Web Services ........ 195
7.6.3 Differences between the protocol and an existing solution in Web Services ........ 196
7.6.4 Differences between the protocol and state-of-the-art ABAC protocols in Web Services ........ 196
7.7 Impact of the Research ........ 196
7.8 Chapter Summary ........ 198
Chapter 8. Conclusions and Future Work ........ 199
8.1 Introduction ........ 199
8.2 Discussion of the Research ........ 199
8.3 Research Contributions and Impacts ........ 210
8.4 Future Work ........ 212
8.4.1 Exploring factors relevant to conceptual interoperability ........ 212
8.4.2 Uncommon strategy and capability issue ........ 213
8.4.3 Performance test challenge ........ 213

8.4.4 Security consideration......213
8.5 Chapter Summary......215
References......216
Appendix A. Protocol Messages......244
Appendix B. Case Study Evaluation in Chapter 6......254

# List of Figures

Figure 2.1. AAI solutions and functionalities (Schlager et al., 2006) .......... 37

Figure 2.2. Attribute-based AAI reference model (Schlager et al., 2006) .......... 38

Figure 3.1. Illustration of an interoperability issue in relation to capability in a first circumstance .......... 63

Figure 3.2. Illustration of the interoperability issue in relation to capability in a second circumstance .......... 64

Figure 4.1. A protocol design and development methodology –Validation .......... 78

Figure 4.2. Overview of an improved TN protocol .......... 91

Figure 4.3. A Web Service Requester sends out a <TNPrepareRequest> message to a Web Service Provider .......... 94

Figure 4.4. A Web Service Requester processes a <TNPrepareRequest> message .......... 96

Figure 4.5. A Web Service Requester processes a <TNPrepareResponse> message .......... 98

Figure 4.6. A Web Service Provider processes an <AuthzDecisionQuery> message .......... 100

Figure 4.7. A Web Service processes a <PolicySet> message and decides to compare local credentials with the received policies .......... 101

Figure 4.8. A process illustrating how the policy compliance checker makes a decision .. 103

Figure 4.9. A process of verifying the authenticity and credential chains of the received credentials to decide what to do .......... 106

Figure 4.10. A process of comparing the received credentials against local policies by using the eager strategy .......... 108

Figure 4.11. A process of comparing the received credentials against local policies by using the parsimonious strategy .......... 110

Figure 4.12. A Web Service Provider processes a <Response> message .......... 111

Figure 4.13. A Web Service Requester processes an <AuthzDecisionStatement> message .......... 112

Figure 5.1. An example STD .......... 118

Figure 5.2. Completeness of the example STD .......... 120

Figure 5.3. The entire STD of the preparation stage of the protocol .......... 121

Figure 5.4. The entire STD of the negotiation stage of the protocol, while the parsimonious strategy is used ........ 122

Figure 5.5. The entire STD of the negotiation stage of the protocol, while the eager strategy is used ........ 123

Figure 5.6. All of the paths of STD of the protocol ........ 123

Figure 5.7. Two transitions from the initial state ........ 124

Figure 5.8. Transitions from state 1, when the parsimonious strategy is used ........ 126

Figure 5.9. Four possible transitions from state 4 ........ 128

Figure 5.10. Three possible transitions from state 5 ........ 129

Figure 5.11. One transition from state 6 ........ 130

Figure 5.12. Transitions from state 1, when the eager strategy is used ........ 131

Figure 5.13. A transition from state 8 ........ 131

Figure 5.14. Four transitions from state 9 ........ 132

Figure 6.1. The position of the database within the process of TN ........ 141

Figure 6.2. An algorithm for addressing the two vulnerability issues ........ 143

Figure 7.1. A category of case designs (Yin, 2013) ........ 152

Figure 7.2. A general overview of embedded single-case designs ........ 156

Figure 7.3. Possessed information in circumstance 1 ........ 163

Figure 7.4. Message 1 in the preparation stage in circumstance 1 ........ 164

Figure 7.5. Message 2 in the preparation stage in circumstance 1 ........ 165

Figure 7.6. Possessed information in circumstance 2 ........ 165

Figure 7.7. Message 1 in the preparation stage in circumstance 2 ........ 166

Figure 7.8. Message 2 in the preparation stage in circumstance 2 ........ 166

Figure 7.9. Possessed information in circumstance 3 ........ 167

Figure 7.10. Message 2 in the preparation stage in circumstance 3 ........ 168

Figure 7.11. Message 3 in the negotiation stage in circumstance 3 ........ 169

Figure 7.12. Message 4 in the negotiation stage in circumstance 3 ........ 169

Figure 7.13. Message 5 in the negotiation stage in circumstance 3 ........ 170

Figure 7.14. Message 6 in the negotiation stage in circumstance 3 ........ 170

Figure 7.15. Possessed information in circumstance 4 171

Figure 7.16. Message 5 in the negotiation stage in circumstance 4 172

Figure 7.17. Message 6 in the negotiation stage in circumstance 4 172

Figure 7.18. Possessed information in circumstance 5 173

Figure 7.19. Message 4 in the negotiation stage in circumstance 5 174

Figure 7.20. Message 5 in the negotiation stage in circumstance 5 175

Figure 7.21. Possessed information in circumstance 6 175

Figure 7.22. Message 4 in the negotiation stage in circumstance 6 176

Figure 7.23. Possessed information in circumstance 7 177

Figure 7.24. Message 5 in the negotiation stage in circumstance 7 178

Figure 7.25. Possessed information in circumstance 8 179

Figure 7.26. Message 5 in the negotiation stage in circumstance 8 180

Figure 7.27. Possessed information in circumstance 9 181

Figure 7.28. Message 5 in the negotiation stage in circumstance 9 182

Figure 7.29. Possessed information in circumstance 10 183

Figure 7.30. Message 6 in the negotiation stage in circumstance 10 184

Figure 7.31. Message 7 in the negotiation stage in circumstance 10 185

Figure 7.32. Message 8 in the negotiation stage in circumstance 10 185

Figure 7.33. Message 9 in the negotiation stage in circumstance 10 185

# List of Tables

Table 3.1. Interoperability issues between authorisation systems in Web Services ............ 66

Table 3.2. An improved multi-layered interoperability model ............ 70

Table 4.1. A conceptual multi-layered interoperability-solution design ............ 81

Table 5.1. An overview of states designed in the protocol ............ 121

Table 6.1. Conclusion of the effectiveness of the database design for policy-exchanged strategies ............ 144

Table 7.1. An overview of ten circumstances in the case scenario ............ 162

Table 7.2. Comparison between PERMIS, Akenti and the improved TN protocol ............ 164

Table 7.3. Comparison between TrustBuilder2 and the improved TN protocol ............ 171

Table 7.4. Comparison between Trust-X and the improved TN protocol ............ 174

Table 7.5. Comparison between Trust-Serv and the improved TN protocol ............ 180

# Acknowledgements

Towards the completion of my PhD degree, I would like to express my appreciation and thanks to my first supervisor Dr. Tanko Ishaya for his guidance and support throughout the first two and a half years of my PhD research project. In terms of guidance and support for the rest of the time of my PhD research project, I would like to express my sincere appreciation and gratitude to my second supervisor Dr. Darren Mundy. As the most important supervisor to me, he took over my PhD supervision responsibility after Dr. Tanko Ishaya left the University. Without Dr. Darren Mundy's knowledge and his patient, careful and prudent guidance, this Thesis might never have been completed and presented in this current version.

Special thanks also goes to the external examiner Dr. Mike Joy and the internal examiner Dr. Craig Gaskell for pointing out some critical issues within my PhD research and providing some helpful recommendations for improving the quality of my PhD research and the Thesis writing up.

Other persons deserving my appreciation include my office mates, who can always provide me new ideas through group discussions. My friends and family members in the UK and back home in Shanghai, China also offered me help when I felt frustrated, so thank you so much.

Taking this opportunity, I must thank my dad and my mum, my dearest friends Chung Kuen Ian Lau and Meiling Chen as well as their family members. Without their spiritual support, encouragement and prayers, I might not have been able to persist with my PhD research until the end.

Finally and most importantly, with respect to my hard work for the whole PhD research project, this Thesis is indeed a best reward.

# Abstract

The existing authorisation systems within the context of Web Services mainly apply two access control approaches – Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC). The RBAC approach links an authenticated Web Service Requester to its specific access control permission through roles, but RBAC is not flexible enough to cater for some cases where extra attribute information is needed in addition to the identity. By contrast, the ABAC approach has more flexibility, as it allows a Web Service Requester to submit necessary credentials containing extra attribute information that can fulfil the policies declared by a Web Service Provider, which aims to protect the sensitive resources/services.

RBAC and ABAC can only help to establish a unilateral trust relationship between two Web Services to enable a Web Service Provider to make an access control decision. Unfortunately, the nature of Web Services presents a high probability that two Web Services may not know each other. Therefore, successful authorisation may fail, if the Web Service Requester does not trust the Web Service Provider.

Trust Negotiation (TN) is also an access control approach, which can provide a bilateral trust relationship between two unknown entities, so it sometimes can enable authorisation success in situations where success is not possible through RBAC or ABAC approaches. However, interoperability issues will arise between authorisation systems within Web Services, where a bilateral trust-based authorisation solution is applied. In addition, a lack of a unified approach that can address the interoperability issues remains as a research problem. This research aims to explore possible factors causing the lack of interoperability first, and then to explore an approach that can address the interoperability issues. The main contributions of this research are an improved interoperability model illustrating interoperability issues at different layers of abstraction, and a novel interoperability-solution design along with an improved TN protocol as an example of utilising this design to provide interoperability between authorisation systems within Web Services.

# Chapter 1. Introduction

## 1.1 Background and Motivation

Traditional business transactions between companies required the processing of paperwork by employees. However, as human involvement in processing paperwork caused a significant business cost and resulted in low economic efficiency (Seacord, Plakosh and Lewis, 2003), an idea for automated business transactions without human intervention arose, which eventually gave birth to electronic data interchange. This technology was later standardised as Electronic Data Interchange (EDI) by the U.S. National Institute of Standards and Technology in 1996 (NIST, 1996). According to the formal definition of EDI, electronic messages used for business transactions must be expressed in a standard format, when they are exchanged between computers of companies as trading partners of each other. With the popularity of the use of EDI, a range of different EDI providers appeared. Unfortunately, as each provider defined their own standard format in their software, an interoperability issue developed amongst them that the software developed by one EDI provider could not understand the format used in other software developed by another EDI provider. This interoperability issue led to EDI-based business transactions becoming more expensive (Manes, 2003). For instance, if a company intended to run business transactions with different trading partners, and if the software used by each trading partner was purchased from different EDI providers, it would require the company to purchase software from multiple EDI providers. This process caused a substantial financial cost for EDI-based business transactions.

A preferable solution for addressing this interoperability issue is to enable companies to reach an agreement on the use of a standard format for communication. In 1998, such a standard format called Extensible Markup Language (XML) was developed. The structure of this standard language is flexible enough to be used in different business-transaction case scenarios. In addition, this language provides the flexibility to allow users to define their own syntax and structure of messages (Bray, Paoli and Sperberg-McQueen, 1998). To support the use of this standard language, a new technology named as Web Services was proposed as a successor of EDI (Locke, 2004).

When Web Services were first developed, Web application technology had been adopted within different kinds of organisations besides companies. As Web Service technology can be integrated into existing Web applications, the application of Web Service technology is not restricted to business companies any more, that is, other kinds of organisations such as governmental organisations (Alonso et al., 2009) or educational organisations (Skogsrud et al., 2004c) may also use Web Services. Therefore, Web Services can not only be used for electronic business (referred to as E-Business hereafter) between organisations (where EDI is mainly used for E-Business), but can also be used for providing other kinds of services between different organisations.

Use of Web Service technology allows an organisation to publish information about its resources to the Internet. Any other potential organisations supporting the use of Web Services can request access to the resources (Curbera, Nagy and Weerawarana, 2001). If a requested resource is not openly available, access control (the term “access control” and the term “authorisation” are treated as interchangeable words throughout this Thesis) is normally required to help a Web Service Provider decide whether a requested resource can be accessed. An access control decision is made on whether or not authorisation is successful. In the context of Web Services, there are two kinds of authorisation approaches. The first approach called Role-Based Access Control (RBAC) requires an involvement of successful authentication, and an explicit defined link between an authenticated identity and its access control permission. However, the access control decision made based on identity authentication may not be flexible enough to be used in cases where more information is required e.g. where sensitive credentials are present etc. (Bhatti et al., 2003; Wonohoesodo and Tari, 2004; Bhatti, Bertino and Ghafoor, 2004; Liu and Chen, 2004; Xu et al., 2004; Mohammad et al., 2011).

Apart from the scenarios mentioned above, the occurrence of a circumstance whereby a Web Service Requester and a Web Service Provider are unknown to each other is also possible. For example, if the Web Service Requester searches for a specific resource over the Internet, it may find the Web Service Provider by chance (Bellwood et al., 2004; Garofalakis et al., 2006). In this circumstance, making an access control decision requires a second authorisation approach called Attribute-Based Access Control (ABAC), as the first approach may not be feasible. With the use of this approach, the Web Service Requester will submit credentials according to the rules declared in the access control policies relevant to the resources provided by the Web Service Provider. Due to its flexibility, this authorisation approach is widely used within the current Web Services contexts (Yuan and Tong, 2005; Shen and Hong, 2006; Schlager et al., 2006; Mewar, Aich and Sural, 2007; Emig et al, 2007; Sabbari and Alipour, 2011; Paci et al., 2011; Zhang et al., 2014).

The use of the ABAC approach relies on an assumption that the Web Service Requester is willing to disclose all of the credentials required in the policies declared by the Web Service Provider. However, this is not always the case. If some of the required credentials are treated as sensitive, then the Web Service Requester may not be willing to disclose them to the unknown Web Service Provider, as it does not trust the Web Service Provider. This is because the trust relationship established by using the ABAC approach is unilateral. This phenomenon will result in failed authorisation, where successful authorisation may be possible (Yu, Winslett and Seamons, 2001; Winslett et al., 2002; Hess et al., 2004; Frikken, Li and Atallah, 2006; Mbanaso et al., 2006; Winsborough and Li, 2006).

Fortunately, Trust Negotiation (referred to as TN hereafter) as another access control approach proposed by Winsborough, Seamons and Jones (1999, 2000) can address the lack of bilateral-trust establishment in authorisation. This TN approach can help two unknown entities establish a bilateral trust relationship, which can enable the Web Service Requester to trust the unknown Web Service Provider. Sensitive credentials possessed by the Web Service Requester that cannot be disclosed at early stages, may be disclosed at later stages based on an established trust level. This establishment of a trust level can help to convert authorisation failure into authorisation success in some circumstances (Winsborough and Li, 2002a; Winsborough and Li, 2002b). Shen and Hong (2006) identify that TN can be more powerful and flexible than ABAC in terms of successful authorisation achievements within Web Services.

## 1.2 Problem Statement, Research Aims, Questions and Objectives

### 1.2.1 Research problems

At present, there is a multitude of authorisation systems used in Web Services. These authorisation systems can support the use of the RBAC or ABAC approaches, but in each system, the syntax and semantics of languages for expressing credentials and policies are different (Lang et al., 2006). This phenomenon may cause an interoperability issue that the Web Service Provider may not be able to understand the credentials submitted by the Web Service Requester. For instance, if the syntax and semantics of credentials used by the Web Service Requester are unknown to the Web Service Provider, the Web Service Provider is not able to compare the credentials against its local policies.

The TN approach is superior to the RBAC and ABAC approaches due to its unique benefit of the provision of the establishment of a bilateral trust relationship between two unknown entities. In addition, a number of TN-based authorisation systems are also available, but to use these systems within Web Services also produces different interoperability issues.

In some circumstances, the above specific interoperability issues could have been addressed between authorisation systems (i.e. ABAC-based and TN-based) within Web Services, so that potential successful authorisation could have been achieved. Unfortunately, as there is no existing approach that can be used to resolve interoperability issues between authorisation systems within Web Services, potential successful authorisation may eventually fail.

In conclusion, the specific research problems are:

1. Lack of a comprehensive understanding of indeterminable factors causing interoperability issues may weaken the effectiveness of solutions for addressing interoperability issues between authorisation systems within Web Services.

2. There is no unified approach that can address interoperability issues in relation to multiple factors so that potential successful authorisation between authorisation systems within Web Services may fail.

### 1.2.2 Research questions

To enable the identification of a solution for addressing the above research problems, the below research questions are defined for this Thesis:

1.What are the factors that cause interoperability issues between authorisation systems within Web Services?

2. How can a unified approach address interoperability issues caused by the identified factors to ensure that potential successful authorisation between authorisation systems within Web Services will not fail?

### 1.2.3 Research aims

To address the research problems as stated above, this research aims to:

- Explore the current state-of-the-art access control approaches within Web Services and the relevant key factors that may cause interoperability issues between authorisation systems within Web Services;
- Explore a unified approach that can help to deliver potential successful authorisation, if the interoperability issues caused by the relevant key factors can be addressed.

### 1.2.4 Objectives

To accomplish the research aims, the specific objectives of this research are listed as follows:

- Review the existing RBAC and ABAC approaches used in Web Services, and analyse their characteristics to understand how they are used within Web Services;
- Review the existing interoperability models to understand the identified interoperability issues and explore any potential new interoperability issues between systems;
- Review state-of-the-art TN-based authorisation systems to assess the main functionalities of their key components;
- Explore a potential approach that can address interoperability issues for authorisation systems within Web Services;

- Evaluate the potential approach to identify whether it can enable potential successful authorisation not to fail due to the interoperability issues;
- Conclude the findings through the evaluation and explore whether there are potential limitations within the solution.

## 1.3 Research Methodology and Methods

The research aims and objectives have been stated above. Towards accomplishing the aims and detailed objectives, an appropriate research methodology should be selected before the conduction of the research. As stated in Kumar (2008), a research methodology is a systematically planned, organised process that can direct a researcher to accomplish the specific research goal. A similar point of view is found in Kothari (2009) that a research methodology can guide a researcher to carry out a research in a scientific way. A researcher not only needs to understand how to use different research methods to identify answers for research questions, but also needs to have a very clear mind on the reasons why the selected research methods are of the most relevance to the research. This opinion is also supported by Jonker and Pennink (2010), who treat a research methodology as a kind of "action repertoire". This repertoire is designed based on the premises, considerations and practical conditions strictly relevant to the research. According to this repertoire, a researcher is able to justify the logic for selecting the most appropriate research methods in each stage during the research. With the guidance of this strong logic, it is believed that the research findings are able to answer the research questions, and the potential research solutions are of the most appropriateness with addressing the research problems.

A wide range of research methodologies are available, however, it is often best to select a methodology that corresponds most closely with the researcher's basic philosophy. Normally, the choice of a methodology is related to the research methods, and the choice of the most appropriate research methods within a piece of research or a research project is in turn related to the nature and the way specific research questions are asked and how a researcher wants to answer them (Jonker and Pennink, 2010). In this research, as the nature of the research questions is related to the identification of unique characteristics of the research objects and the exploration of a potential solution, an empirical methodology along with qualitative data analysis is regarded as the most appropriate research methodology.

To carry out this research, the research methods selected for each stage are listed as follows.
The state of the art of research in the areas of interest related to this study could be evaluated through theoretical methods. To enable the researcher to obtain and analyse the relevant information, literature review, more precisely, the documentation analysis (Taylor et al., 2006) was treated as one of the most appropriate research methods suitable to this step for exploring information in relation to the ABAC-based and TN-based authorisation systems within Web Services and that in relation to existing interoperability models. In addition, the case study method (Yin, 2013) was used along with documentation analysis for verifying the existence of discovered interoperability issues between authorisation systems in practice.

Based on the analysis of the data obtained using the two research methods, an improved multi-layered interoperability model illustrating multiple layers at different levels of abstraction of interoperability issues between authorisation systems within Web Services was constructed. This improved interoperability model can provide an understanding of the existence of multiple layers of interoperability, but cannot supply any propositions for potential solutions relevant to each layer. Through a documentation analysis of the review of protocols, it was identified that a protocol-based approach might be the most appropriate solution for providing interoperability for the majority of the layers. Therefore, this improved interoperability model was extended to a novel conceptual multi-layered interoperability-solution design, which proposes how to use a protocol-based approach for the provision of interoperability for each relevant interoperability layer.

Following the guidance of the interoperability-solution design, an improved TN protocol was then created as an example of utilisation of the interoperability-solution design. This protocol was designed and developed following a widely used protocol design and development methodology named validation (Merlin, 1976; Bochmann and Gecsei, 1977; Merlin, 1979). A formalism called Finite State Machine was used for verifying the completeness of all possible states designed in the protocol (Bochmann and Gecsei, 1977; Bochmann, 1978; Sunshine, 1979a; Sunshine et al., 1982).

After conducting this completeness test, two intrinsic flaws were explored within this protocol. As there was no complete solution provided by the state-of-the-art TN-based authorisation systems for addressing them, a conceptual solution design based on the idea of “remembrance of local information status” was created. The identification of the key reasons causing the occurrence of the two flaws was through a critical analysis of the relevant review and typical representative case studies. This solution was evaluated to be an effective complementary solution for the protocol to some extent in the case studies for evaluation. With the addition of this solution to the protocol, its completeness was ensured. Finally, a detailed evaluation of the effectiveness of the proposed interoperability-solution design including a protocol correctness test was presented, wherein the case study research method along with a model-based testing (Utting, and Legeard, 2006; Kull, 2009) were selected as the most appropriate evaluation methods. The evaluation result could demonstrate the effectiveness of the proposed solution design as well as the correctness of the improved TN protocol.

## 1.4 Scope and Limitations of the Research Contributions

In terms of research contributions, there are four in this Thesis. The first contribution is an improved multi-layered interoperability model. This model aims to clarify the existence of multiple layers of interoperability issues between authorisation systems (i.e. ABAC-based or TN-based) within Web Services. It might also be used to illustrate interoperability issues between systems in other distributed systems environments.

The second contribution is a novel conceptual multi-layered interoperability-solution design, which specifies how to use a protocol-based approach to provide interoperability for the majority of the interoperability layers identified in the improved interoperability model as the first contribution. This interoperability-solution design can provide guidance aiding protocol developers in addressing interoperability issues, when they design and develop protocols. However, limitations of this interoperability-solution design still exist, in which an interoperability issue at the highest layer cannot be completely resolved by the protocol-based approach recommended by this interoperability-solution design. In the research of this Thesis, this interoperability-solution design is leveraged for guiding the design of an improved TN protocol as a concrete example for addressing interoperability issues at specific layers.

The third contribution is an improved TN protocol, which can enable an ABAC/TN-based authorisation system to communicate with another ABAC/TN-based authorisation system without specific interoperability issues (e.g. functional, capability, strategy) within Web Services. In addition, this protocol can deliver successful TN-based authorisation, if multiple credential and policy languages designed within the existing ABAC/TN-based authorisation systems are used in the context of Web Services. In other words, it can raise the success probability for using TN in some circumstances. However, there are limitations within this protocol, where the novel mechanism designed in this protocol can only work, when two Web Services have a common capability for processing language combinations and have a common strategy.

The fourth contribution is a solution design to address intrinsic vulnerability issues within the proposed TN protocol. This solution design can also be utilised to protocols designed within the state-of-the-art TN-based authorisation systems, when policy-exchanged strategies (see section 3.3.1.2) are used. In addition, this solution design might also be applied to resolve variations of Denial of Service (DoS) attacks. There are limitations within this solution, so that it is not effective for all of the policy-exchanged strategies designed for TN.

## 1.5 Thesis Outline

This section outlines the structure of this Thesis to give a reader an initial idea about the relationship amongst the chapters.

Chapter 2 presents a definition of Web Services. Security-related standards and authentication services used within Web Services are then introduced, as they are relevant to the use of authorisation. Following that, this chapter describes the details of two existing authorisation approaches (i.e. RBAC and ABAC) used within Web Services along with a conclusion of their merits and limitations. Existing ABAC-based authorisation systems are then introduced along with the way they are used within Web Services. In addition, benefits and limitations of the use of ABAC-based authorisation systems in Web Services are discussed. Within these review, interoperability issues are identified.

In Chapter 3, the concept of TN is presented associated with its general process. A detailed review and analysis of TN is then provided for identifying the necessary components of TN followed by a discussion of state-of-the-art TN-based authorisation systems. Possible interoperability issues between ABAC/TN-based authorisation systems are discussed. After the review, an embedded case study is presented for verifying the aforementioned interoperability issues in the research context. Existing interoperability models of the most relevance are also reviewed. Following a critical analysis of the review, an improved conceptual multi-layered interoperability model is constructed, which aims to address the first research problem. In addition, work in relation to other existing interoperability models of less relevance is discussed to distinguish the differences between them and the improved interoperability model. The second research problem is then presented followed by a discussion of potential solutions.

Chapter 4 first proposes a conceptual multi-layered interoperability-solution design aiming to address the second research problem. This interoperability-solution design provides guidance of how a protocol-based approach can provide interoperability for the majority of the interoperability layers. A protocol design and development methodology is then introduced. Following this methodology, elicited protocol requirements are listed based on a critical assessment of the feasibility of the adoption of state-of-the-art TN techniques within Web Services. A proposed protocol providing a TN-based authorisation service is then designed strictly according to the elicited requirements and the interoperability-solution design. This protocol specifies the details of the logical processes and the relevant syntax and semantics of communication messages in order to provide interoperability between authorisation systems within Web Services.

In Chapter 5, a protocol verification method is introduced for verifying the completeness of the proposed protocol. An entire verification process is then clarified.

During the process of this verification method, two intrinsic flaws within the protocol are discovered.

Chapter 6 presents a conceptual solution design based on the idea of “remembrance of local information status” to address the two intrinsic flaws explored in the protocol completeness test. The realisation of this solution design through the relational database technology and the relevant evaluation result analysis are detailed.

Chapter 7 provides a detailed discussion of utilising the case study evaluation method for evaluating whether the designed proposed protocol following guidance of the interoperability-solution design can effectively address the second research problem. Effectiveness and limitations of the protocol are discussed, and differences between the proposed protocol and related work are clarified.

Finally, Chapter 8 is a conclusion chapter, which presents a discussion on the process of the entire research including the research problems identified in Chapter 1, review and case study in Chapter 2 and Chapter 3, the process of the conceptual protocol design in Chapter 4 and the protocol verification process in Chapter 5, the solution design in Chapter 6 and the protocol evaluation process in Chapter 7. It also discusses the contributions along with their impacts followed by potential future work.

## 1.6 Chapter Summary

This chapter has introduced the background information and motivation for the research carried out in this Thesis. It has also articulated the research problems, research questions, aims and specific objectives. A relevant appropriate research methodology and methods have been justified and linked to the potential achievement of the research goals. In addition, the scope and limitations of the research contributions have been outlined. The next chapter presents a detailed review in the field of Web Services with a particular focus on identifying possible interoperability issues between ABAC-based authorisation systems within Web Services.

# Chapter 2. Review of Security-Related Standards, Authentication Services, RBAC, ABAC Approaches and the Relevant Authorisation Systems Within Web Services

## 2.1 Introduction

As stated in Chapter 1, this research focuses on exploration of a potential solution for addressing interoperability issues between authorisation systems within Web Services. Identification of such a solution initially requires the researcher to have an overall understanding of the interoperability issues. Chapter 2 and Chapter 3 together introduce a thorough investigation of the relevant literature. This investigation critically assesses the state of the art in various fields (Web Services, interoperability, RBAC, ABAC approaches along with the relevant authorisation systems and TN (discussed in Chapter 3)). All of the relevant factors along with their characteristics causing the interoperability issues (highlighted in bold and italic) within the research context are explored through the analysis of the relevant review.

In the areas of Web Services, interoperability, RBAC, ABAC approaches and TN, there is already a vast array of pre-existing research. Therefore one of the major challenges to this Chapter and Chapter 3 is determining the research of most relevance to the research problems within this Thesis, and presenting this in a logical order. After thoughtful consideration, the review begins with the clarification of a definition of Web Services. Based on the discussion of this definition, the key issue existing within Web Services, that of interoperability is analysed in depth. Through a review of security-related standards (authorisation is related to security), authentication services (authentication is closely related to authorisation) within Web Services, an understanding about what services they can provide is formulated. Based on the analysis of this understanding, possible interoperability issues and the relevant solutions within these protocols are identified.

Due to the difference between the RBAC and ABAC approaches and their application within the relevant authorisation systems, the review in relation to the use of ABAC-based authorisation systems within Web Services is further explored. This includes a critical analysis of the authorisation services they can supply and an assessment of their suitability. In addition, a discussion of the existing solutions providing interoperability for ABAC-based authorisation systems used within Web Services is also presented to point out their possible interoperability issues. As there exists a large amount of research related to TN, a decision is made to present the relevant review in Chapter 3 for easy discussion and clarification.

## 2.2 Definition of Web Services

The reason for the initial emergence of Web Services is stated by Christensen et al. (2001), "Web Services are a natural consequence of the evolution of the Web into an open medium which facilitates complex business and scientific application interactions". In other words, the advent of Web Services is attributed to the demand for achieving automated E-Business in electronic markets, providing higher efficiency and increased profits with the help of Web-based application-to-application interactions (Zhao, 2006; Davis and Vladica, 2007). From the perspective of E-Business, it is not surprising that the majority of developers initially treated a Web Service as a resource to be consumed by software rather than by humans (Manes, 2003). There are multiple distinct perspectives on the essence of Web Services, for example, one is that Web Services should be treated only as messaging technologies (Vogels, 2003); whilst another suggests that Web Services provide the packaging strategy for business logic (Shah and Apte, 2004), thus suggesting an emphasis on the inclusion of business processes. Thankfully, a formal definition of Web Services is stated in Web Services Architecture (Booth et al., 2004), which presents a general architecture for the utilisation of Web Services.

*A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

As a concise definition, it introduces the general characteristics of Web Services, but it is not straightforward enough to express various key points underpinning the use of Web Services. To highlight these key points, there is a need to expand this definition as described as follows: Web Service technologies allow software systems to be deployed as Web Services, which are able to interact with one another with the prerequisite of interoperability. Interaction or communication between two Web Services normally occurs over a network (e.g. intranet, extranet etc.). In particular, if they are located in different intranets, their communication has to occur over the Internet. An interface of a Web Service written in the WSDL, a machine-processable format, (details of WSDL are discussed in section 2.4.4) operates as an entrance to retrieve an incoming SOAP message (details of SOAP are discussed in section 2.4.3) sent from another Web Service. The Web Service should be able to deal with the content or data contained within the message in accordance with the rules indicated by the descriptions of the content. The most commonly used delivery method for transmitting a message over the network from one Web Service to another Web Service is HTTP (stands for Hypertext Transfer Protocol), which is an application protocol for distributed systems (Fielding et al., 1999). In addition, other application protocols such as SMTP (stands for Simple Mail Transfer Protocol) etc. are also available for transmitting a SOAP message (Howard, 2001; Mitra and Lafon, 2007). The content of the message contained in the SOAP envelope should conform to other Web-related standards that are presented in an XML-based structure (details of XML are discussed in section 2.4.2) for serialisation.

As discussed in the expanded definition above, the most significant feature is that the communication between Web Services should be interoperable with the help of a combination of technologies such as machine-processable interfaces, rules indicated by the descriptions of messages, and support for cross-platform application-layer protocols etc. From the perspective of interoperability, a Web Service should neither be treated as only a resource, nor as only messaging technologies or strategies for business logic. Instead, it is preferable to regard Web Services as a technology-based framework for the guidance of system designs, ensuring their interoperability in communication, and ability not only to support e-business, but also to accomplish large-scale resource sharing (Foster, Kesselman and Tuecke, 2001). This focus on the technology distinguishes Web Services from another term referred to as Service Oriented Architecture (SOA), as the focus of SOA is on the architecture (Mances, 2003; Rosen et al., 2008; Papazoglou, 2012), which mainly aims to achieve the provision of a specific business service (Channabasavaiah, Tuggle and Holley, 2003).

The prospect of using Web Services is appealing, in that a Web Service Requester can communicate with any other Web Service Provider for accomplishing specific executions (e.g. resource sharing, business transaction etc.) over a network. However, it does not mean that Web Service technologies are suitable for all cases. To guide the appropriate use of Web Services, four cases are suggested by Booth et al. (2004) that Web Services can be the selected implementation mechanism for applications (shown as follows):

- *"That must operate over the Internet where reliability and speed cannot be guaranteed;*
- *Where there is no ability to manage deployment so that all requesters and providers are upgraded at once;*
- *Where components of the distributed system run on different platforms and vendor products;*
- *Where an existing application needs to be exposed for use over a network, and can be wrapped as a Web service. "*

Having discussed the definition of Web Services, the next section presents a detailed clarification of the interoperability issues within Web Services based on an analysis of the expanded description of the definition.

## 2.3 Interoperability Issues and Protocols

Observing the first sentence of the expanded explanation of the formal definition, the key point that needs to be highlighted is the concern in relation to interoperability, which is the prerequisite to ensure that two Web Services are able to communicate with each other. Thus, the assurance of interoperability for Web Services always needs to be taken into consideration.

The IEEE provides four definitions for the term interoperability "(1) the ability of two or more systems or elements to exchange information and to use the information that have been exchanged, (2) The capability for units of equipment to work together to do useful functions, (3) The capability, promoted but not guaranteed by joint conformance with a given set of standards, that enables heterogeneous equipment built by various vendors, to work together in a network environment and (4) The ability of two or more systems or components to exchange information in a heterogeneous network and use that information." (IEEE press, 2000).

Through an analysis of the above definition, Diallo (2010) identifies that "the goal of interoperability is to exchange useful information". In other words, interoperability is embodied when exchanged information between communicating systems can be understood and processed by each other for achieving specific purposes in a certain context. As this point of view can point out the key feature of interoperability, it is used as a main criterion for assessing interoperability between systems throughout this Thesis.

To ensure interoperability between communicating systems, the use of protocols has been identified as the best solution (Rezaei, Chiew and Lee, 2014). For instance, with respect to the history of distributed systems, the relationship between protocols and distributed systems interoperability was first mentioned in Merlin (1979), "Distributed systems naturally employ protocols because if the interacting entities are physically remote to each other, message exchange is the only possible way of coordinating their activities". This statement points out that the basic interoperability issue between distributed systems is due to the lack of communication, which can only be achieved through message exchange as defined within protocols. Foster et al. (2001) point out the relationship between protocols and interoperability, "A protocol definition specifies how distributed system elements interact with one another in order to achieve a specified behaviour, and the structure of the information exchanged during this interaction". Since Web Services are one kind of distributed system

(Glass, 2001; Josuttis, 2007), it is a natural result that the interoperability issues also exist in this context, and the use of standardised protocols is a clear solution. This opinion is demonstrated in the informal definition of Web Services proposed by Christensen et al. (2001), "A Web Service is a networked application that is able to interact using standard application-to-application Web protocols over well defined interfaces, …". In addition, the opinion that the use of protocols is the only approach in enabling any two Web Services to communicate is also emphasised by Ballinger et al. (2004).

Researchers have written a number of different definitions describing what a protocol is or delivers. As a result, there also exist different perspectives on what interoperability issues can be addressed by the use of a protocol (see section 3.5). West (1978) presents an initial definition of a communication protocol, in that "a communications protocol may be defined as the set of rules that govern the exchange of information between processes in a communications system." This definition is accepted by researchers such as Gouda and Manning (1976); Danthine and Bremer (1978). However, arguments have occurred over this definition. The arguments point out that the definition proposed by West (1978) only focuses on the general rules for specifying the order of the exchanged messages at an abstract level. Apart from this level, a detailed level specifying how each message should be processed by systems was also important, so this detailed level should be included within the concept of a protocol as well (Sunshine, 1979a; Bochmann and Sunshine, 1980). This opinion has been treated as the most appropriate perspective taken within this Thesis, since it supplies a comprehensive level of understanding of the characteristics of a protocol at both the abstract level and detailed level. Thus, the notion of a protocol adopted within this Thesis consists of two parts: **service specification** and **protocol specification**. A service specification is a general abstract description of what service a specific protocol can provide. It can include the general description of an input message and the relevant output messages. Rules of dealing with the input message along with the generation of the output message can also be given. Developers are normally the main readers for whom the service specification of a protocol is provided. They need to understand what service the specific protocol can provide to determine whether or not they need to implement the protocol within their systems. A protocol specification is further twofold: **protocol messages** and **internal structures**.

A protocol message includes the definition of the syntax and semantics used within a message. It also specifies the specific data contained in the message for communication. The internal structure details the logical process expanding from the rules that are generally described in the service specification.

With the adoption of the notion of a protocol, it is easier to decompose a protocol into three sub components for analysis. A simple example is helpful to illustrate the usefulness of this notion. If there is a protocol allowing a Web Service Requester to request the weather information from a Web Service Provider (Paolucci and Sycara, 2003), the service specification presents the general information of the protocol at an abstract level including: (1) the service provided by the protocol is to enable the Web Service Provider to return the weather information to a Web Service Requester, which has sent a request message to the Web Service Provider and (2) an output message containing the weather information should be returned, if the received input message is an understandable request for the weather information. Otherwise, an error message should be returned instead. However, the task of how these messages should be presented (syntax and semantics of the message) is not the responsibility of the service specification, but is achieved by the protocol messages (as the first part of the protocol specification). Although the service specification describes the general rules about which output message should be returned in correspondence with the input messages, the detailed process of dealing with the input message to generate a relevant output message is not included within it. Therefore, the clarification of the detailed process is the responsibility of the internal structure (as the second part of the protocol specification).

With the construction of a precise understanding of the protocol term, it is useful to use this understanding in the analysis and assessment of the design of existing authorisation protocols within Web Services as well as their interoperability. However, due to the nature of Web Services, understanding the nature of underlying Web Services protocols along with other standard specifications used can facilitate the comprehension of existing authorisation protocols. As these standard specifications and protocols are published by official organisations, the next section starts with an introduction of the official organisations followed by the review of the relevant standards.

## 2.4 Security-related Standards, Authentication Services Within Web Services

### 2.4.1 Official consortiums and protocol standardisation

Basically, it is inconvenient to develop a throwaway protocol for two Web Services, when they need to communicate with each other. This would result in the potential existence of multiple protocols providing the same or similar services causing confusion as to which protocol to use between any two Web Services. This in turn may also mean that developers would need to implement these different protocols within Web Services. An ideal approach is to standardise a proposed protocol providing the specific service used by Web Services (Fensel and Bussler, 2002; Naedele, 2003; Benatallah et al., 2005; Wang et al., 2004; Roman et al., 2005).

To enable a standard protocol to be publicly accessed in the world, two official consortiums – World Wide Web Consortium (W3C), and the Organisation for the Advancement of Structured Information Standards (OASIS) – have been established. They are responsible for the ratification of standards used for Web Services (W3C, 2012; OASIS, 2012; Fensel and Bussler, 2002; Naedele, 2003).

### 2.4.2 XML and Schema

Through the protocol standardisation approach, interoperability issues in relation to ***syntax*** and ***semantics*** of communication messages arise. Two platforms (i.e. J2EE, .NET) are mainly used for the development of Web applications. Programming languages used in J2EE may not be recognised by .NET (Mallalieu and Carriere, 2004; Microsoft, 2004). As the prospect of using Web Services is to enable any two systems available on the Internet to communicate with one another regardless of their specific platforms, a common language independent of any specific platform is required. To meet this requirement, a standard language called Extensible Markup Language (XML) was proposed and published by W3C in 1998 (Bray et al., 1998), and XML can bring Web Services a platform-independence benefit (Ho and Yen, 2005; Guruge, 2004).

Although XML has been used as the *lingua franca* within Web Services, and recognised by multiple platforms, it does not mean that there are no weaknesses within it. On the contrary, the controversy over whether or not to use XML has never stopped due to its verbosity and a requirement for intensive parsing. These limitations largely affect the performance of using XML (Beznosov et al., 2005; Pallis, Stoupa and Vakali, 2008). Nonetheless, reasons supporting the use of XML are given by Christensen et al. (2001), "[V]erboseness actually becomes one of its greatest strengths when enabling communication between diverse sets of systems. XML's clear representation of structured data makes it an ideal foundation for the Web services framework." Upon the acceptance of the use of XML in Web Services, research has been undertaken to identify ideal approaches for improving the effectiveness and efficiency by using XML (Zhang and Engelen, 2008; EI-Bakry and Mastorakis, 2009; Tere and Jadhav, 2010).

To enable the XML to be able to define the syntax of messages, schema (short for XML schema) was proposed by W3C (Brown et al., 2001). The presentation of schema is still an XML-based structure, but the syntax of messages can be defined easily with the predefined structures within schema. Therefore, with the use of the XML and schema, the interoperability issue in relation to ***syntax*** or structure of communication messages within Web Services can be addressed. At the same time, the XML and schema together form the cornerstone of the protocols and specifications designed within Web Services. XML and schema are capable of structuring messages in a clear way, but are unable to express meanings of messages. In other words, incapability of XML and schema enabling a Web Service automatically to understand the ***semantics*** of messages is still a weakness. Current solutions for making up for this weakness are to present semantics in a natural language and require an involvement of human developers for understanding natural language-based semantics (Klein, 2001).

### 2.4.3 SOAP

Simple Object Access Protocol (SOAP) was the first member of the Web Services triumvirate (the other two members are WSDL and UDDI, which are discussed in section 2.4.4 and 2.4.5 respectively), designed and published by W3C (Mitra and Lafon, 2007). SOAP, as a *de facto* protocol, provides its specific service specification, and its protocol specification including the protocol messages (specifying the syntax and semantics of the general communication messages exchanged between Web

Services), and the internal structure (dealing with the data in correspondence with their semantics). For instance, by reading the service specification, one can understand that a SOAP message is designed as an envelope (called SOAP envelope) including two parts: header and body. The header part can contain the information such as addressing, which is specified in the WS-Addressing specification (Gudgin, Hadley and Rogers, 2006). The body part can contain the information that needs to be processed by the Web Service, which should be the correct SOAP message receiver. Within a SOAP message, multiple header parts can occur, but only one body part is allowed to exist. Following the defined syntax and semantics of the protocol messages within its protocol specification, knowledge including the meaning of the <Envelope>, <Header> and <Body> messages as well as their structures can be obtained. The implicit internal structure within the descriptions of each message explains the process of how a Web Service should deal with them e.g. how an intermediate Web Service should deal with a SOAP message when reading the <Header> message (Curbera, Nagy and Weerawarana, 2001; Curbera, et al., 2002).

Through an analysis of the SOAP protocol, it can be identified that this protocol is designed only to provide communication in the form of a Web Services-related message. Its use in Web Services is analogous to the use of an envelope and a blank piece of paper for posting a letter in reality. From this perspective, the SOAP protocol can only provide interoperability in relation to ***syntax*** and ***semantics*** of the information in the header part (e.g. which Web Service initiates this SOAP message and which Web Service is supposed to receive this SOAP message) and the relevant ***functionality*** for processing the information (e.g. an intermediate Web Service should transmit this SOAP message to an endpoint Web Service, which is the target receiver). The reason is straightforward that this protocol is designed to be independent of any specific services, as providing interoperability in relation to ***syntax***, ***semantics*** and ***functionality*** for any other specific-services-related communication is the responsibility of other protocols, which are designed to provide the relevant services. These protocols are embedded within the header or body part of the SOAP protocol analogous to the reality that a person is allowed to write any contents on blank paper. To ensure the seamless integration of other protocols, the SOAP protocol provides for extensibility due to the benefit of using XML (Chumbley et al., 2010).

### 2.4.4 WSDL

As the use of different protocols along with the SOAP protocol can provide interoperability in relation to ***syntax***, ***semantics*** and ***functionality*** for different specific-services-related communication, it means that two Web Services are able to communicate for performing specific services as defined in the service specification of the protocols. This phenomenon implicitly indicates that the Web Service Requester has ensured that the services (a service can be a specific service or a specific resource) provided by the Web Service Provider are the correct services it wants. This assurance is held after the Web Service Requester has read the relevant service descriptions provided by the Web Service Provider, and before its communication with the Web Service Provider. In Web Services, such a service description is expressed by using the standard called Web Services Description Language (WSDL) published by the W3C (Christensen et al., 2001).

WSDL specifies ***syntax*** and ***semantics*** of a WSDL file, which allows a Web Service to describe its services in a machine-processable format. With the use of a WSDL file, a Web Service can consist of a collection of separate services. Since a separate service is generated by means of binding an abstract definition to a port type, the change of an internal structure of a separate service will not affect the change of the relevant abstract definition. Such a characteristic called loosely coupling becomes another benefit for Web Services (Woods and Mattern, 2006; Pautasso and Wilde, 2009).

A weakness of using WSDL still exists that human intervention for understanding the ***semantics*** of a WSDL file of a Web Service Provider is needed. The WSDL file cannot convey the ***semantics*** of information and an ontology is not widely used to help adequately determine the meaning of the WSDL for a Web Service (Lewis and Wrage, 2006). This problem occurs particularly when developers on the Web Service Requester side want to ensure that their Web Service Requester can properly communicate with the Web Service Provider (Paolucci and Sycara, 2003; Nezhad et al., 2006).

### 2.4.5 UDDI

With the help of the WSDL file, the Web Service Requester can compare the Web Service Provider's service information with its own query to decide on whether they are the correct services it desires. Currently, there are two existing approaches available for such comparison as discussed in Garofalakis et al. (2006). The first approach is the catalogue/keyword-based method. By using this method, the Web Service Requester compares the keywords within its query to match against the service descriptions of the Web Services Provider. However, this method may cause a Web Service Requester to miss the opportunity to find a Web Service Provider that provides the correct service but has different keywords used in its service descriptions. To make up for this drawback, an ontology-based approach is proposed to enable a Web Service Requester to perform semantic-based matching (Ankolekar et al., 2002).

The conduction of the service comparison of a Web Service Requester is based on an assumption that the Web Service Requester has discovered such service descriptions on the Internet. This requires an approach to enable a Web Service Requester to discover a Web Service Provider, through detailing the services it provides on the Internet. To fill this gap, the open standard named Universal Description, Discovery and Integration (UDDI) was proposed and published by OASIS (Bellwood et al., 2004).

UDDI is designed as a form of yellow pages (directory) to enable a Web Service Requester to look up the address of a Web Service Provider. With the use of UDDI, three approaches including: (1) Registries, (2) Index approach and (3) Peer-to-Peer (P2P) Discovery can be used to allow a Web Service Requester to discover a potential Web Service Provider that provides the correct services (Booth et al., 2004).

Although the use of UDDI at a theoretical level has been accepted, its real utilisation is not popular. Nezhad et al. (2006) point out that the majority of platforms can support UDDI, but UDDI is not widely used. This implies a current situation that the use of Web Services nowadays is still between two Web Services that are known to each other. This may be the reason why approaches that can help two unknown Web Services establish a strong bilateral trust relationship are not widely used in the context of Web Services. However, it is believed that with the popularity and maturity of Web Service technologies, especially UDDI (Garofalakis et al., 2006), the probability of communication between two unknown Web Services will increase in practice. Therefore, identifying such an approach at an early stage for Web Services is necessary.

### 2.4.6 XML-Encryption and XML-Signature

Through the use of various standards including SOAP (along with different protocols), WSDL and UDDI, Web Services can now communicate with one another. Unfortunately, these standards cannot provide any security-related protection for transferred messages, when two Web Services belonging to different security domains need to communicate over the Internet (Imamura, Dillaway and Simon, 2002; Bartel et al., 2008). Example security-related protection includes the need for necessary security requirements such as confidentiality, integrity, non-repudiation, authentication (the detailed concept is discussed in section 2.4.7) and authorisation (the detailed concept is discussed in section 2.4.11). Cryptography has been used to provide for confidentiality and integrity, and the use of digital signatures has been used for supplying non-repudiation (Schneier, 1995; Adams and Lloyd, 1999; Boncella, 2004; Rowan, 2005; Mahmoud, 2005; Steel, Nagappan and Lai, 2005).

Interoperability issues in relation to ***syntax*** and ***semantics*** arise, when applying these approaches (i.e. cryptography and digital signature) to Web Services. For instance, if a Web Service Requester sends out a SOAP message, information contained within this SOAP message will include an encrypted message, the name of the encryption algorithm, and the relevant key encrypted using the other service's public key to another Web Services Provider. To decrypt the encrypted message, a Web Service Provider must understand three points obtained from the SOAP message including: (1) The value representing the encrypted message, (2) The value representing the key and (3) The value representing the name of the cryptography method. The Web Service Provider is only able to run the correct cryptography method (since a variety of different cryptography methods do exist) to decrypt the encrypted message with the key by understanding all of the three points. The only approach to enable the Web Service Provider to achieve these tasks is that it can obtain these points from the message by understanding its ***syntax*** and ***semantics***. This prerequisite is similar to situations when there is a need to verify digital signatures. To address the issues, XML-Encryption and XML-Signature were developed and published by W3C respectively (Imamura, Dillaway and Simon, 2002; Bartel et al., 2008). XML-

Encryption details the ***syntax*** and ***semantics*** of the messages that enables a Web Service to explicitly inform another Web Service which encryption methods (e.g. symmetric and asymmetric cryptography methods) along with the encrypted key it has used to encrypt the data. Similarly, XML-Signature specifies the ***syntax*** and ***semantics*** of the messages for Web Services to state the specific digital signature method associated with the digital signature used within a SOAP message.

### 2.4.7 WS-Security

In addition to the security requirements mentioned in section 2.4.6, authentication is also treated as one of the most important security requirements (IBM and Microsoft, 2002; Boncella, 2004; Rowan, 2005; Mahmoud, 2005). Authentication is to verify the identity of either a user or a service requester. To achieve authentication, technologies such as cryptography and digital signatures are needed, but are not adequate. Traditionally, a password-based authentication method is regarded as a major authentication method, but weak password design by users is its main drawback. Furthermore, it is suitable to be used in a user-system interaction, but not suitable in a system-system interaction (Smith, 2001; Burnett and Kleiman, 2005). To provide for a stronger authentication in a system-system interaction, a new mechanism called Public key Infrastructure (PKI) is accepted as a preferred approach (Adams and Lloyd, 1999; CGI, 2004).

Generally, PKI is a technical combination of the utilisation of symmetric, asymmetric cryptography and digital signatures. Additionally, PKI proposes the use of public-key certificates. The public key within a certificate is used to authenticate the digital signature signed by the owner's private key. To ensure interoperability in relation to ***syntax*** and ***semantics*** for the use of public-key certificates in PKI in public communication systems, X.509 certificates designed in the ASN.1 format are proposed as a standard type of public-key certificates (Housley et al., 2008). The use of PKI requires the involvement of a Trusted Third Party (referred to as TTP hereafter), with the prerequisite that both entities must trust the same TTP. A TTP in this case is responsible for generating public-key certificates for both entities. It has its own private key and public key. Its public key is stored in a public-key certificate, which are distributed to both entities at the start. It then signs both entities' public-key certificates by using its private key, so both entities can use the obtained TTP's public key to verify the authenticity of the public-key certificates of each other (Adams and Lloyd, 1999; CGI, 2004).

The PKI authentication mechanism is much stronger than the password-based authentication, and it is also suitable for a system-system interaction. Since Web Services are mainly designed for automatic communication between systems, this mechanism is suitable, and it has already been used as well. Due to the existence of different PKI certificates (certificates are referred to as tokens in the field of Web Services) such as Kerberos tokens, SAML tokens, extensibility of including different tokens is needed. Thus, the WS-Security specification was proposed and published by OASIS, which specifies the combined use of XML-Encryption, XML-Signature along with the tokens (Lawrence and Kaler, 2004). Its compatibility to the SOAP protocol is also considered, so it can be used embedded in the header part of a SOAP message (see section 2.4.3). With the help of the WS-Security specification, both password-based authentication and PKI authentication methods can be used within Web Services.

As each kind of a token has its own syntax and semantics, to ensure that they can be embedded in the WS-Security specification, the interoperability issues in relation to ***syntax*** and ***semantics*** of a token are addressed.

## 2.4.8 WS-Trust

With the use of the WS-Security specification, Web Services can achieve authentication, when they can understand how to process common kinds of tokens. However, this is not always the case. An interoperability issue in relation to a Web Service's ***capability*** of processing different kinds of tokens may occur, if a Web Service Requester does not hold any kind of tokens that can be understood by a Web Service Provider. To address this issue, the WS-Trust specification was proposed and published by OASIS (Lawrence and Kaler, 2009a). WS-Trust allows a Web Service Requester to send a request to a TTP called a Security Token Service (referred to as STS hereafter) to obtain the kind of token that can be recognised by the Web Service Provider. This token will be sent to the Web Service Provider. The Web Service Provider will then send this token to the STS to confirm its authenticity. Once the authenticity of the token can be verified, successful authentication is achieved

(Nordbotten, 2009). As indicated by the WS-Trust specification, the concept of trust establishment is used as a synonym for successful authentication in Web Services. In other words, a unilateral trust relationship is established once authentication between two Web Services is successful, and the established trust relationship forms a foundation for using authorisation to help a Web Service Provider make access control decisions. In essence, the WS-Trust specification can provide interoperability in relation to ***syntax*** and ***semantics*** of messages along with the relevant aforementioned ***functionality***. In addition, interoperability in relation to ***capability*** of processing different tokens can also be provided by this specification.

### 2.4.9 WS-SecureConversation

Basically, WS-Security and WS-Trust are developed for authentication between two Web Services, when communication rounds between them only last for a short term (i.e. communication messages are transmitted for only one request and one response). When scenarios require more communication messages to be transmitted between two Web Services, the performance of the use of the two specifications may not be ideal, since the same public-key token (e.g. X.509 token) will occur in each message, thereby verifying the token in each message is a waste of time. WS-SecureConversation was thus published by OASIS (Lawrence and Kaler, 2009b) to address this issue.

The concept of a secure context was proposed in the WS-SecureConversation specification to provide a communication context that is secure enough for lengthy transactions. It is established before two Web Services start to exchange multiple messages. Thus, the public-key token is used just once to establish such a secure communication context. The efficiency of using WS-SecureConversation in comparison to WS-Security has been demonstrated in the experiments carried out by Liu, Pallickara and Fox (2005). In terms of the provision of interoperability, the WS-SecureConversation specification can provide interoperability in relation to ***syntax*** and ***semantics*** of its messages along with the ***functionality*** for processing the messages.

## 2.4.10 WS-Policy and WS-SecurityPolicy

Specifications such as WS-Security, WS-Trust and WS-SecureConversation etc. are useful and effective. For instance, WS-Trust can successfully provide ***capability*** interoperability overlooked by WS-Security, but it is assumed that the Web Service Requester can discover this issue before its communication with the Web Service Provider. Discovery of this issue cannot be achieved by using the WSDL file, due to the limited obligation of the WSDL, which only takes the responsibility for the specification of service descriptions (see section 2.4.4). To make a ***capability*** interoperability issue of a Web Service Provider known to other Web Service Requesters, WS-Policy was published by OASIS (Vedamuthu et al., 2007). WS-Policy provides a general framework providing interoperability in relation to ***syntax*** and ***semantics*** of messages to allow a Web Service Provider to describe its policies declaring supported information. It also specifies the rules about how Web Services should tackle different expressions of the same policies. These rules can be regarded as a provision of interoperability in relation to ***functionality***. Policies expressed in WS-Policy can be embedded in the WSDL file, so they can be publicly accessed by other Web Services to let them identify the ***capability*** interoperability issue before their communication.

However, the use of WS-Policy cannot completely provide declarations of security requirements, since its ***syntax*** and ***semantics*** do not suffice to specify security-related policies. This results in interoperability issues in relation to unknown ***syntax*** and ***semantics*** of security-related policies caused by the integration of security-related approaches. Thus, WS-SecurityPolicy was also published by OASIS (Lawrence and Kaler, 2009c), as a complement to WS-Policy. Interoperability in relation to ***syntax*** and ***semantics*** along with the relevant ***functionality*** can be provided by WS-SecurityPolicy to enable a Web Service to express policies declaring the supported information in relation to security.

In order to determine the tokens that can be supported by any Web Service Provider, information may be found in a human readable form within the WSDL description. This creates a problem for a system-to-system interaction, a fact that has been demonstrated by (Halevey, 2005)

*"Resolving schema heterogeneity is inherently a heuristic, human-assisted process. Unless there are very strong constraints on how the two schemes you are reconciling are different from each other, one should not hope for a completely automated solution. "*

This opinion is strongly supported by Nezhad et al. (2006), "It's unlikely that applications will be interested in reading and parsing the WSDL file at runtime to, for example, view which operations are supported and assess whether they're semantically and syntactically equivalent or similar enough to the desired operation". This human-involved limitation of WSDL matching largely restricts the ideal automated communication between Web Services.

In addition, confusion of whether to use the two specifications may arise due to the emergence of SAML and XACML (Nurse, 2010). The details are discussed in the next section.

### 2.4.11 SAML and XACML

Authentication is one of the most important security requirements, but it is still used as a basis for authorisation (Steiner, Neumant and Schiller, 1988). In other words, authorisation sometimes is more important than authentication, since authorisation directly controls whether a user or a service requester can access the resources of particular systems. This point of view is mentioned by Feigenbaum, (1998), and is strongly supported by Lopez, Oppliger and Pernul, (2004), "A merchant may be more interested in the authorization of his customers than in their authenticity". Traditionally, authorisation is a human-system interaction process that the system providing the resource needs to make decisions for granting the appropriate access control for the successful authenticated human user in relation to the resource (IBM and Microsoft, 2002; Boncella, 2004; Rowan, 2005; Mahmoud, 2005). However, in the field of Web Services, authorisation is normally a system-to-system interaction. This normally occurs when a Web Service Provider needs to make an access control decision for a specific service requested by a Web Service Requester. That is, the Web Service Provider will assign an appropriate access control level to the Web

Service Requester. The specific access control level restricts to what extent the Web Service Requester can access the service (e.g. read only, read and write etc.).

Currently, two access control approaches (i.e. RBAC and ABAC, discussed in the next section) are mainly used in Web Services, both of which share a common characteristic: the process of making decisions for access control will only be executed within the Web Service Provider, and will be finished with a decision. The decision as a result will then be sent back to the Web Service Requester. This characteristic restricts the trust relationship established by using existing access control methods to be unilateral, since the only action that the Web Service Requester performs is the submission of the related tokens or credentials for an access control decision to be made by the Web Service Provider (details of credentials are discussed in section 2.5.2). The unilateral characteristic of the existing access control methods can cause failures in authorisation, where potential successful authorisation is possible (the details of the issue are discussed in section 2.8).

Observing the process above, if the existing access control methods are applied in Web Services, potential interoperability issues in relation to ***syntax***, ***semantics*** and ***functionality*** can arise. Fortunately, messages defined within the Security Assertion Markup Language (SAML) specification published by OASIS (Mishra, Philpott and Maler, 2005) can be used to provide the ***syntax*** and ***semantics*** of messages for forming authentication/authorisation between two Web Services, although the SAML specification is initially designed for achieving Single-Sign-On (referred to as SSO hereafter) within Web Services.

In Web Services, access control policies can be expressed in either SAML or the eXtensible Access Control Markup Language (XACML) published by OASIS (Parducci and Lockhart, 2010). XACML is also designed based on XML, and it is well suited when cooperating together with SAML messages for achieving access control in Web Services (Nordbotten, 2009). XACML can provide interoperability in relation to ***syntax*** and ***semantics*** of messages along with the relevant ***functionality*** for processing the policy messages. However, complaints against XACML also exist. Developers may be confused with whether to use SAML, XACML or the combination of WS-Policy and WS-SecurityPolicy in Web Services, since many semantic similarities can be identified (Nurse, 2010). For example, Lee and Winslett (2008c) propose some modifications on the WS-SecurityPolicy to enable it to express access control policies along with the use of WS-Policy, but this functionality has been defined within XACML.

In addition, in accordance with the syntax of XACML, the value of each component of a rule (a policy is consisting of different rules) is atomic. This constraint makes it unable to contain the policy rules written in other policy languages as a value, since some policy rules are perhaps compound values similar to tokens. Therefore, existing systems using other policy languages for expressing authorisation rules cannot be directly used by XACML, if they are to be deployed as Web Services. This interoperability issue in relation to ***capability*** may result in the rewrite of these policies in XACML, since the existing algorithms for comparing the policies written in other policy languages against credentials not written in SAML cannot be directly used. In addition, it may also require the system to design new algorithms that can compare the policies written in XACML and the credentials not written in SAML.

Having discussed all the basic protocols and specifications in relation to authentication/authorisation within Web Services, the next section presents a review of access control methods (TN is discussed in Chapter 3) for achieving authorisation. The reason for reviewing them is due to the fact that the SAML messages do not provide interoperability in relation to ***functionality***. Fortunately, the ***functional*** interoperability can be provided by the internal structure stated in the existing access control methods and the authorisation systems.

## 2.5 Access Control Methods

Before the advent of Web Services, several access control methods were designed for traditional systems. The history of access control methods can be traced back to the 1960s. Originally, the description of access control is very abstract. The first occurrence of a clear description is possibly about system-level access control for programs as discussed in Lampson (1969). The main concepts proposed associated with access control were "subjects" (similar to the concept of "user" in current systems) and "objects" (similar to the concept of "resources" in current systems), which were linked to each other by using an "access matrix". This access control method was later referred to as Identity-Based Access Control (IBAC). IBAC actually integrates authorisation with authentication, since a specific access control level is immediately linked to a user's identity, after the occurrence of successful authentication of a user's identity. A typical use of IBAC is Access Control Lists (referred to as ACL hereafter), which list the users along with their relevant permission (e.g. write only, write and read, write only, read and modify) in relation to the specific resource.

The IBAC method is easy to implement, but there is a significant drawback. This drawback is stated by Yuan and Tong (2005) as "the number of identifiers in the ACL will increase and become difficult to maintain as more users request access, making this approach impossible to scale". Emig et al. (2007) point out another weakness of IBAC that it is inflexible to assign access control to identities. It can be understood why there are few attempts to apply IBAC within Web Services such as the work in Karp (2006), since the weaknesses within IBAC are addressed in RBAC (discussed in section 2.5.1) and ABAC (discussed in section 2.5.2), which have been more recently proposed. Therefore, more works can be found with respect to the suitability of the two access control methods (i.e. RBAC and ABAC, discussed in section 2.5.1 and 2.5.2 respectively) within Web Services.

### 2.5.1 RBAC

The Role-Based Access Control (RBAC) approach is initially proposed by Ferraiolo and Kuhn (1992). Unlike IBAC, it separates a user's access control from his/her identity, with access control levels linked directly to the assigned roles. One or more roles can be assigned to each user, and one or more control levels can be assigned to each role. Since roles are operating as intermediate players responsible for the connection of users' identities and their relevant access control levels, management of access control becomes more flexible in comparison with IBAC (Sandhu et al., 1996; Ferraiolo et al., 2001; Bhatti, Bertino and Ghafoor, 2004). In addition, the traditional application of RBAC in Web applications requires a Web server to predefine a relationship between users and roles and a relationship between roles and access control levels in its local databases. Due to the benefit of such a design, a Web server can have complete control over an access control decision by using RBAC (Barkley et al., 1997). For instance, whenever a new user intends to access a specific resource of a

Web server, and authentication of the user's identity has succeeded, the Web server will determine the user's access control according to its local RBAC databases, wherein the user's identity has been mapped onto a pre-defined role, which has been linked to a specific access control level. Due to its flexibility, RBAC is accepted as a ubiquitous access control approach in Web applications providing user-system interactions.

RBAC's reliance upon a user's authenticated identity requires the pre-storage of the user's identity, the pre-storage of the relationship between the user's identity and assigned roles, and the pre-storage of the relationship between the user's assigned roles and assigned access control permission within the system. In Web Services, a Web Service Provider can automatically create the storage of the identity of an unknown Web Service Requester, if PKI is used for achieving authentication. The storage of the link between the identity and the assigned role, and that of the link between the assigned role and the assigned access control permission can also be automatically created, if an implicit policy has been declared by the Web Service Provider. For instance, the identity of an unknown Web Service Requester will be linked to a specific role, if its public-key token is certified by a certain TTP. However, this automatic creation of the link cannot work, if the kind of a public-key token certified by a TTP is unknown to the Web Service Provider, as there exists a multitude of different TTPs such as Verisign, GlobalSign (Verisign, 2013; GlobalSign, 2013). In addition, an access control decision made depending on the restricted number of roles linked to the specific given identity may not be suitable in all the cases, whereby more information is required besides the identity (Hu et al., 2013). In addition, the coarsely grained nature of RBAC is another drawback restricting its usefulness within Web Services (Yuan and Tong, 2005).

The two drawbacks stated in the paragraph above do not imply the unsuitability of RBAC within Web Services. On the contrary, RBAC is still suitable to be used within a system, if it works across both Web Services and Web applications. As mentioned in section 2.2, the last case for the appropriate use of Web Services suggested by Booth et al. (2004) implicitly indicates that a system can still operate as a Web application, even though it has been deployed as a Web Service. The suitability of RBAC within traditional Web applications providing human-system interactions has been discussed earlier. From this perspective, it is not surprising that there are various research works investigating the application of RBAC within Web Services (Bhatti et al., 2003; Wonohoesodo and Tari, 2004; Bhatti, Bertino and Ghafoor, 2004; Liu and Chen, 2004; Xu et al., 2004; Mohammad et al., 2011).

### 2.5.2 ABAC

As RBAC is neither powerful nor flexible enough to be used for the Web Service Provider to make authorisation decisions for the Web Service Requester due to its intrinsic weaknesses, Yuan and Tong (2005) initially propose an Attribute-Based Access Control (ABAC) method for authorisation within Web Services. In ABAC, access control has no relation to the identity of a requester; it is instead decided upon whether the attribute information submitted by a requester can fulfil the policies declared by a service provider, as each specific access control level is tightly linked to the relevant policies. So, if all of the relevant policies can be met by the submitted attribute information, access control is successful; otherwise, it is failed. In addition to the independence of the identity of a requester, ABAC can further make up for the second weakness of RBAC: being coarsely grained. For instance, easy alteration or modification of the policies enables this approach to be more flexible and fine-grained (Yuan and Tong, 2005; Shen and Hong, 2006; Mewar, Aich and Sural, 2007; Emig et al., 2007; Sabbari and Alipour, 2011; Paci et al., 2011). With the use of ABAC, the authorisation issues occurring within Web Services mentioned above can be successfully addressed.

In fact, the notion of ABAC is not unfamiliar. Through observation, it can be identified that attribute information and policies are the fundamental components forming the basis of the feasibility of this approach. From the perspective of using policies to determine the relevant control level, ABAC actually is a kind of policy-driven access control method as proposed by Johnston et al. (1998), where access control permission can be granted, once the relevant policies have been fulfilled. From the perspective of the use of attribute information, they can be presented within digital credentials.

The concept of credentials (short for digital credentials) is mentioned by Winslett et al. (1997), who propose it for the use on the Web. A digital credential is a digital analogue of paper credentials in normal life such as passports, driving licences. It contains a combination of attribute names along with their values, which can represent the specific properties (e.g. membership, ownership etc.) belonging to the owner. More support of the use of credentials for showing the attribute information in ABAC can be identified in a number of other research papers (Winsoborough, Seamons and Jones, 1999; Winsborough, Seamons and Jones, 2000; Yu, Winslett and Seamons, 2001; Winsborough and Li, 2006; Lee, Winslett and Perano, 2009). A precise concept of a credential is stated in Winsborough, Seamons and Jones, (2000), "A credential is a digitally signed assertion by the credential issuer about the credential owner, …, [it] is signed using the issuer's private key and verified by using the issuer's public key". Authentication or verification of credentials by using digital signatures can prevent a fake user from stealing and using a user's own credential.

The motivation to use credentials on the Web is spurred by practical demands of authorisation. The focus of authorisation is on the properties owned by the users rather than their identities. For instance, if a university student intends to borrow some books from the university's library, a librarian is only interested in knowing whether s/he is a valid university student rather than his/her identity. So once the student can submit some attribute information within a valid credential (i.e. a student ID card) to prove his/her student membership, the access to these books can be granted (i.e. the student is allowed to borrow the books).

## 2.6 Existing RBAC/ABAC-based Authorisation Systems

As mentioned above, the typical characteristics of RBAC include the reliance on authentication of a user's identity, use of roles stored in a Web server's local databases, no use of policies and no use of credentials, whereas the characteristics of ABAC include no reliance on authentication of a user's identity, use of policies and use of credentials. There are several authorisation systems proposed for grid computing that can be used within Web Services. Interestingly, it is hard to judge whether the access control methods used within them belong to RBAC or ABAC, since the characteristics of both RBAC and ABAC can be identified. On the one hand, their authorisation relies on authentication of a user's identity and use of roles. On the other hand, credentials and policies are also used within these authorisation systems. Roles are not stored in the Web server's local databases, but are contained in credentials held by a user. From the perspective of the application of RBAC, the relationship between a user's identity and the assigned role are stored in the credentials rather than stored in the Web server's local databases. From the perspective of the application of ABAC, the roles contained in the credentials can be treated as one kind of attribute. These include PERMIS (Chadwick and Otenko, 2002), Akenti (Thompson, Esiari and Mudumbai, 2003), Shibboleth (Welch et al., 2005), VOMS (Alfieri, et al., 2004) and CAS (Pearlman and Welch, 2002).

## 2.7. Existing Solutions for Providing Interoperability between ABAC-based Authorisation Systems within Web Services

### 2.7.1 SAML Messages

Researchers such as Foster (2006) and Garzoglio et al. (2009) state that interoperability issues in relation to ***syntax*** and ***semantics*** between existing ABAC-based authorisation systems (in fact, these authorisation systems are still RBAC/ABAC-based, but researchers treat them as ABAC-based, as access control decision-making in these systems are mainly ABAC-based) within Web Services can be addressed by using SAML messages, which are supported in the fourth version of Globus, as a middleware toolkit (Foster, 2006).

This use of SAML messages exchanged between two ABAC-based authorisation systems is also leveraged within an attribute-based Authentication and Authorisation Infrastructure (AAI) for e-commerce by using Web Service technologies proposed by Schlager et al. (2006). Existing widespread used authentication services such as Microsoft .NET Passport (Oppliger, 2003), Liberty Alliance (Aarts et al., 2003) and authorisation systems (e.g. PERMIS, Akenti) can be integrated into this infrastructure shown in figure 2.1 below.

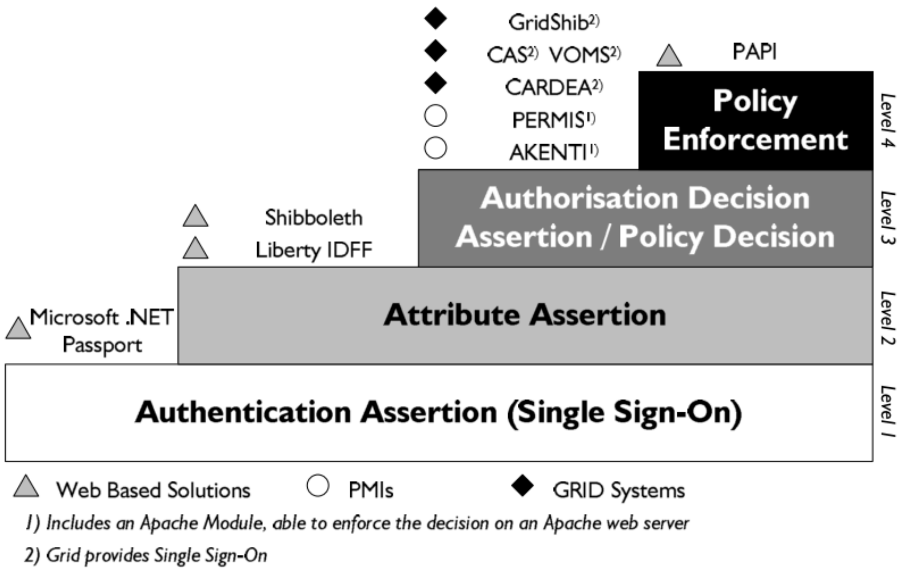
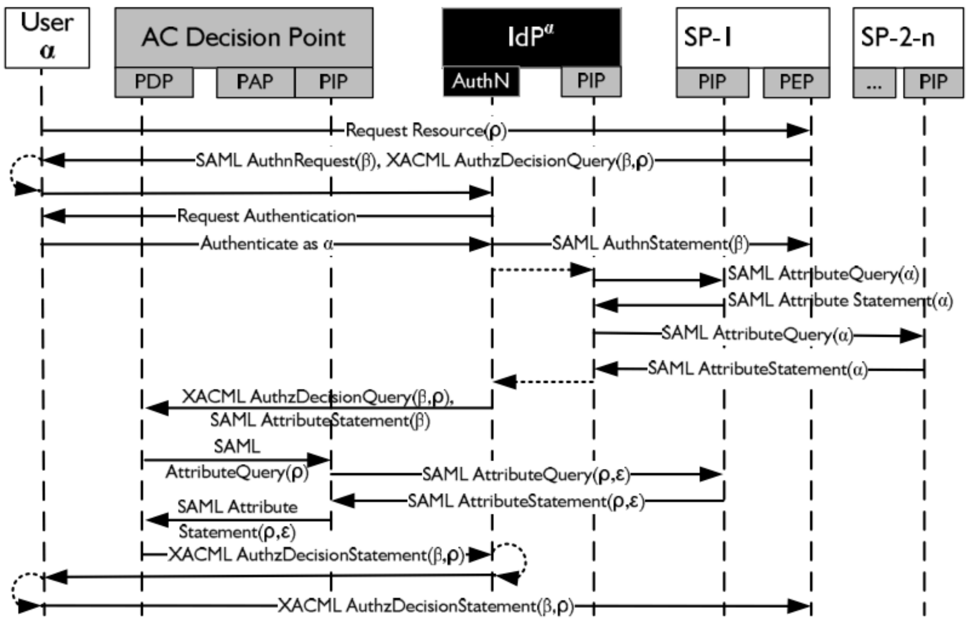

GridShib2)
CAS2) VOMS2)
CARDEA2)
PERMIS1)
AKENTI1)
PAPI
Policy Enforcement
Level 4
Authorisation Decision Assertion / Policy Decision
Level 3
Shibboleth
Liberty IDFF
Attribute Assertion
Level 2
Microsoft .NET Passport
Authentication Assertion (Single Sign-On)
Level 1
Web Based Solutions
PMIs
GRID Systems
1) Includes an Apache Module, able to enforce the decision on an Apache web server
2) Grid provides Single Sign-On

Figure 2.1. AAI solutions and functionalities (Schlager et al., 2006)

Within this AAI, level 1 is mainly used for providing authentication functionality such as SSO. Researchers propose a Point of Access to Providers of Information (PAPI) for achieving policy enforcement at level 4 of the AAI. At level 2 and level 3, XACML and SAML are mainly used for achieving authorisation functionality. More precisely, a data-flow model introduced in XACML enabling a Web Service provider to make an access control decision is widely leveraged within Web Services. Within this data-flow model, several points are defined including (1) Policy Administration Point (PAP) used for creating a policy or policy set, (2) Policy Decision Point (PDP) used to evaluate applicable policy for making an authorisation decision only based on the attribute information without knowing the identity of user or the actual requested resource, (3) Policy Enforcement Point (PEP) used for performing access control based on the decision made by the PDP and (4) Policy Information Point (PIP) used to act as a source of attribute values (Parducci and Lockhart, 2010).

Figure 2.2 below depicts a process of conducting authorisation within the AAI by combining the use of XACML and SAML for authentication and authorisation. This process occurs from level 1 to level 4 of the AAI. Therefore, it can be concluded that the use of XACML and SAML is an important feature of authorisation systems within the context of Web Services. It should be noted that although XACML messages are used in this process, (e.g. XACML AuthzDecisionQuery message and XACML

AuthzDecisionStatement message), these messages have been integrated into the latest version of the SAML specification (Mishra, Philpott and Maler, 2005). In other words, messages such as AuthzDecisionQuery and AuthzDecisionStatement are currently defined within the SAML specification rather than the XACML specification.



User α
AC Decision Point
PDP
PAP
PIP
IdP$^{\alpha}$
AuthN
PIP
SP-1
PIP
PEP
SP-2-n
...
PIP
Request Resource(ρ)
SAML AuthnRequest(β), XACML AuthzDecisionQuery(β,ρ)
Request Authentication
Authenticate as α
SAML AuthnStatement(β)
SAML AttributeQuery(α)
SAML Attribute Statement(α)
SAML AttributeQuery(α)
SAML AttributeStatement(α)
XACML AuthzDecisionQuery(β,ρ), SAML AttributeStatement(β)
SAML AttributeQuery(ρ)
SAML AttributeQuery(ρ,ε)
SAML AttributeStatement(ρ,ε)
SAML Attribute Statement(ρ,ε)
XACML AuthzDecisionStatement(β,ρ)
XACML AuthzDecisionStatement(β,ρ)

Figure 2.2. Attribute-based AAI reference model (Schlager et al., 2006)

Observing the use of SAML messages in AAI, it can be identified that the researchers (Schlager et al., 2006) actually have proposed new authorisation protocols to be added to the existing ABAC-based authorisation systems (e.g. PERMIS, Akenti). This finding is drawn on comparing the SAML messages-based protocols used within AAI (referred to as new authorisation protocols) against the original protocols used within the ABAC-based authorisation systems (referred to as old authorisation protocols). In particular, the protocol messages and internal structures designed within the new protocols and the old protocols are different. The internal structures defined within the protocols proposed in the AAI can provide interoperability in relation to ***functionality*** for processing the SAML messages for ABAC-based authorisation systems within Web Services.

AAI requires all of the existing ABAC-based authorisation systems to mandatorily use SAML messages for communication, including messages for expressing credentials and policies. In other words, the original languages for expressing credentials and policies used within each ABAC-based authorisation system cannot be used. Under this phenomenon, there is a need to convert credentials and policies used in the existing ABAC-based authorisation systems from their initial languages to SAML languages. This conversion process may be time-consuming and can be a waste of resource that the credentials and policies expressed within their initial form cannot be used.

Moreover, the syntax and semantics of SAML messages for expressing policies are not rich enough. For instance, the XACML policy language can provide a unique benefit that multiple policies need to be contained in a policy set or that combined algorithms (e.g. two rules are contradictory) can be used in policies. Languages designed in the WS-SecurityPolicy specification and the WS-Policy specification as suggested by Lawrence and Kaler (2009c) can supply their exclusive features such as policy assertion, intersection and association etc. for achieving simplicity of policy expression. SAML messages for expressing policies might be used in scenarios where simple attribute information needs to be submitted.

As each policy language owns its unique features and some policies may only be expressed in a specific policy language due to some unique requirements, it is possible that developers try to enable their Web Services to support different policy languages in order to provide policy expression flexibility for different circumstances (Lang et al., 2006). This benefit should also be applied to credential languages in Web Services.

However, if multiple policy and credential languages are used in the ABAC-based authorisation systems within Web Services, an interoperability issue in relation to ***capability*** may arise, as some of the languages for expressing credentials and policies recognised in a Web Service Requester may not be understood by a Web Service Provider, and vice versa.

### 2.7.2 An object-oriented framework for adopting different policy languages

In contrast to AAI, Globus also supports the mandatory use of SAML messages for expressing credentials, but it allows the use of different policy languages for expressing policies. To enable an ABAC-based authorisation system to support multiple policy languages, an object-oriented framework is designed within Globus (Lang et al., 2006). Unfortunately, this framework can only assure that two authorisation systems can process policies expressed in different languages, but without providing a guarantee that two authorisation systems can know whether the policy languages used for policies submitted by an entity are supported by another entity. Therefore, using this object-oriented framework cannot provide enough interoperability in relation to ***capability***.

## 2.8 Limitation of the Application of ABAC within Web Services

A conclusion can be drawn from the review of the application of current access control methods within Web Services that ABAC is well suited for a Web Service Provider to perform authorisation decisions. With respect to RBAC, it is most appropriately used to support scenarios where the Web Service still operates as a Web application providing a human-system interaction for authentication purposes, or in scenarios where the identity of a Web Service Requester is already stored within the system of the Web Service Provider. In the particular case that the Web Service Provider cannot gain the identity of the Web Service Requester for their first-time communication (see section 2.5.1), ABAC is the only available option to enable the authorisation. In addition, ABAC can still be available for a Web Service Provider to make access control decisions for a Web Service Requester, even though they have known each other. The fine-grained characteristic even boosts the use of ABAC within Web Services. Unfortunately, limitations still exist within ABAC.

Normally, successful authorisation can be achieved by using ABAC, when a Web Service Requester can submit all of the credentials required by the policies published by the Web Service Provider. However, when the credentials possessed by a Web Service Requester are treated as sensitive (i.e. they cannot be disclosed to an unknown Web Service Provider), potential successful authorisation may fail (Yu, Wislett and Seamons, 2001; Winslett et al., 2002; Hess et al., 2004; Frikken et al., 2006; Mbanoso et al., 2006; Winsborough and Li, 2006). This issue arises due to the fact the trust establishment by using ABAC is still unilateral as discussed in section 2.4.11. A case scenario is presented below to point out this limitation:

*In the UK, each university has its own local Web application as a platform allowing students to share resources in an E-Learning environment. Assume that, they intend to expose their Web applications as Web Services to the Internet so that students of one university can access resources from students in other universities. Suppose that Alice is a student of University A, and Bob is a student of University B. Through a collaboration between the Web Service A (referred to as WSA hereafter) of University A, and Web Service B (referred to as WSB hereafter) of University B, it allows Alice to request access to the resource provided by Bob. In this scenario, Bob can declare his own policies. Alice does hold the credentials that can fulfil Bob's policies, but Alice treats some of them as sensitive. So Alice also declares policies for protecting the disclosure of these sensitive credentials.*

By using an existing ABAC solution, potential successful authorisation will fail, since Alice is reluctant to disclose her credentials, unless all the pertinent policies can be met. A further issue causing the successful authorisation to fail in this scenario is due to the existence of sensitive policies (Seamons, Winslett and Yu, 2001; Seamons et al., 2002a; Holt et al., 2003; Yu and Winslett, 2003a; Bertino, Ferrari and Squicciarini, 2004b; Li, Li and Winsborough, 2005; Paci et al., 2011).

Fortunately, an access control approach called TN can be borrowed to address this issue, as it can help two unknown entities to establish a bilateral trust relationship for achieving access control (Winsborough and Li, 2002a; Winsborough and Li, 2002b; Shen and Hong, 2006; Skogsrud et al., 2009). However, utilisation of TN within Web Services will also cause multiple interoperability issues between authorisation systems within Web Services. Due to the huge amount of literature in the field of TN, and the demand of a clear discussion in this Thesis, presenting its state of the art in a new chapter is considered as the most appropriate way. Therefore, the next chapter introduces a thorough review of TN and the specific issues of its application within Web Services.

# Chapter 3. Trust Negotiation and Interoperability: State of the Art

## 3.1 Introduction

Chapter 2 has reviewed the literature in the fields of authorisation protocols within Web Services and access control methods along with the relevant authorisation systems. ABAC has been treated as the most appropriate access control method within Web Services. However, due to the limitation of the unilateral characteristic of ABAC, potential successful authorisation may fail, when protection of the disclosure of sensitive credentials or policies is required. The result caused by this limitation is demonstrated in the case scenario presented in section 2.8. TN is also an access control approach, which can enable two unknown entities to establish a bilateral trust relationship (Winsborough and Li, 2002a; Winsborough and Li, 2002b; Shen and Hong, 2006; Skogsrud et al., 2009). With the application of TN, authorisation failure can be transformed into authorisation success in some circumstances.

To adopt the state-of-the-art TN approaches within Web Services also presents a multitude of interoperability issues. To clarify the issues (highlighted in bold and italic) existing within the state-of-the-art TN, this chapter begins by introducing the concept of TN (see section 3.2). Following this concept, a detailed review of TN is then presented (see section 3.3). TN involves many different components, and each of these components is critically reviewed in order to gain a thorough understanding of the state of the art.

In addition, several TN-based authorisation systems have been implemented to support the use of TN. To enable the review of TN to be complete, a review of these TN-based authorisation systems is also presented. Following this review, critical analysis is provided to point out the outstanding interoperability issues (highlighted in bold and italic) between an ABAC-based authorisation system and a TN-based authorisation system and those between two TN-based authorisation systems (discussed in section 3.4). To explore the possible factors along with their characteristics causing the relevant interoperability issues becomes the first goal of this research.

Based on the critical analysis of the interoperability issues through the review of ABAC-based authorisation systems within Web Services discussed in Chapter 2, and the review of the TN-based authorisation systems and the existing interoperability models of the most relevance in this Chapter, an improved multi-layered interoperability model is constructed (see section 3.5). This improved interoperability model concludes the identified interoperability issues existing between the authorisation systems within Web Services. In addition, it may also be applied in other distributed systems environments to aid researchers and practitioners in identifying hidden interoperability issues between systems. Following the clarification of the improved interoperability model, review of the work in relation to other interoperability models of less relevance is presented to point out the difference between the improved interoperability model and other interoperability models (see section 3.6). Lastly, the improved interoperability model serves as a foundation for guiding the exploration of possible solutions for addressing the second research problem (see section 3.7). To explore a potential solution, a discussion of possible solutions for addressing the issues is provided (see section 3.8).

## 3.2 Concept of TN

TN is initially proposed by Winsborough, Seamons and Jones (1999). Over the past decade, it has been acknowledged as another access approach to allow two unknown entities to establish a bilateral trust relationship by exchanging digital credentials containing attribute information to help two entities make access control decisions. While credentials have been the dominant means for transmitting attribute information between negotiating entities, further information such as policies may also be exchanged as other requirements for TN. Currently, the concept of TN can be described as: a selected set of information including credentials, declarations, policies or other required information exchanged between two unknown entities to establish a trust relationship in a *bilateral*, *iterative* and *cumulative* process (Winsborough, Seamons and Jones, 2000; Yu, Ma and Winslett, 2000; Yu, Winslett and Seamons, 2001; Winsborough and Li, 2002b).

The *bilateral* characteristic indicates that two entities are equal in status, regardless of which is the service requester or the service provider. In other words, both of them can supply their own policies to require the counterpart to submit necessary files (e.g. credentials, declarations) containing the essential attribute information. The *iterative* characteristic expresses that the process of the exchange of the information between two entities can be repeated in multiple steps, until they disclose all of the associated files. The *cumulative* characteristic conveys that the trust level can only be escalated during the process of negotiation. That is, sensitive resources (i.e. sensitive credentials or policies) that can be divulged at a low level must be revealed at a high level as well.

Furthermore, a unique characteristic of TN is that each entity involved in TN can choose to use its own particular strategy to determine how the resources should be disclosed (i.e. policies, credentials with some of these may contain sensitive information) to the counterpart. Through the observation of the nature of TN, it is actually designed as an extension of ABAC.

Additionally, in comparison with ABAC, the TN approach provides several benefits including the following.

- A service requester owns the ability to question the trustworthiness of a service provider. This rule not only allows the service requester to submit credentials to fulfil the policies declared by the service provider, but also requires the service provider to submit credentials to the service requester in accordance with the policies disclosed by the service requester. Following this rule, the trust relationship established by ABAC becomes bilateral rather than unilateral, and both entities are on equal terms with each other in TN.
- Trust is gradually raised, when each entity's policies can be fulfilled by each other's detailed attribute information within the credentials. Complete trust is only established when all of both entities' relevant policies have been fulfilled by each other's credentials, and access control is only granted if complete trust has been established.
- Each entity can choose their own strategy to determine how to disclose their resources, if some of them are treated as sensitive.

The use of TN can address the problem discussed in the case scenario presented in section 2.8. For instance, if TN could be adopted within Web Services, WSA would disclose Alice's policies to WSB to see whether Bob had the credentials that could fulfil Alice's policies. If Bob was in possession of the insensitive credentials containing the required attribute information, WSB would disclose them to WSA. As a result, WSA would submit Alice's sensitive credentials to WSB to reach successful authorisation, which is failed by using the existing ABAC approach.

However, interoperability issues do exist when adopting state-of-the-art TN within Web Services. To understand where these issues come from, a detailed review of TN from different aspects (as necessary components of TN) is presented in the following sections.

## 3.3 Review of TN

In accordance with the concept of TN, the main components of TN as covered in a variety of literature have included strategy (Winsborough, Seamons and Jones, 1999; 2000), digital credential (Seamons, Winslett and Yu, 2001), declaration (Bertino, Ferrari and Squicciarini, 2003b; 2005), and access control, disclosure policy (Bertino, Ferrari and Squicciarini, 2003a; Smith, Seamons and Jones, 2004; Koshutanski and Massacci, 2005). Before presenting the review of each component, their concepts are described first shown as follows.

- **Strategy**: is to help two entities determine when and what necessary information should be requested and disclosed in TN (Winsborough, Seamons and Jones, 1999; 2000).
- **Credential** (short for digital credential): is a representation of the combination of attributes of the owner, which can be signed using the issuer's private key and be verified by the issuer's public key (Seamons, Winslett and Yu, 2001).
- **Declaration**: contains extra information not included in the credentials to aid the process of TN. Unlike a credential, a declaration is not certified, but stated by the owner per se (Bertino Ferrari and Squicciarini, 2003b; 2005).
- **Policy**: there are two notions of policy: access control policy and disclosure policy (Bertino, Ferrari and Squicciarini, 2003a; Smith, Seamons and Jones, 2004). An access control policy claims the rules to grant the permission to access the requested

resource. A disclosure policy states the rules for disclosing sensitive credential(s) to the counterpart (Koshutanski and Massacci, 2005).

As TN has been developing for more than a decade, there is no surprise that a variety of frameworks and implementations of TN have existed. To enable the review of TN to be complete, a discussion of the existing TN-based authorisation systems is also presented.

### 3.3.1 Strategy

Strategies designed for TN can be classified into two categories: non-policy-exchanged strategies and policy-exchanged strategies. Strategies within the non-policy-exchanged strategies category only allow two entities to exchange authorisation credentials, whereas strategies within the policy-exchanged strategies category enable two entities to exchange authorisation policies and credentials. The non-policy-exchanged strategies only include the eager strategy (Winsborough, Seamons and Jones, 1999; 2000). The policy-exchanged strategies include: parsimonious strategy (Winsborough, Seamons and Jones, 1999; 2000), Prudent Negotiation Strategy (Yu, Ma and Winslett, 2000), Disclosure Tree Strategy; Binding Tree Strategy (Yu, Winslett and Seamons, 2003), Deterministic Finite Automaton Negotiation Strategy (Lu and Liu, 2009), adaptive strategy (Guo and Jiang, 2010) and Semantically Relevant Negotiation Strategy (Liu et al., 2013).

#### *3.3.1.1 Non-Policy-Exchanged Strategies*

The eager strategy (Winsborough, Seamons and Jones, 1999; 2000) is currently the only strategy within the non-policy-exchanged strategies category. It should be noted that it is possible that any new potential strategies that also own the characteristics of the non-policy-exchanged strategies may come up in the future. Thus, a plural form for the non-policy-exchanged strategies is used throughout this Thesis. The eager strategy allows two participating entities to exchange as many credentials as possible with each other. Each of the two participating entities' sensitive credentials unlocked by credentials sent from the counterpart will be treated as those that can be disclosed. The aim of this strategy is to exchange credentials that can be disclosed to unlock more sensitive credentials protected by the counterparts' security policies (unlock of sensitive credentials is achieved, if the received credentials can fulfil the local policies of an entity). This aim applies to the two entities, when both of them are using this strategy in TN.

One unique characteristic of using the eager strategy is that credentials sent in a previous step from an entity will also be sent in the next step with new unlocked sensitive credentials. The conditions for making a decision to terminate negotiation are different in the two participating entities. From the perspective of the service provider, at each round, it will check whether or not the received credentials from the service requester have fulfilled the policies protecting the requested resource. If the policies can be met, it will terminate negotiation with the service requester with a successful ending. If the policies cannot be met, and it has discovered that the received set of credentials is exactly the same as the set received in the prior step, or that they cannot unlock more local sensitive credentials, it will terminate negotiation with the service requester with a failed ending. From the perspective of the service requester, it will terminate negotiation in success, when it receives a message informing successful negotiation from the service provider. Alternatively, it will terminate negotiation in failure, if it has discovered that the received set of credentials is exactly the same as the set received in the prior stage, or that they cannot unlock more local sensitive credentials. Another unique characteristic of using the eager strategy is that there is no exchange of any policies, when both entities are using the eager strategy to run TN, since the core rule of this strategy is that the received credentials are used to unlock more local sensitive credentials. The third characteristic is that the service provider will disclose its insensitive credentials, after receiving the initial request from the service requester.

#### *3.3.1.2 Policy-Exchanged Strategies*

The parsimonious strategy (Winsborough, Seamons and Jones, 1999; 2000) allows entities to explicitly disclose the policies protecting the relevant sensitive credentials. Disclosure of local sensitive credentials is only available when the credentials sent from the counterpart have fulfilled the relevant local policies. In other words, when both entities are using the parsimonious strategy, the received credentials should aim to fulfil the specific policies disclosed in the previous steps. Therefore, the behaviour to process the received credentials with the use of the parsimonious strategy is different from that used in the eager strategy.

Yu, Ma and Winslett (2000) argue that TN may fail by using the two strategies, when potential successful TN is possible, but no detailed proof is provided. To improve this stated issue, they develop a new strategy named Prudent Negotiation Strategy (PRUNES) to guarantee that all potential successful TN can succeed. When two entities use this strategy, they initially need to exchange their policies (referred to as the policy-exchange phase) protecting sensitive resources (sensitive credentials are also treated as sensitive resources), to produce a negotiation search tree before exchanging their credentials (referred to as the credential-exchange phase). During the time of the policy-exchange phase, in each round, only one part of a rule requesting one required credential is sent out in a policy message. According to the XACML policy language designed by Parducci and Lockhart (2010), a policy message has a policy set, which can contain multiple policies. Each policy can have multiple rules, and each rule can require different combinations of multiple credentials by using logic symbols such as AND/OR. In addition, within the policy-exchange phase, the entities will only convey information that there are local credentials that can fulfil the remote policies, but without providing any real credentials.

The use of PRUNES is based on an assumption that the strategy implemented by both entities must completely conform to the process discussed above. This is also based on the assumption that both systems are not maliciously designed. However, this assumption is not appropriate in terms of the TN context. TN is used in the context that two entities are unknown to each other. In other words, before their communication, there is no trust relationship between them. Without trust, there is no point that they are willing to believe the unauthenticated information conveyed by each other in relation to what local credentials can fulfil the remote policies. Therefore, it is possible that the sensitive policies of an honest Service Requester/Service Provider would be disclosed to a malicious Service Provider/Service Requester, even though the malicious Service Provider/Service Requester did not possess the credentials that could fulfil these policies. In addition, the algorithm for implementing this strategy is more complex in comparison with the eager and parsimonious strategies.

As different strategies are created, ***strategic*** interoperability becomes an issue. Yu, Winslett and Seamons (2001) state, “no two of the strategies proposed so far [including the eager strategy, parsimonious strategy and PRUNES strategy] will interoperate”, and they suggest that interoperable strategies can provide for a higher success rate of TN. Unfortunately, in their paper, they do not give any explanation of why the three strategies cannot be interoperable with one another. Therefore, the researcher of this Thesis had to analyse the three strategies to identify the reason himself. After a critical analysis of the three strategies, the researcher eventually found out the implicit reason explained as follows.

With the use of the eager strategy, an entity will not expect to receive any other files (e.g. policies used in the parsimonious strategy and PRUNES strategy) other than credentials. No idea of how to process the incoming policies enables the eager strategy to not be interoperable with the other two strategies. With the use of the parsimonious strategy, an entity will not expect to receive any credentials without knowing what local policies they aim to fulfil (this phenomenon will occur when the counterpart uses the eager strategy), since it does not know how to process this kind of credentials. It also will not expect to receive messages informing what remote credentials can fulfil the specific local policies (this phenomenon will occur when the counterpart uses the PRUNES strategy), since it does not know how to process this kind of message.

To address this issue, the notion of a Disclosure Tree Strategy (DTS) family is introduced. The operation of a DTS assumes that all of the policies can be exchanged by two entities regardless of their sensitivity. During TN, strategies can be transferred to one another, if they come from one DTS family. The Binding Tree Strategy (BTS) family is proposed by extending the operability of DTS to convey more information (Yu, Winslett and Seamons, 2003). Although DTS and BTS have been proposed, and can address the strategic interoperability issue in theory, it is difficult to categorise any new designed strategy to the existing DTS or BTS in practice. In addition, Baselice, Bonatti and Faella (2007) point out another weakness, “These works [DTS and BTS families] are tailored to specific frameworks - so their results cannot be extended to competing approaches - and introduce assumptions that cannot be always guaranteed”. Thus, the families are not suitable to be adopted in practice.

So far the most appropriate approach ensuring interoperability between strategies is the one stated by Yu, Winslett and Seamons (2001), that strategy interoperability should be addressed at least by using self-compatibility. That is, a strategy can at least be interoperable with itself, when both entities are using the same strategy, and proof of strategy interoperability is that TN can be successful when two entities are using their own strategies. Furthermore, they suggest that a TN protocol be designed to support simple strategies at a higher priority compared to more intelligent strategies, which are hard to be implemented in real systems.

Lu and Liu (2009) suggest a strategy named Deterministic Finite Automaton Negotiation Strategy (DFANS) that can improve the computational efficiency in comparison with PRUNES, avoid Policy Cyclic Dependencies (referred to PCD hereafter) to some extent by applying the Oblivious Signature Based Envelope (OSBE) protocol proposed by Li, Du and Boneh (2003) and protect the disclosure of sensitive policies (neglected in PRUNES). However, DFANS only allows an entity to submit or process one received credential or a received request for one credential (as an atomic part of a rule of a policy) at a time. OSBE is an approach which enables the protection of the disclosure of the sensitive fact that a trusted authority has signed a credential, but does not enable protection of the disclosure of the sensitive attribute information within credentials. PCD happen when entity A declares a policy p1 protecting a credential c1, which will only be disclosed when entity B submits a credential c2, whilst entity B defines a policy p2 protecting a credential c2, which will only be disclosed when entity A discloses a credential c1. The result of this problem can be disastrous, since two entities will keep sending the same policies resulting in an undesirable infinite communication loop. Therefore, PCD must be addressed by possible approaches, but OSBE is not suitable for PCD cases, where the protection of the sensitive attribute information is needed.

Guo and Jiang (2010) propose an adaptive strategy by integrating the notion of reputation systems. Once an entity can make sure that the counterpart has a high reputation of trust, sensitive credentials protected by its policies can be disclosed, even though their relevant policies that request necessary credentials have not yet been fulfilled. The use of the adaptive strategy is based on the assumption that both entities have supported the use of reputation systems and that both entities are willing to believe the counterpart's reputation. However, limitations can still be discovered within this method when it is adopted in practice. Firstly, specific approaches for an entity to retrieve the reputational level of the counterpart are not clarified. In other words, there is no clear answer for questions such as whether an entity needs to communicate with the counterpart for obtaining its reputation level or whether the entity has already obtained the reputation level in advance. Secondly, whenever an entity receives policies from the counterpart, it not only needs to analyse the received policies, but also needs to analyse the local policies declaring what credentials can be disclosed in relation to a certain reputation level of a counterpart. This complex process eventually increases the computational cost of a system.

Liu et al. (2013) propose a Semantically Relevant Negotiation Strategy (SRNS). Extra ontology-based information exchanged with policies is used to perform semantics-based matching, when comparing local credentials against remote policies.

Taking into consideration the eager strategy, parsimonious strategy, PRUNES, DFANS and SRNS, an interoperability issue in relation to ***strategy*** can still occur between any two of them. Apart from the explanation given above, an additional explanation of their interoperability issues is given below: (1) when an entity uses the DFANS, the counterpart using the eager strategy cannot process policies provided by the entity. If the counterpart uses the parsimonious strategy, this entity cannot process multiple policies and credentials, as the DFANS only allows an entity to process one policy and one credential in a time. If the counterpart uses the PRUNES, the entity cannot process the information used in the PRUNES declaring that the counterpart does have a credential that can fulfil a rule declared by the entity. (2) When an entity uses the SRNS, the counterpart using any one of the eager, parsimonious, PRUNES or DFANS strategies cannot process ontology-based information provided by the entity.

To determine the suitability of the adoption of the strategies mentioned above along with TN in Web Services, the assessment of their feasibility is needed, except the DTS and BTS families, as the inappropriateness of their adoption have been discussed earlier.

Observing the policy-exchanged strategies, there are similarities and differences amongst them. In terms of the similarity between the parsimonious strategy and the adaptive strategy, an entity can disclose more than one policy to the counterpart in an outgoing message. The similarity held by the PRUNES and DFANS is that an entity will only submit an atomic element of a policy for requesting one credential held by the counterpart.

The action of submitting credentials by using the DFANS is the same as that of using the parsimonious strategy, adaptive strategy or SRNS. More precisely, with the use of any one of the four strategies, an entity submits local non-sensitive credentials to the counterpart immediately in the next round, after receiving policies from the counterpart.

In terms of differences, with the use of the DFANS, an entity will disclose a relevant credential fulfilling the atomic element of a policy disclosed by the counterpart immediately. By contrast, with the use of the PRUNES, an entity will not immediately disclose a relevant credential, even if it can fulfil the atomic element of a policy disclosed by the counterpart. In terms of the SRNS, Liu et al. (2013) do not explicitly express whether an entity discloses its policies like the way of the parsimonious strategy or that of the PRUNES.

When using the adaptive strategy, an entity will also only disclose one credential to the counterpart, but can in addition disclose a policy message containing a request for multiple credentials.

In conclusion, it can be identified that the use of the PRUNES, DFANS or adaptive strategy can result in more rounds of negotiation in comparison with the use of the parsimonious strategy. Without the use of the OSBE protocol, DFANS does not show any advantage in comparison with the parsimonious strategy except in the protection of sensitive policies, since it depends on complex algorithms and requires a high communication cost. Researchers proposing strategies including PRUNES (Yu, Winslett and Seamons, 2003), DFANS (Lu and Liu, 2009) and SRNS (Liu et al., 2013) state that the communication cost of their strategies only subsumes the number of credentials. However, as they also admit that the transfer of policies does exist within their strategies, then the exchange of policies should also be calculated in their communication costs. Thus, it can be identified that there is an additional impact on communication costs.

By contrast, the ease of implementation and need for fewer communications enable the parsimonious strategy to be superior. As the parsimonious strategy can protect the disclosure of sensitive credentials, it is not difficult to add the functionality for the protection of sensitive policies to it. Furthermore, since the OSBE protocol is independent of any strategies, the parsimonious strategy can also leverage it to address PCD to some extent. Therefore, concluded from the discussion of the above benefits, it would be more appropriate to use the parsimonious strategy in practice.

#### *3.3.1.3 Comparison between Non-Policy-Exchanged Strategies and Policy-Exchanged Strategies*

In terms of the non-policy-exchanged strategies, they can protect the disclosure of sensitive credentials owned by a Service Requester/Service Provider. However, the third characteristic of the eager strategy (as a typical non-policy-exchanged strategy) that the Service Provider always discloses its insensitive credentials first implies that it is not suitable to be used in scenarios where all of the credentials possessed by the Service Provider are sensitive. Furthermore, this characteristic may enable it not to be widely accepted, since it may not be reasonable that a Service Provider has to disclose credentials to an unknown Service Requester first during the process of authorisation, even though the credentials are insensitive. This unreasonableness can be inferred from the features of the existing access control approaches such as RBAC or ABAC (Ferraiolo and Kuhn, 1992; Yuan and Tong, 2005), which always aim to protect the benefit of the Service Provider at a higher priority. This requirement means that the use of the non-policy-exchanged strategies can be disadvantageous to a Service Provider.

In comparison with the non-policy-exchanged strategies, the use of policy-exchanged strategies may be more acceptable, since they always require the Service Requester to submit credentials first to unlock the sensitive credentials owned by the Service Provider, if all of them are treated as sensitive. These credentials are provided as indicators of trust to encourage the Service Provider to trust the Service Requester at a basic level, so that the Service Provider is willing to disclose its credentials based on a basic level of trust. The characteristic allowing the initial disclosure of credentials at the beginning of the process from a Service Requester means that the use of the parsimonious strategy can be more advantageous to a Service Provider. Therefore, from the perspective of providing this benefit, the use of the policy-exchanged strategies may be more easily accepted than that of the non-policy-exchanged strategies in terms of their application in practice.

### 3.3.2 Credential and policy

In the field of TN, there is a variety of credential languages. The first informal language for expressing credentials used in TN was developed by Winsborough, Seamons and Jones (2000). It consists of two parts named the Property-based Authentication Language (PAL) and the Role-based Authorisation Language (RAL). Winsborough and Li (2002a) state that assumptions about credential languages made by Winsborough, Seamons and Jones (2000) are so simple that they cannot fulfil practical demand. Therefore, they introduce $RT_0$, a member of a family of role-based trust management languages.

An XML-based TN Language (X-TNL) for Web communication was developed and presented by Bertino, Ferrari and Squicciarini (2003b). In addition, an Attribute-based TN Language (ATNL), a family of Role-based Trust management languages, which can specify credentials based on Role-based Trust-management framework (RT) has been developed by Li, Li and Winsborough (2005). This credential language can present membership and delegation as well as provide the capability of presenting values of attributes. One benefit of ATNL is that each attribute can be linked to different credentials. Attributes can even be assigned a sensitive or non-sensitive value depending on the sensitivity degree as determined by the entity.

As there are numerous credential languages in the field of TN, there is no surprise that various policy languages have also been designed relevant to these credential languages. To prevent (non) possession-sensitive credentials from being inferred by the counterpart mentioned by Seamons et al. (2002b), a notion of Attribute acknowledgement policies (Ack policies) is recommended by Winsborough and Li (2002a). X-TNL can represent a policy language cooperating with X-TNL certificates for expressing credentials and declarations (Bertino, Ferrari and Squicciarini, 2003b). Olmedilla et al. (2004) present the use of a PeerTrust language for TN. The use of ATNL mentioned above to specify policies is described by Li, Li and Winsborough (2005). An extension of WS-SecurityPolicy has been suggested to enable their specification of TN policies in Web Services (Lee and Winslett, 2008c). By extending WS-SecurityPolicy with the use of WS-Policy, the syntax and semantics allows entities to state specific constraints on the values of attributes in a credential for TN between Web Services.

Once decisions over the expression of credentials and policies have been solved, the next task is to equip an entity with an ability to compare them to reach a decision. This ability is required in scenarios where an entity receives credentials submitted by the counterpart. The entity needs to compare them against the local policies to make a decision on whether policies can be fulfilled by credentials. This ability is also required in scenarios where policies sent from the counterpart have been received by the entity. The combination of the two functionalities demands more abilities of a policy compliance checker (Seamons et al., 2002a; Smith, Seamons and Jones, 2004). The policy compliance checker is able to match remote credentials with local policies to determine whether or not certain credentials can fulfil policies, or vice versa. In addition, whenever an entity discloses its credentials, the credentials need to contain relevant information (name or type of credential, which policies the credentials can fulfil etc.) for fulfilling the disclosure requirement with the communicating counterpart (Yu, Winslett and Seamons, 2003). This can aid the policy compliance checker of the counterpart in processing these credentials more efficiently. This requirement is particularly necessary when both entities are using the parsimonious strategies, since they may have disclosed several messages containing policies before receiving any credentials from the counterpart. Likewise, when an entity discloses policies, relevant information (sample type of credentials, owner of policies etc.) should also be contained within them.

Through the observation of the credential and policy languages designed for TN, each of them has its own syntax and semantics. It is feasible that they can be used directly in Web Services, as multiple token languages have been used in Web Services without any problems. However, a ***capability*** interoperability issue may occur, as some credential and policy languages recognised by the policy compliance checker of one communicating system may not be understood by the policy compliance checker designed within another communicating system.

### 3.3.3 Declaration

A notion of declaration is presented in the XML-based X-TNL declaration (Bertino, Ferrari and Squicciarini, 2003b). Similar to credentials, it can store personal attribute information to help establish a trust relationship for entities more effectively (Bertino, Ferrari and Squicciarini, 2005). However, its outstanding drawback of not being certified by a TTP will bring potential danger to Web Services, due to the existence of malicious Web Services as stated in other works (Geer, 2003; Curphey, 2005). The intrinsic nature of the use of declarations allows an entity to create arbitrary declarations containing attributes, and to use its own private key and public key for issuing and verifying declarations without any involvement of a TTP, so the authenticity of declarations cannot be guaranteed. This weakness could be utilised by a malicious Web Service Requester to issue its own fake declarations to enable itself to access sensitive resources, if a Web Service Provider allowed the use of declarations, and the fake declarations could fulfil the relevant access control policies. This result should have been avoided, as the Web Service Provider would not have granted access to the malicious Web Service Requester, if it had discovered that the submitted declarations were not trustworthy enough. However, the Web Service Provider has no capability of judging the trustworthiness of declarations due to the nature of declarations. Therefore, the researcher of this Thesis does not support the use of declarations in the context of using TN.

### 3.3.4 Existing TN-based authorisation systems

Existing RBAC/ABAC-based authorisation systems are discussed in section 2.6. This current section discusses the existing TN-based authorisation systems. There are similarities that can be found between the RBAC/ABAC-based authorisation systems discussed in section 2.6 and the systems discussed in this section, but there are also outstanding differences. Similarities such as the use of an object-oriented design for system architectures can be found in Globus (a framework containing the existing authorisation systems) and TrustBuilder2 (one representative of the typical TN-based authorisation systems, discussed later). In addition, systems in this section have a similar ultimate purpose of use in making access control decisions for a service or resource.

In terms of differences, the communication messages between the existing RBAC/ABAC-based authorisation systems may be restricted to one request message containing a token (e.g. PERMIS), or restricted to one request and one response message, where a token is contained in the request message and an access control decision can be found in the response message (e.g. Akenti). Although systems such as Shibboleth or CAS support more than a single round communication process, the transfer of credentials is still only sent from a service requester to a service provider. The one-way direction of credential transfer causes the trust relationship established by using the RBAC/ABAC-based authorisation systems to be unilateral.

Unlike these RBAC/ABAC-based authorisation systems, the TN-based authorisation systems discussed in this section are all designed and developed to support the use of TN. That is, typical components required by TN mentioned in the above sections are designed and developed in these systems (in particular, the design of strategies is required), whereas these components do not exist in the existing RBAC/ABAC-based authorisation systems. In addition, multiple rounds of communication for determining access control and the exchange of policies may occur between these systems, whilst they do not occur between the RBAC/ABAC-based authorisation systems. Furthermore, with the use of SAML messages, the existing RBAC/ABAC authorisation systems can be integrated into Web Services seamlessly through Globus or AAI (see section 2.7.1). However, the adoption of their protocols is not sufficient enough to help the existing TN-based authorisation systems to be seamlessly integrated into Web Services, as the ***syntactic***, ***semantic*** and ***functional*** interoperability issues in relation to TN-related communication cannot be provided by their protocols. To clarify the characteristics of these TN-based authorisation systems, the following begins with an introduction of the first TN-based authorisation system.

TrustBuilder is the first implemented module-based system for using TN in open distributed systems such as for Web-based transactions (Barlow et al., 2001; Winslett et al., 2002; Seamons et al., 2003; Basney et al., 2004). There are three modules (credential verification module, negotiation strategy module and policy compliance checker) designed within this system. Two different compliance checkers have been implemented in TrustBuilder. One can compare X.509v3 tokens (used as the credentials) with policies written in an XML-based language. The other is able to compare RT credentials with the relevant policies. Two communication protocols including HTTPS and TLS are used in this system. It is accepted that TrustBuilder is the most influential TN-based authorisation system, even though there are weaknesses within this system (Bertino, Ferrari and Squicciarini, 2004a).

For instance, Hess et al. (2004) identify that disclosure of sensitive credentials cannot be protected by using this system. They then propose a general access control model that can be implemented in TrustBuilder. This model is able to check the sensitivity of credentials relevant to local policies. Ryutov et al. (2005) use TrustBuilder along with Generic Authorisation and Access-Control API to aid in defending against DoS Attacks. TrustBuilder is even used as a third-party system to help two entities achieve authorisation to relieve the performance pressure (Olson et al., 2006; Lee et al., 2006). As TrustBuilder supports the use of the PRUNES - a tree-based model (discussed in section 3.3.1.2), a simple protocol is designed within this system, and its protocol messages can support the disclosure tree model (Yu, Winslett and Seamons, 2001). This model is extended in the TN service called HiTrust (Li, Li and Meng, 2010). However, the inflexibility of the addition of new credential, policy languages and strategies and the in-built coarsely grained policy languages are the intrinsic limitations of TrustBuilder (Lee, Winslett and Perano, 2009; Zhang et al., 2009).

Trust-X is developed as an XML-based framework for using TN in a peer-to-peer environment (Bertino, Ferrari and Squicciarini, 2004a). As the comparison of credentials against policies is one of the required functionalities within TN, there is no surprise that the policy compliance checker is a core module designed within this framework. Apart from this module, a sequence prediction module is introduced. This module provides a service to “keep track [of] the sequence of certificates more often changed, instead of recalculating them for each negotiation” (Bertino, Ferrari and Squicciarini, 2004a), if different counterparts request to access the same resource. Trust tickets are developed as a new feature to TN. Once TN is successful, each entity will generate an issued trust ticket and send it to the counterpart.

The functionality of a trust ticket is similar to that of a cookie stored in a Web browser to avoid repeated authorisation in a certain period. In other words, with the use of trust tickets, two entities can avoid using TN to achieve authorisation during the valid period of the tickets. Further functionalities such as support for multiple credential languages and the use of P3P policies to protect the disclosure of sensitive credentials are added in this system (Squicciarini et al., 2007). Another functionality supporting multisession operation allowing two entities to pause TN and then to resume TN in the same session is also added to the system. This functionality allows an entity to delegate its right to a third entity to continue TN with the counterpart (Squicciarini et al., 2012). However, the only recognition of the X-TNL language (expressing credentials, policies, trust tickets) is its main limitation, so that it cannot be interoperable with other TN-based authorisation systems as stated by Lee, Winslett and Perano (2009).

Both TrustBuilder and Trust-X can only support a limited number of policy compliance checkers and strategies, and their architectures are not flexible enough to support new languages (expressing credentials and policies) and strategies. In addition, there is a need to redesign the architecture of a TN-based authorisation system to address this issue as stated by Lee, Winslett and Perano (2009). They thus propose TrustBuilder2, a reconfigurable architecture adopting plug-in modules. In such a design, new languages and strategies can be added as new plug-ins to the relevant modules such as the policy compliance checker module and the strategy module etc. As the architecture of TrustBuilder2 is a redesign of TrustBuilder, the functionalities supported in TrustBuilder can also be used in it. TrustBuider2 can even serve TN among multiple entities (Orkphol and Li, 2012).

Trust-Serv is a framework proposed by Skogsrud, Benatallah and Casati (2003, 2004a, 2004c) and Skogsrud et al. (2004b. 2009). Its specific benefit is to provide lifecycle management of policies and dynamic policy migration. To achieve the dynamic policy migration, three migration strategies are introduced. The first strategy "lets negotiations in progress be completed according to the old policy, but requires all new negotiations to follow the new policy" (Skogsrud, Benatallah and Casati, 2003). The second strategy requires the existing TN to migrate to the new policy instead of the old one, if there are common rules required in the negotiation, and the common rules can be found in both the new policy and the old policy. The last strategy is the worst-case strategy that demands all of the negotiation in progress to terminate. Trust-Serv has been implemented in Web Services by using SAML messages for expressing credentials, and WS-SecurityPolicy for expressing policies. However, the only support for SAML credentials and WS-SecurityPolicy policies may restrict its ***capability*** interoperability with other TN-based authorisation systems used within Web Services.

Apart from the TN-based authorisation systems mentioned above, there are other models, frameworks and systems designed for TN, but the motivations for these system developments are not strong enough through the assessment of their rationales. For instance, the work presented in Andro (2010) is only a review of the existing TN models with no critique. Chen and Jiang (2011) introduce their own policy and credential languages with no critical assessments of the existing languages. Yu et al. (2011) propose their TN mechanism to improve the efficiency of using TN according to the history, but its idea is the same as the sequence prediction module used in Trust-X.

There are other research works in relation to TN-based authorisation systems. In these works, system designs are presented only to show how they can process their own specific policy languages with detailed syntax and semantics. For instance, several TN-based authorisation systems used in (Gavriloaie et al., 2004; Olmedilla et al., 2004; Nejdl, Olmedilla and Winslett, 2004) are specifically designed to process policies written in the PeerTrust language, which is based on Definite Horn clauses used for logic programs. Chen and Jiang (2011) develop a TN-based authorisation system to mainly serve a policy language called AATN-Jess. The TN-based authorisation system called PROTUNE developed by Bonatti et al. (2010) mainly uses its own language, which is based on normal logic program rules.

In terms of the use of these TN-based authorisation systems within Web Services, it is feasible to adopt these systems in the context of Web Services to provide TN as an optional access control approach for different Web Services. Due to the heterogeneous nature of Web Services, there is a high probability that two Web Services may use different authorisation systems to communicate with each other.

Under this phenomenon, unfortunately, several interoperability issues may arise. With respect to provision of interoperability for communication between two different TN-based authorisation systems within Web Services, TN-based authorisation systems such as TrustBuilder2, Trust-X cannot provide interoperability in relation to ***syntax*** and ***semantics***, as their messages are not designed in an XML-based structure. In addition, provision of interoperability in relation to ***capability*** and ***strategy*** is not taken into consideration in these TN-based authorisation systems either. Trust-Serv is a TN-based authorisation system implemented in Web Services, so it can provide interoperability in relation to ***syntax*** and ***semantics***, but it fails to provide interoperability in relation to ***capability*** and ***strategy***. In addition, for all of these TN-based authorisation systems, interoperability in relation to ***functionality*** cannot be guaranteed either, as some unique features designed in one system (e.g. trust ticket designed in Trust-X) cannot be supported by other systems (e.g. TrustBuilder2). In addition, with the adoption of TN-based authorisation systems to Web Services, communication between a TN-based authorisation system and an ABAC-based authorisation system is also possible in Web Services. Unfortunately, interoperability issues in relation to ***syntax***, ***semantics***, ***functionality***, ***capability*** and ***strategy*** may arise between them. To demonstrate these interoperability issues, the next section presents different circumstances of a case scenario described in section 2.8.

## 3.4 Interoperability Issues between Authorisation Systems in Web Services

To illustrate the above-mentioned interoperability issues, the background information of the case scenario described in section 2.8 is used here. Four circumstances of this case scenario are presented to explain the existence of the relevant interoperability issues. In particular, circumstance 1 demonstrates interoperability issues between an ABAC-based authorisation system and a TN-based authorisation system. Circumstances 2, 3 and 4 demonstrate interoperability issues between two TN-based authorisation systems.

Circumstance 1

WSA uses an ABAC-based authorisation system, and WSB uses a TN-based authorisation system. As there are different strategies used in TN, WSB randomly chooses the eager strategy. The communication process is shown below:

**Step 1:** WSA sends a request message to WSB informing WSB that Alice wants to access Bob's resource.
**Step 2:** As WSB uses a TN-based authorisation system, and the eager strategy has been randomly chosen, it then sends back Bob's non-sensitive credentials to WSA.

As the internal structure designed within the ABAC-based authorisation system used by WSA is supposed to receive policies at this step, it does not know how to process the received credentials. The reason that WSA cannot process the credentials is due to the fact that it lacks knowledge of ***syntax*** and ***semantics*** of the credentials and those of their protocol messages along with the relevant ***functionality.*** More precisely, lacking knowledge of the existence of different ***strategies*** causes the lack of the above knowledge. Thus, interoperability issues occur at this step, and the communication stops here.

This circumstance can demonstrate that the interoperability issues in relation to ***syntax***, ***semantics***, ***functionality*** and ***strategy*** may occur between an ABAC-based authorisation system and a TN-based authorisation system.

<u>Circumstance 2</u>
WSA uses a TrustBuilder2 system, and WSB uses a TrustServ system. Assume that the TrustBuilder2 and the TrustServ systems could understand the syntax and semantics of the protocol messages of each other, and they both use the parsimonious strategy to run TN. The policy compliance checker of TrustBuilder2 could (1) compare credentials written in the X-TNL language against policies written in the PeerTrust language (see section 3.3.2) and (2) compare credentials written in the $RT_0$ language (see section 3.3.2) against policies written in the Ack policy language (see section 3.3.2) for expressing policies, whereas the policy compliance checker of TrustServ could only compare credentials written in the X-TNL language against policies written in the PeerTrust language.

When TrustBuilder2 sends a credential written in the $RT_0$ language (see figure 3.1) or a policy written in the Ack policy language to TrustServ, an interoperability issue in relation to ***capability*** will occur. This is because the policy compliance checker designed in TrustServ could only recognise credentials written in the X-TNL language and policies written in the PeerTrust language. However, observing the conditions held by two systems, a potential TN without this interoperability issue could have occurred, if TrustBuilder2 had sent such a credential written in the X-TNL language or a policy written in the PeerTrust language.

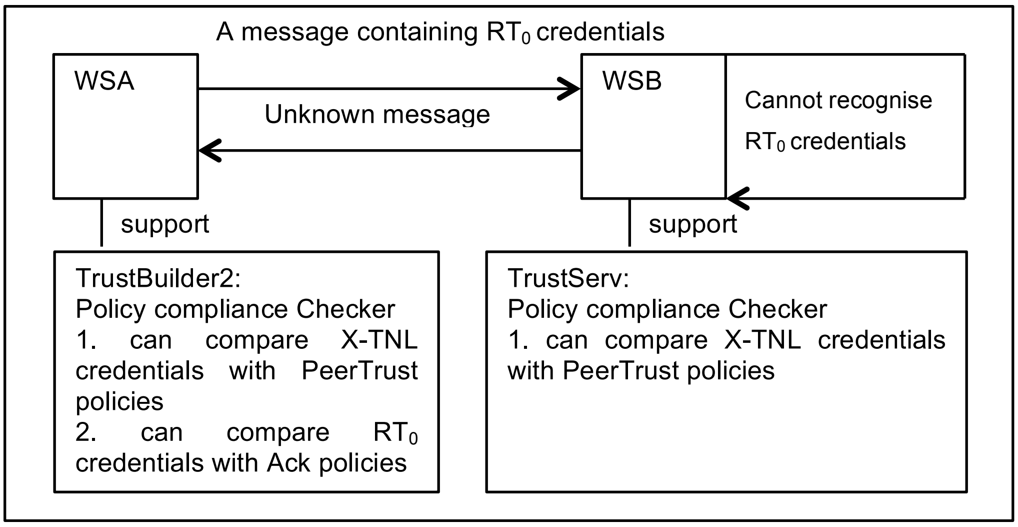

A message containing RT0 credentials
WSA
WSB
Unknown message
Cannot recognise RT0 credentials
support
support
TrustBuilder2: Policy compliance Checker 1. can compare X-TNL credentials with PeerTrust policies 2. can compare RT0 credentials with Ack policies
TrustServ: Policy compliance Checker 1. can compare X-TNL credentials with PeerTrust policies

Figure 3.1. Illustration of an interoperability issue in relation to capability in a first circumstance

This circumstance can demonstrate that the ***capability*** interoperability issue may occur between two TN-based authorisation systems. In fact, there is another circumstance that the ***capability*** interoperability issue will also occur. This circumstance is demonstrated in circumstance 3.

Circumstance 3

WSA uses a TrustBuilder2 system, and WSB uses a TrustServ system. Assume that the TrustBuilder2 and the TrustServ systems could understand the syntax and semantics of the protocol messages of each other, and they both use the parsimonious strategy to run TN. The policy compliance checker of TrustBuilder2 could (1) compare credentials written in the X-TNL language against policies written in the PeerTrust language and (2) compare credentials written in $RT_0$ against policies written in the Ack policy language, whereas the policy compliance checker of TrustServ could (1) compare credentials written in the X-TNL language against policies written in the Ack policy language, (2) compare credentials written in $RT_0$ against policies written in PeerTrust and (3) compare credentials written in X-TNL against policies written in PeerTrust.

After TrustBuilder2 sends a credential CA1 only written in the X-TNL language (see figure 3.2) to TrustServ, WSB will decide whether it is going to compare CA1 against a policy written in the Ack or in the PeerTrust language. WSB randomly chooses the policy PB1 written in the Ack language. After it compares CA1 against PB1, it decides to send PB1 to TrustBuilder2. Upon receiving PB1, as TrustBuilder2 cannot compare PB1 against CA1, as there is no CA1 written in the $RT_0$ language, a ***capability*** interoperability issue occurs resulting in failed communication.

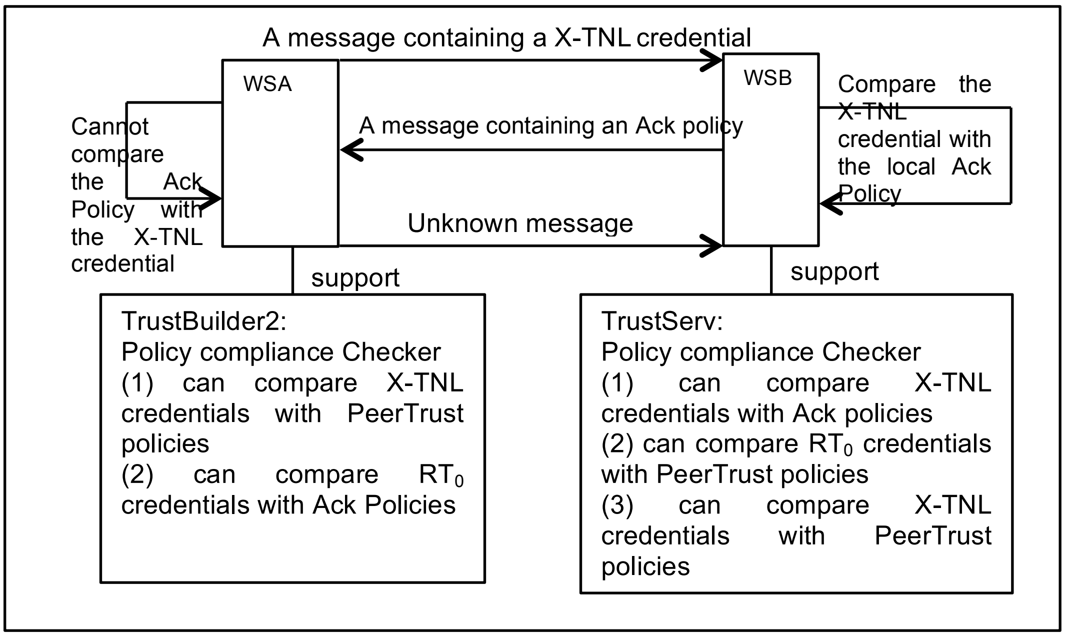

A message containing a X-TNL credential
WSA
WSB
Compare the X-TNL credential with the local Ack Policy
Cannot compare the Ack Policy with the X-TNL credential
A message containing an Ack policy
Unknown message
support
support
TrustBuilder2:
Policy compliance Checker
(1) can compare X-TNL credentials with PeerTrust policies
(2) can compare RT0 credentials with Ack Policies
TrustServ:
Policy compliance Checker
(1) can compare X-TNL credentials with Ack policies
(2) can compare RT0 credentials with PeerTrust policies
(3) can compare X-TNL credentials with PeerTrust policies

Figure 3.2. Illustration of the interoperability issue in relation to capability in a second circumstance

Observing the conditions held by two systems, a potential TN without the ***capability*** interoperability issue could have occurred if WSB had sent such a policy written in the PeerTrust language. The current solution used in the existing authorisation systems is to use the plug-in modules recommended in the TrustBuilder2 system (see section 3.3.4) for supporting different languages for credentials and policies. This solution can aid a Web Service in understanding as much policy languages as possible to increase its probability of successfully processing policies, if they are written in different policy languages. However, this solution cannot provide enough ***capability*** interoperability, since its benefit and limitation are the same as those of the object-oriented framework for adopting multiple policies used in the existing ABAC-based authorisation systems (see section 2.7.2).

Circumstance 4

WSA uses a TrustBuilder2 system, and WSB uses a TrustServ system. TrustBuilder2 could use the eager and parsimonious strategies, whereas TrustServ could only use the eager strategy. WSA randomly chooses to use the parsimonious strategy to conduct TN with WSB, which can only use the eager strategy. WSA owns a credential CA1, and treats this credential as sensitive, thereby declaring a policy PA1 protecting this credential. The policy PA1 requires a credential CB1. WSA wants to access a resource RB1 provided by WSB. WSB owns the insensitive credential CB1, and declares a policy PB1 protecting the resource RB1. The policy PB1 requires a requester to submit a credential CA1 to unlock this policy.

The process of communication for achieving authorisation between WSA and WSB are as follows:

**Step 1:** WSA initially sends a message to WSB for requesting an access to the resource RB1.

**Step 2:** WSB processes this message and returns a message containing the credential CB1 to WSA.

**Step 3:** WSA is expecting to receive a message containing policies at this particular time, as it is using the parsimonious strategy. In other words, even though WSA does possess the functionality for processing a message containing credentials, according to the used strategy, it is preparing to process a message containing policies at this time point. In other words, WSA has no idea of how to process the received credentials at this time point; it therefore has to treat this message as an unknown message.

However, observing the conditions held by two systems, a potential successful communication without the occurrence of this interoperability issue could have happened, if WSA had decided to use the eager strategy to conduct TN with WSB. This circumstance can demonstrate the occurrence of a ***strategic*** interoperability issue.

Concluded by the four circumstances, the interoperability issues between different authorisation systems within Web Services are listed in table 3.1 below, if TN-based authorisation systems are used into Web Services.

| **Reason causing the interoperability issue between different authorisation systems (i.e. ABAC, TN)** | **Existing Solutions used in different authorisation systems (i.e. ABAC, TN)** | **Weakness of the existing solutions used in different authorisation systems (i.e. ABAC, TN)** |
|---|---|---|
| A received message cannot be processed at a specific time point according to a specific ***strategy***, even a system has the relevant functionality for processing such a message | Nil | Nil |
| Not enough ***capability*** of the policy compliance checkers | Use an object-oriented framework or plug-in modules for supporting different languages of credentials and policies | There is a probability of missing interoperability |
| ***Functionality*** for processing the exchanged message is different in a ABAC-based authorisation system and an TN-based authorisation system, so credentials are not recognised by the a ABAC-based authorisation system as a message sender | Nil | Nil |
| Unknown ***semantics*** of messages | Use the SAML messages | Semantics supporting TN does not exist |
| Unknown ***syntax*** and of messages | Use the SAML messages | Syntax supporting TN does not exist |

Table 3.1. Interoperability issues between authorisation systems in Web Services

Within table 3.1, to provide interoperability between different authorisation systems (i.e. ABAC-based, TN-based) used in Web Services, the use of SAML messages in the AAI for ABAC-based authorisation systems, and the use of SAML messages and SecurityPolicy messages in Trust-Serv for TN-based authorisation systems are the existing solutions for providing ***syntactic*** and ***semantic*** interoperability. The plug-in approach designed in TrustBuilder2 for TN-based authorisation systems and the object-oriented framework leveraged in the Globus for ABAC-based authorisation systems for adopting multiple policies are the current approaches for providing

***capability*** interoperability. Unfortunately, these existing solutions cannot provide enough interoperability. In addition, ***functional*** and ***strategic*** interoperability issues also exist between different authorisation systems without a solution. Therefore, there is a strong need to explore a new solution that can make up for the weaknesses within the current approaches for supplying multiple interoperability between different authorisation systems in Web Services, if TN-based authorisation systems are to be used within Web Services.

## 3.5 An Improved Multi-layered Interoperability Model

Through an analysis of the critical review of the interoperability issues between authorisation systems (i.e. ABAC and TN) within Web Services in Chapter 2 and this Chapter, a variety of data has been collected demonstrating that the interoperability issues between authorisation systems is multiple rather than one. An overall understanding of the factors along with their characteristics causing the interoperability issues is helpful to aid in exploring a potential solution for addressing the second research problem. This section reviews the existing conceptual interoperability models of the most relevance, and proposes an improved multi-layered interoperability model (based on the data collected through the review in Chapter 2 and this Chapter).

The majority of the factors presented in the improved interoperability model can cover more characteristics causing the relevant interoperability issues that cannot be expressed by the factors presented in the existing interoperability models. In addition, one novel factor – strategic interoperability is also presented in the improved interoperability model, whereas it is neglected in the existing models. Although the improved interoperability model is constructed based on the data analysis in the context of Web Service authorisation systems, it may also be applied in other distributed systems environments to help researchers in academia and practitioners in the industry identify hidden interoperability issues between their systems. From this perspective, the improved interoperability model can be treated as a contribution as an extension of state of the art.

Existence of multi-layered interoperability has been illustrated in a conceptual model called Levels of Conceptual Interoperability Model (LCIM) proposed by Tolk and Muguira (2003). In this model, five layers of interoperability are presented including

- Layer 0: system specific data
- Layer 1: documented data
- Layer 2: aligned static data
- Layer 3: aligned dynamic data
- Layer 4: harmonized data

This five-layered model is further extended in the work conducted by Turnitsa (2005), in which seven layers of interoperability are presented. The seven layers are

- Layer 0: no interoperability
- Layer 1: technical interoperability - requires an information exchange between systems
- Layer 2: syntactic interoperability - requires a common format of the exchanged information between systems
- Layer 3: semantic interoperability - requires the meaning of the exchanged information between systems
- Layer 4: pragmatic interoperability - requires methods and procedures of systems to obtain the data from the exchanged information through the correct understanding of their semantics
- Layer 5: dynamic interoperability - requires the systems to understand the effect of the exchanged information, even though data used in the exchanged information are dynamically changed
- Layer 6: conceptual interoperability - requires a meaningful abstraction of communication in reality

Observing the two models, it can be identified that layer 0 in the two models indicates no interoperability; thereby excluding this layer from the model would not affect the provision of interoperability. Layer 1 in the two models suggests that the use of protocol messages is a proper solution for providing interoperability at this layer. Note that, in both models, they originally suggest that the use of a protocol is the solution for providing technical interoperability. However, after comparing the notion of the protocol they mention against the accepted notion of the protocol identified in section 2.3, it is identified that protocol messages is what they mean. In addition, other solutions are required to provide interoperability for other higher layers. For instance, Ontology can be used as an approach for providing interoperability at layer 2 in the first model, and at layer 3 in the second model respectively. Other approaches such as Unified Modelling Language (UML) or XML (discussion of XML is in section 2.4.2) are also suggested as solutions for ensuring interoperability at the dynamic layer and the syntactic layer in the second model respectively.

Unfortunately, when adopting the conceptual model proposed by Turnitsa (2005) within Web Services, the layers presented in this model cannot precisely cover all identified factors along with the characteristics causing the interoperability issues between authorisation systems within Web Services. For instance, as demonstrated in the case scenario (see section 3.4), the ***strategic*** interoperability issue is demonstrated in circumstances 1 and 4, but this factor is not presented in Turnitsa's model. In circumstances 2 and 3, to enable a system to deal with the comparison of its local credentials against remote policies, it requires its policy compliance checker to own the ***capability*** of not only recognising both the credential and policy languages, but also equipping the functionality for comparing the credentials written in the recognised language against the policies written in the recognised language. This capability factor along with its characteristic cannot be covered by the dynamic factor presented in Turnitsa's model either.

Table 3.2 presents an improved multi-layered interoperability model for illustrating multi-layered interoperability. This model (the right column of the table) is established based on the improvement of Turnitsa's model (2005).

| **Layered Interoperability in** Turnitsa's model (2005) | **Layered Interoperability between authorisation systems within Web Services** |
|---|---|
| 6 – Conceptual Interoperability | Layer 7 – Conceptual Interoperability |
| Nil | Layer 6 – Strategic Interoperability |
| 5 – Dynamic Interoperability | Layer 5 – Capability Interoperability |
| 4 – Pragmatic Interoperability | Layer 4 – Functional Interoperability |
| 3 – Semantic Interoperability | Layer 3 – Semantic Interoperability |
| 2 – Syntactic Interoperability | Layer 2 – Syntactic Interoperability |
| 1 – Technical Interoperability | Layer 1– Connected Interoperability |
| 0 – No Interoperability | Nil |

Table 3.2. An improved multi-layered interoperability model

Layer 1: This layer of the model concentrates on ensuring that a connection shall be established between systems. This connection is not dependent on a particular form of technical architecture, as multiple different approaches can be provided for connection to occur. It is assumed that at this layer of the model, if a connection can be established between two systems, then this is the underlying infrastructure required for message exchange. Turnista's model (2005) concentrates on the notion of technical interoperability; however, just having the technology in place does not necessarily mean that two systems can connect to each other. Therefore the term connected is selected as recommend by DoD (1998).

Layer 2: Syntactic interoperability is supplied at this layer, which is the same as layer 2 mentioned in Turnitsa's model (2005). At this layer, there is a requirement that messages exchanged between communicating systems should be defined in a common format, language or structure. For instance, in the context of Web Services, the mandatory use of XML for forming the structure of the exchanged messages is indispensable (see section 2.4.2).

Layer 3: Once syntactic interoperability has been achieved, there is a need to ensure that systems can understand the material exchanged. Therefore taking the same approach as Turnista's model (2005), Layer 3 concentrates on semantic interoperability. The meaning of the exchanged messages is provided through a precise definition of their semantic meaning. There needs to be an agreement between the systems on the semantics of the exchanged messages to ensure that they can process the same exchanged message in the same way. NOTE: This layer of semantic agreement is focused primarily on messaging infrastructure. Problems may arise at higher layers of the model with differences in the meaning of data items e.g. different atomic data or compound data expressing the same meaning (see layer 5).

Layer 4: This layer focuses on ensuring that systems can process data within the exchanged messages. It also focuses on each system being able to react appropriately to the exchanged messages such that all input and output is expected. Both communicating systems therefore need to understand the effect of processing on the exchanged messages. The functionalities of each system are implemented in the relevant methods, structures or procedures. In Turnista's model (2005), the term pragmatic focuses more on practical interoperability. However, from the analysis above, at this layer, interoperability is more related to a system's understanding of the effect of processing data within the exchanged messages. This understanding as knowledge of a system is implemented through its internal functionalities. From this perspective, the term functional is more suitable, as it can convey this meaning more precisely.

Layer 5: This layer ensures capability interoperability, which is revised from Layer 5 – dynamic interoperability of Turnista's model (2005). Dynamic interoperability in Turnista's model features dynamic changes of data within a system over time. This interoperability issue caused by dynamically updating data normally occurs when two similar systems are communicating. For example, two entities may both use the same system (e.g. PERMIS, Akenti), but there may be differences in their update cycles such that one system (the message sender) may update data every two hours, whereas the other system (the message receiver) may update data every twenty-four hours. As the message sender updates its data more often than the message receiver, new data used in the message sender cannot be recognised by the message receiver, thereby causing an interoperability issue.

However, through a further analysis of this example in depth, it can be identified that dynamic changes of data is just a phenomenon on the surface causing the interoperability issue between two systems. The underpinning reason is the different levels of capability for recognising the syntax and semantics of data (can be atomic or compound data) within the two systems (Tolk, Diallo and Turnitsa, 2007; Lang et al., 2006). There is a distinction between capability and functionality from the perspective of a system. The functionality only ensures that two systems can have the same knowledge about the procedure of processing data, whereas two systems may possess different levels of capability in relation to the same functionality.

For example, in circumstance 3 (see section 3.4), the existing TN-based authorisation systems such as TrustBuilder2 or TrustServ can both provide the same functionality for comparing credentials against policies for reaching an access control decision. In terms of different levels of capability of recognising the syntax and semantics of the data (processing compound data requires an understanding of both syntax and semantics, whereas processing atomic data requires an understanding of semantics only) in relation to this functionality, TrustBuilder2 has the capability of comparing credentials against policies written in two languages (both credentials and polices are compound data), whilst Trust-X might only be able to compare credentials against policies written in one specific language. In this example, a dynamic change of data in systems is not the phenomenon that can cause the interoperability issue at this layer. Instead, the occurrence of the interoperability issue is due to the need that two different systems with similar functionalities want to communicate. Through the analysis of the two examples shown above, possessing different levels of capabilities of two systems for recognising the syntax and semantics of data is the common reason causing the interoperability issue. The term capability is therefore more appropriate to reflect the cause for the occurrence of the interoperability issue at this layer in comparison with the term dynamic.

Layer 6: This layer is a novel layer guaranteeing strategic interoperability. A strategy of a system controls the system behaviour, this can include how data is processed within a system and when and what exchanged messages the system should send out and expect to receive (Winsborough, Seamons and Jones, 1999, 2000). Within the existing systems, a strategy is embodied in the connected processes of a system (one process of a system defined in this Thesis is that a system will receive (send out) one incoming (outgoing) message and send out (receive) one piece of outgoing (incoming) message if necessary). Currently, observing the process of the existing

RBAC/ABAC-based authorisation approaches, the majority of the communication between these systems implicitly includes one strategy. In other words, the order for exchanging messages between systems is fixed, thereby not incurring an interoperability issue in relation to strategy. However, with the development of technologies, systems will be becoming more and more intelligent so that they may have a variety of alternative strategies for achieving the same task. An example of the existence of multiple strategies supported in one approach is TN (see section 3.3.1). If two systems have multiple strategies, and if there is no proper solution to help a system dynamically switch from one strategy to another strategy, an exchanged message sent from one system may not be expected to be received by the counterpart in the correct time, even though interoperability at all of the lower layers are provided. Eventually, the counterpart's incapability of processing the unexpected message will cause an interoperability issue at this layer.

Layer 7: Conceptual interoperability is the highest layer between communicating systems. Interoperability at this layer requires communicating systems to understand the whole concept of the communication. This concept can include a specific purpose/objective/goal or context enabling a system especially a service requester to decide the proper time of using a specific protocol. Other unique characteristics in relation to specific communication (e.g. strengths and limitations of the communication) should also be provided in this concept. In particular, conceptual interoperability is needed, when multiple optional protocols can be used for achieving a same purpose (e.g. either an ABAC-based or a TN-based approach be help a service requester make an access control decision). In such an instance, it requires a service requester to know the difference between the multiple protocols at a conceptual layer so as to choose the most appropriate protocol.

## 3.6 Related Work

In addition to the existing interoperability models mentioned in section 3.5, which aim to present interoperability between two communicating systems, there are other interoperability models. As they focus on identifying interoperability issues amongst multiple systems rather than two communicating systems, these other interoperability models are considered as of less relevance.

For instance, the Levels of Information Systems Interoperability (LISI) model (DoD, 1998) is a widespread model, which has been acknowledged as a foundation model for assessing interoperability amongst multiple types of systems in different domains of an enterprise (Rezaei et al., 2014). In addition, a complex matrix called LISI Capabilities Model for assessing interoperability between systems is also designed. Observing the LISI Capabilities Model, it covers a large number of aspects such as security-related policies, software, hardware etc. Therefore, it is suitable to be used in scenarios where a new enterprise needs to be established requiring interoperability amongst different types of systems (including hardware, software etc.).

Clark and Jones (1999) identify that the LISI Capabilities Model is insufficient to assess interoperability amongst Command and Control Support (C2S) systems; they therefore present an Organisational Interoperability Maturity Model. Hamilton, Rosen and Summers (2002) identify the complexity and inappropriateness of the utilisation of the LISI Capabilities Model for measuring interoperability for legacy systems. They therefore propose a matrix-based metric called Stop Light Model for measuring the degree of interoperability amongst legacy systems. A Europe Integrated Project called Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications Integrated (ATHENA IP) introduces an Enterprise Interoperability Maturity Model for performing assessments on interoperability maturity for enterprise-level systems (ATHENA, 2005). Mykkanen and Tuomainen (2008) propose a conceptual framework for evaluating interoperability of standards in order to help system developers select the appropriate standards based on the evaluation results. Rezaei, Chiew and Lee (2014) present an interoperability model for assessing interoperability amongst multiple ultra large-scale systems.

## 3.7 Research Problem

Although the improved interoperability model can present all the identified key factors along with their characteristics causing the interoperability issues between authorisation systems within Web Services, it does not indicate any solution providing interoperability for each relevant factor presented at each layer. Observing the four circumstances presented in section 3.4, factors causing interoperability issues may occur together (e.g. in circumstance 1 and 4, the strategy interoperability issue causes the occurrence of the interoperability issues at lower layers). The existing interoperability model – LCIM – suggests some solutions (e.g. XML, protocol messages, ontology, UML), but each of which can only provide interoperability for a relevant layer. In other words, the lack of a unified solution integrating these solutions for providing interoperability for all different factors still remains as a problem.

## 3.8 Discussion of Potential Solutions

In terms of the potential solutions, the first straightforward solution might be to force all of the Web Services to implement the same TN-based authorisation system (e.g. TrustBuilder2). Unfortunately, due to the heterogeneous nature of Web Services, this assumption is not valid, as each TN-based authorisation system has its own unique benefits that are not provided in other systems. Another potential solution might be to integrate the functionalities of all of the TN-based authorisation systems into one system along with some interpretation mechanisms for translating different languages to the specific language implemented within the system. This would require a new framework to integrate all of the TN-based authorisation systems together seamlessly. However, due to the time and budget issues, this solution is still not ideal enough for different organisations. A third possible solution might be to design and develop a system operating as an intermediate system for providing TN/ABAC-related authorisation service for any two Web Services. However, this solution would require the design of a communication mechanism amongst three entities (a service requester, a service provider and a system providing this service). Eventually, the design of such a communication mechanism would be more complex than the communication between two systems.

After a thoughtful consideration of the balance between effectiveness and feasibility of the potential solution in the context of Web Services, the selected solution is to design a protocol-based approach by integrating necessary functionalities relevant to TN into authorisation systems within Web Services. There is a distinction between the use of a system and the use of a protocol. A system is normally used as a whole, whereas the use of a protocol as a whole is not necessary. In other words, as a protocol consists of its sub elements such as protocol messages and internal structures (see section 2.3), system developers can add partial protocol messages and internal structures to the existing authorisation systems, if the partial protocol messages and internal structures are not yet supported in the existing authorisation systems. With the use of this solution, it is not necessary for a Web Service to implement a whole new protocol.

## 3.9 Chapter Summary

This chapter has thoroughly reviewed the state-of-the-art TN from different aspects to the existing TN-based authorisation systems. After analysing each aspect of TN, relevant interoperability issues have been identified. Through the use of different circumstances of a case scenario, the occurrence of the identified interoperability issues between authorisation systems within Web Services has then been demonstrated. Upon identifying the possible factors along with their characteristics causing the interoperability issues, an improved multi-layered interoperability model is proposed. As there is a lack of a unified solution that can provide interoperability for all of the factors presented in the improved interoperability model, a discussion of potential solutions is provided. Eventually, a decision was made that a protocol-based approach might be the most appropriate solution. So, the next chapter introduces a relevant conceptual multi-layered interoperability-solution design along with an improved TN protocol that can be used for addressing the research problem.

# Chapter 4. A Protocol-based Approach for Providing Interoperability between Authorisation Systems within Web Services

## 4.1 Introduction

Multiple interoperability issues between authorisation systems (which includes the use of TN-based authorisation systems) within Web Services have been identified through an analysis of the review in Chapter 2 and Chapter 3, and also been demonstrated in a case scenario consisting of four circumstances shown in section 3.4. According to the potential solutions discussed in section 3.8, a protocol-based approach is considered to be the most effective solution to investigate that may be able to address the second research problem as outlined in Chapter 1.

The design and development of such a protocol requires a clear mapping between each factor presented at each relevant layer of the improved interoperability model (shown in table 3.2) and the proposed solution design. Therefore, this mapping as a novel solution is presented in a conceptual multi-layered interoperability-solution design as shown in table 4.1. This mapping provides an overview of how a protocol-based approach can provide multi-layered interoperability to provide interoperability for Web Service authorisation models. An improved TN protocol is produced as a concrete example of utilising the interoperability-solution design.

To ensure the correctness of the improved TN protocol design, this research applies a protocol design and development methodology called validation (Merlin, 1976; Bochmann and Gecsei, 1977; Merlin, 1979). This methodology indicates that the process of the protocol design and development should be iterative. The determined process (shown in figure 4.1) used in this protocol design and development methodology includes three steps:

- Step one: Protocol requirements elicitation (Merlin, 1979; Sunshine, 1979a; Bochmann and Sunshine, 1980; Sunshine et al., 1982; Mitra and Lafon, 2007);
- Step two: Protocol design and development (Merlin 1976);

- Step three: Protocol verification (Bochmann and Gecsei, 1977; Bochmann, 1978; West, 1978; Sunshine, 1979b; Sunshine et al., 1982).

The process consisting of the above three steps forms a cycle that can be conducted iteratively. Each iterative cycle enables protocol designers to obtain insight into the problems that may not be discovered in the previous cycles. This process will finish when the protocol designers are satisfied that the protocol can provide the specific service for the communicating entities. With the use of this methodology, a protocol can always be improved through the repeated process, so the latest version of a protocol can be different from a previous version to some extent. This iterative improvement cycle also indicates that a protocol design may never be complete. That is, as long as the relevant new requirements arise, a protocol needs be modified to cater for new functionalities.

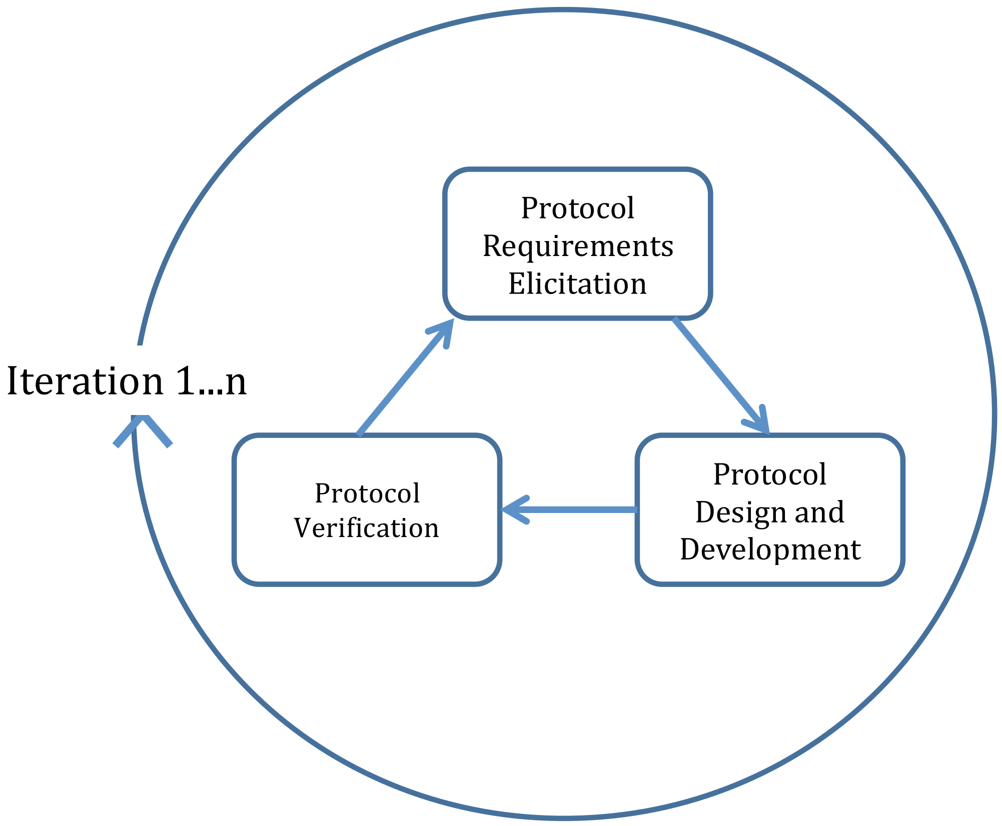

Protocol Requirements Elicitation
Protocol Design and Development
Protocol Verification
Iteration 1…n

Figure 4.1. A protocol design and development methodology –Validation

Following step one of this methodology, specific requirements for a protocol design are presented. These requirements are indispensable, since they are strictly connected with a specific service, so they provide guidelines for the design and development of a specific protocol (Merlin, 1979; Sunshine, 1979a; Bochmann and Sunshine, 1980; Sunshine et al., 1982). There is a major difference between the purpose of the protocol design and development methodology adoption and the requirements of the protocol design and development. The methodology can be applied to the design and development of protocols in general, regardless of the specific services they provide, whereas the requirements relevant to a specific service can only be useful for the design and development of a specific protocol that will provide this service.

Following the discussion of potential solutions for providing interoperability in the conceptual multi-layered interoperability-solution design in table 4.1 and the developed TN service-related requirements, an improved TN protocol is then presented (discussed in sections 4.4, 4.5, 4.6 and 4.7) according to step two of the protocol design and development methodology. This protocol is designed to provide multi-layered interoperability between authorisation systems within Web Services.

## 4.2 Overview of A Protocol-based Solution Design

Table 4.1 presents a conceptual multi-layered interoperability-solution design by using a protocol-based approach. The novelty of this design is in developing an understanding of how a protocol can be used to provide multi-layered interoperability between two systems at the majority of the layers of the model outlined in table 3.2. This solution improves upon existing solutions in the existing research such as Tolk and Muguira (2003), Turnitsa (2005), Wang, Tolk and Wang (2009), as they state that a protocol can only provide syntactic interoperability (layer 1 in the improved interoperability model shown in table 3.2). In other words, interoperability at higher layers cannot be supplied by a protocol, but can be provided by other solutions such as ontology or UML (see section 3.5).

The researcher of this Thesis agrees that other solutions can provide interoperability between communicating systems, but they should be used as components of a protocol rather than as individual solutions in isolation. The different perspectives over the use of a protocol as a solution between other research and this research are attributed to the different understanding on the notion of a protocol. Researchers such as Tolk and Muguira (2003), Turnitsa (2005) and Wang, Tolk and Wang (2009) state that a protocol only consists of exchanged messages for communication. More precisely, they understand the notion of a protocol as only protocol messages as a part of the accepted notion of a protocol (see section 2.3). However, the researcher of this Thesis supports the accepted notion of a protocol, which may not only include the protocol specification including protocol messages and internal structures, but also include the service specification providing abstract service-related information.

To use the protocol-based approach, four steps are required.

- **Step 1**: design a protocol following the solution design (shown in table 4.1). In addition, the protocol design should also follow the protocol design methodology (shown in figure 4.1).
- **Step 2**: compare this protocol against the protocols used within the existing authorisation systems to identify the distinctions. It is a prerequisite that this comparison has been applied to both communicating systems due to the nature of interoperability. More precisely, interoperability issues occur as long as one system cannot understand the communication. Therefore, application of a protocol to only one communicating system is insufficient to supply enough interoperability.
- **Step 3**: find the differences between two protocols. Elements (i.e. protocol messages and internal structures along with functionality components) of this protocol should then be added to the existing authorisation systems, if the elements cannot be identified within the protocol of the existing authorisation systems. In other words, if the relevant elements have been supported within the existing authorisation systems, there is no need to add the elements designed within this protocol to the existing authorisation systems.
- **Step 4**: let the two systems communicate by running the designed protocol.

According to the solutions listed in table 4.1, only five of the layers indicated in the improved interoperability model (from layers 2 to 6) can be ensured with the use of a protocol. Currently wired or wireless technologies are the main approaches for ensuring interoperability at layer 1 - connected interoperability (DoD, 1998). In terms of interoperability at layer 7, potential solutions are generally more related to how to enable communicating systems to understand the entire concept of the communication. It is identified that relevant conceptual information can be provided in the service specification of a protocol, readers of which are currently mainly protocol developers. Upon understanding the conceptual information of different protocols providing similar services, developers may implement all of them within a system. When a system needs to communicate with a counterpart with several optional protocols, it requires this system to select the most appropriate protocol based on the understanding of each protocol conceptually. Unfortunately, this conceptual understanding may not be achieved with the use of this solution design, so other solutions may be required along with the protocol to provide conceptual interoperability for the communicating systems. Therefore, a protocol on its own cannot be sufficient enough to provide interoperability at layer 7.

| **Interoperability** | Solutions in a Protocol |
|---|---|
| Layer 7: Conceptual Interoperability | Provide the purpose and contextual information in which a specific protocol should be used, but a system's understanding of the correct usage of the protocol requires other potential solutions (e.g. ontology) |
| Layer 6: Strategic Interoperability | Awareness of the interoperable strategies (realised through internal structures and protocol messages) |
| Layer 5: Capability Interoperability | Awareness of the common capabilities (realised through internal structures and protocol messages) |
| Layer 4: Functional Interoperability | Common functionalities designed in the internal structures |
| Layer 3: Semantic Interoperability | Common semantics defined in the protocol messages |
| Layer 2: Syntactic Interoperability | Common syntax defined in the protocol messages |
| Layer 1: Connected Interoperability | Nil |

Table 4.1. A conceptual multi-layered interoperability-solution design

Interoperability at layer 2 – syntactic interoperability and layer 3 – semantic interoperability can be assured through the definition of common protocol messages as the first part of the protocol specification (see section 2.3). Within Web Services, the mandatory use of XML is the cornerstone for providing syntax or structure of any protocol messages (Bray et al., 1998; Klein, 2001). In terms of semantics of protocol messages, the involvement of human developers is still required for understanding the meaning of messages (Paolucci and Sycara, 2003; Nezhad et al., 2006). Therefore, presenting the meaning of messages in a natural language is the common mechanism used in current WS-related standards such as WS-Security, WS-Trust etc. (Lawrence and Kaler, 2004; Lawrence and Kaler, 2009a).

Interoperability at layer 4 – common functional interoperability – can be guaranteed through the use of internal structures as the second part of the protocol specification (see section 2.3). Within the internal structures, different techniques such as ontologies (Hitzler et al, 2012), Object-Oriented Models (as Lang et al., 2006), plug-in approaches (Lee, Winslett and Perano, 2009) or UML (Wang, Tolk and Wang, 2009) can be used for not only supplying the functionalities but also different levels of capabilities in relation to the relevant functions. The purpose of the use of the internal structures is to ensure that each protocol message will be linked to its relevant internal structure. In other words, the effect and functionality for processing each protocol message is explicitly defined in the internal structures.

To use a protocol for providing interoperability at layer 5 – capability interoperability – and layer 6 – strategic interoperability – awareness of common capabilities in relation to the same functionality and that of interoperable strategies between two systems is required. This requirement is very important in Web Services. As discussed in section 2.4.5, there is a high probability that two unknown Web Services may communicate with each other (Garofalakis et al., 2006). In such a circumstance, without the knowledge of the common capabilities and interoperable strategies, communication between two Web Services may fail, even though successful communication between them is possible (demonstrations are given in the case study, see section 3.4).  To enable awareness between two systems, a communication process is needed. This process allows two unknown systems to consult with each other to ensure that the interoperability issues at layers 5 and 6 will not impede potential successful communication. To achieve the purpose of this process, it requires the combined use of the design of protocol messages and internal structures (i.e. protocol specification).

Interoperability at layer 7 – conceptual interoperability requires a system to understand which protocol should be used for achieving a certain task, when several optional protocols providing similar services are available. At this layer, the service specification of a protocol can only provide the information in relation to what service a protocol can provide and in what circumstance a protocol should be used. A system's understanding of this information for selecting the use of the most appropriate protocol may require other potential solutions (e.g. an ontology).

Following step one of the protocol design and development methodology, the next section presents the protocol requirements.

## 4.3 Protocol Requirements Elicitation

Protocol designers in both academic and industrial communities are agreed on the significance of the requirements for protocol design. As the requirements in relation to a specific protocol are representatives of different characteristics of the service the protocol can provide, the requirements are always produced first to guide the design and development of the specific protocol (Merlin, 1979; Sunshine, 1979a; Bochmann and Sunshine, 1980; Sunshine et al., 1982; Mitra and Lafon, 2007). In the research presented within this Thesis, the researcher also followed this process: presented the requirements, and then designed the protocol according to these requirements. It should be noted that these requirements were not identified together at the first time of designing the protocol. Instead, they were identified within different cycles of the protocol design and development process following application of the validation methodology.

Observing the interoperability issues between the authorisation systems within Web Services as discovered in table 3.2, including those additionally caused by the adoption of TN-based authorisation systems within Web Services, a series of requirements was developed. In addition to proposing solutions which remove interoperability concerns, a number of requirements are outlined, which provide additional TN functionalities within the conceptual protocol design. Therefore, the requirements elicited below mainly focus on the necessary and the most representative functionalities needed by a TN system.

Through the critical review of TN-related strategies presented in section 3.1, strategies designed for TN can be classified into two categories: non-policy-exchanged strategies and policy-exchanged strategies. As obtained from the critical analysis of pros and cons of these strategies, the eager strategy and the parsimonious strategy are identified as the typical representatives of each category. This produces the first set of requirements for protocol design and functionality design as shown below.

**Protocol Design Requirement 1:** *The two categories of strategies need to be supported by TN.*

**Functionality Design Requirement 1**: *Web Services should be able to support the use of the eager strategy and the parsimonious strategy as the representatives of two strategy categories respectively.*

As stated in table 4.1, to ensure interoperability at layer 6, strategic interoperability should be considered to enable two Web Services to identify an interoperable strategy. As stated in section 3.2, in terms of strategies designed for TN, some strategies may not be interoperable with one another, but at least each strategy is self-compatible (Yu, Winslett and Seamons, 2001). This produces the second set of requirements for protocol design and functionality design respectively as shown below.

**Protocol Design Requirement 2:** *If two Web Services are going to use TN, the protocol should allow them to reach an agreement on the use of a common strategy to ensure that the strategy interoperability issue will not affect the commencement of TN.*

**Functionality Design Requirement 2**: *Web Services should be able to discover whether there is a common strategy for TN between them.*

In order to allow two Web Services to agree on the use of interoperable strategies, all supported strategy names must be explicitly unveiled between two Web Services. The circumstance that a strategy may have different names should also be taken into consideration. For example, the eager strategy sometimes is referred to as the naïve strategy (Yu, Winslett and Seamons, 2001; Seamons et al., 2002b; Yu, Winslett and Seamons, 2003). This produces the third set of requirements for protocol design and functionality design respectively as shown below.

**Protocol Design Requirement 3:** *The protocol should be designed to allow a Web Service to explicitly inform the counterpart what strategies it can support.*

**Functionality Design Requirement 3:** *A Web Service should be able to discover a common strategy not only based on keyword matching, but also based on semantic matching.*

As identified in section 3.3.1, any one of the existing strategies differs from another strategy, since the conditions for using each strategy is different (e.g. the eager strategy cannot be used, when all of the sensitive credentials owned by a service requester are sensitive). This produces the fourth set of requirements for protocol design and functionality design respectively as shown below.

**Protocol Design Requirement 4:** *The protocol should allow a Web Service Requester to inform a Web Service Provider about the specific owner and resource it wants to request to enable a Web Service Provider to check what strategies can be used based on the conditions of the owner.*

**Functionality Design Requirement 4:** *A Web Service Provider should be able to obtain the conditions of any resource owner (can be either a human user or an organisation), whom it will be on behalf of, to determine the use of the most appropriate strategy.*

After strategic interoperability is ensured, interoperability at layer 5 – capability interoperability – should also be taken into consideration. This interoperability strictly relates to the abilities of the policy compliance checkers of both Web Services. As mentioned in the case study (see section 3.4), two circumstances (circumstances 3 and 4) can cause the capability interoperability issue at this layer, and capability interoperability in both circumstances, to develop in relation to the functionality of comparing credentials against policies. To provide interoperability in both circumstances, the solution designed in the protocol should not only enable two Web Services to identify the common languages for expressing credentials and policies, but also enable two Web Services to identify the common language combinations that can be processed by their policy compliance checkers. This produces the fifth set of requirements for protocol design and functionality design respectively as shown below.

**Protocol Design Requirement 5:** *The protocol should be designed to allow a Web Service to explicitly inform the counterpart what language combinations it can support.*

**Functionality Design Requirement 5:** *If there exist common language combinations within the list, a Web Service should be able to discover them. If more than one language combination can be supported, a Web Service should be able to choose the most appropriate language combination that can be processed by the policy compliance checkers more efficiently.*

To provide interoperability at layer 4 between authorisation systems in Web Services, a potential protocol requires two Web Services to have common functionalities. However, taking into consideration that the ABAC approach used in the current SAML authorisation protocol does not have enough functionalities (e.g. compare received credentials against local policies) as those used in TN, so the functionality interoperability issue will occur (see case study – circumstance 1 shown in section 3.4). This produces the sixth set of requirements for protocol design and functionality design respectively shown below.

**Protocol Design Requirement 6:** *The communication protocol should be designed to support the major representative functionalities (e.g. strategy component, policy compliance checker, verification of credential authenticity and credential chain) required by TN.*

**Functionality Design Requirement 6:** *Whenever a Web Service is designed to support this protocol, it should implement all of the major representative functionalities.*

As interoperability at layers 2 – syntactic interoperability – and 3 – sematic interoperability – are closely related to interoperability at layer 4 – functionality interoperability. The requirements for protocol message design and for functionality design are still shown together for ease of understanding.

As discussed in section 3.3.2, there exists a variety of credential and policy languages, and each of them possesses its own syntax and semantics. This requires the protocol messages to support the use of different credential and policy languages. This produces the seventh set of requirements for protocol design and functionality design respectively as shown below.

**Protocol Design Requirement 7:** *The syntax and semantics of the protocol messages should be flexible enough to support different language combinations (expressing credentials and policies) for TN rather than supporting only one specific credential language.*

**Functionality Design Requirement 7:** *The policy compliance checkers of* a *Web Service should at least support one language combination (expressing credentials and policies) for TN. It is ideal that a Web Service can support more than one language combination for TN to raise the probability that it can use TN with other Web Services.*

The discussion of strategies (see section 3.3.1) indicates that an uncertain number of credentials and policies may be disclosed in an outgoing message. For instance, with the use of the eager strategy, a service provider may disclose one insensitive credential at first, and then disclose multiple credentials at later steps. By contrast, with the use of the parsimonious strategy, an entity may disclose one or more policies in an outgoing message in different steps. This produces the eighth set of requirements for protocol design and functionality design as shown below.

**Protocol Design Requirement 8:** *The syntax and semantics of the protocol messages should support the disclosure of both single and multiple credentials as well as policies.*

**Functionality Design Requirement 8:** *Web Services should strictly follow the chosen strategy to disclose credential(s) or policy(ies), when they need to send local credentials containing the required attribute information or to send local policies containing the required attribute information to the counterpart.*

As stated in section 3.3.2, whenever an entity discloses its credential(s) or policy(ies), they should contain relevant information to inform the communicating counterpart (Yu, Winslett and Seamons, 2003). This produces the ninth set of requirements for protocol design and functionality design as shown below.

**Protocol Design Requirement 9:** *The syntax and semantics of the protocol messages for sending credentials should contain the information such as the name or type of each credential, the owner of each credential, and which policy(ies) the disclosed credential(s) can fulfil, if it is needed (e.g. the use of the parsimonious strategy* *requires this information). Likewise, the syntax and semantics of the protocol messages designed for sending policy(ies) should also contain the necessary information.*

**Functionality Design Requirement 9:** *Both Web Services involved in TN should understand the protocol messages expressing credentials and policies.*

Within a TrustBuilder system (see section 3.3.4), status information about policy satisfaction (whether the received credentials from the counterpart can fulfil its local policies) of each entity is supported. In other words, whenever an entity identifies whether the counterpart's credentials can fulfil its local policies, it will store this status locally, and will inform the counterpart. However, it can be argued from the perspective of the process of TN that the design of the policy compliance checker is more important than the design of the status information. When the policy compliance checkers of both Web Services are designed correctly, there will be no argument about the policy satisfaction status after each step, if both of them hold their own policy satisfaction status. Thus, the status information of whether or not a policy has been satisfied should not be designed within the communication protocol, but each Web Service should possess its own copy of the policy satisfaction status used by the policy compliance checker in its system. This produces the tenth set of requirements for protocol design and functionality design as shown below.

**Protocol Design Requirement 10:** *The protocol messages do not need to include the status information of the policy satisfaction of both Web Services.*

**Functionality Design Requirement 10:** *Each Web Service should possess a local copy of policy satisfaction status used by the policy compliance checkers in their system.*

In terms of the requirements of protocol messages in relation to the TN service (e.g. TrustBuilder), two messages are different from other messages (i.e. messages containing policies, credentials). The first message is the initial message sent from the Web Service Requester to explicitly inform the Web Service Provider which resource it intends to access. The second message is the last message sent from a Web Service Provider to a Web Service Requester to explicitly inform of the result of TN. Although the protocol in TrustBuilder is composed of the initial message and last message with other messages together, this design is not suitable for Web Services, since each of the two messages are used only once, whereas policy or credential messages can be used many times in TN. To take into account the simplicity of the schema definition of each interface within a WSDL file (discussion of a WSDL file is shown in section 2.4.4), it is proper to separate the design of the initial message and last message from policy messages, and credential messages. This produces the eleventh and twelfth requirements for protocol design and functionality design as shown below.

**Protocol Design Requirement 11:** *The syntax and semantics of the protocol messages should design a specific initial message sent from a Web Service Requester to inform a Web Service Provider about the detailed information of the requested resource such as the specific resource name or the owner's name of the resource.*

**Functionality Design Requirement 11:** *When two Web Services are attempting to use TN, and when a Web Service operates as a Web Service Requester, it should send the initial message to the Web Service Provider to include the detailed information about the requested resource. The Web Service Provider should understand this information and check whether the resource can be found through the local Web Service.*

**Protocol Design Requirement 12:** *The syntax and semantics of the protocol messages should include a last message to enable a Web Service to explicitly inform the counterpart of the result. In particular, if TN has succeeded, this message must be sent from a Web Service Provider to a Web Service Requester. If TN has failed, a message containing a fault reason should be sent from a Web Service (can be either a Web Service Requester or a Web Service Provider) to inform the counterpart why TN has failed.*

**Functionality Design Requirement 12:** *Whenever the Web Service Provider makes a final decision of whether TN has succeeded or not, it should always explicitly inform the Web Service Requester of the result.*

As the necessary requirements for a protocol have been presented above, the next section introduces a proposed improved TN protocol that can provide interoperability between authorisation systems within Web Services. The presentation begins with an overview of this protocol.

## 4.4 Overview of An Improved TN Protocol

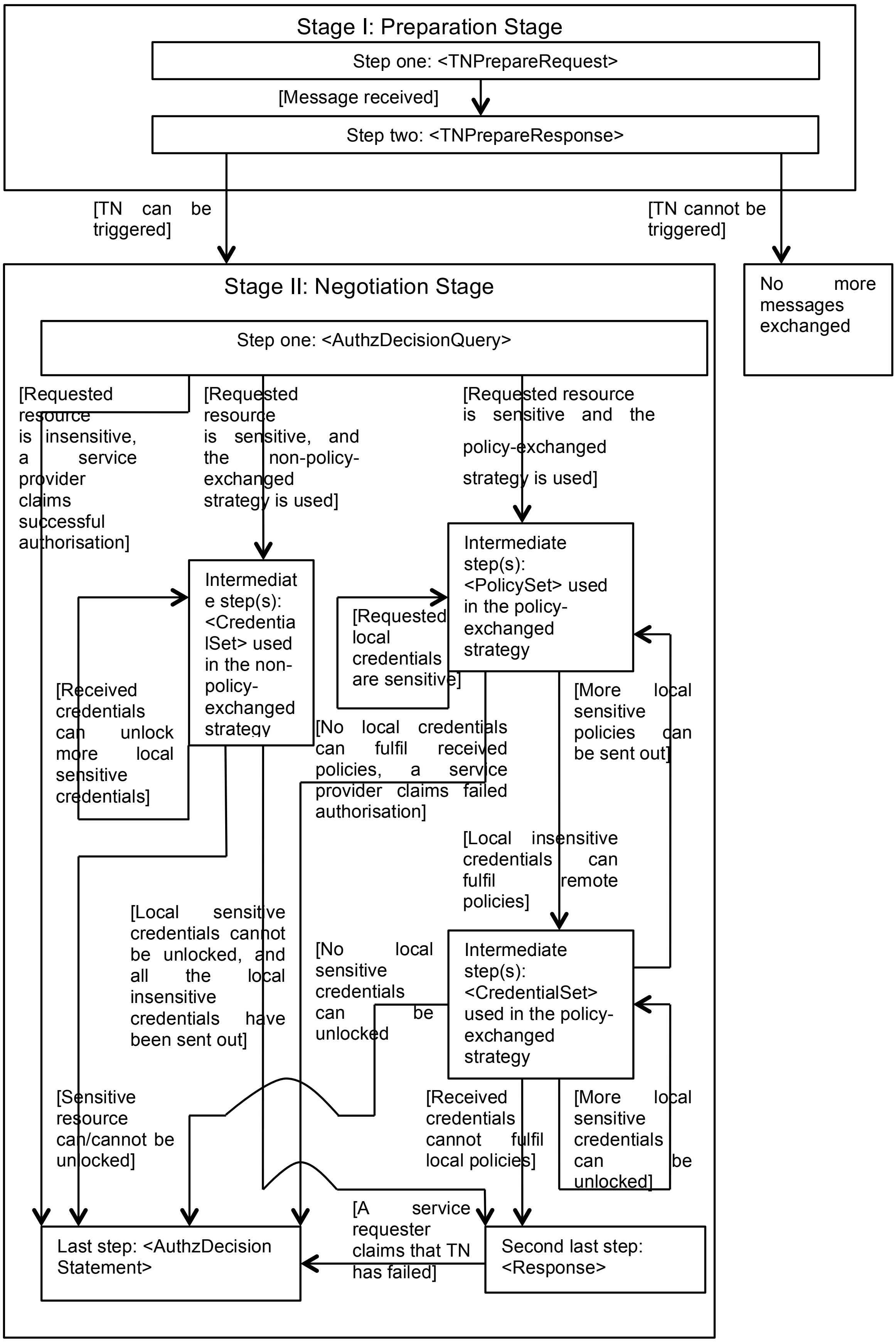
### 4.4.1 Scope and limitation of the protocol

As stated above, a protocol design may never be complete, since it will be modified to cater for new functionalities, as long as relevant new requirements arise. Thus, it is a natural result that existing protocols in Web Services are still in development, and will still be developing in the future (Nurse, 2010).

Similar to the existing protocols, the protocol designed in this chapter should not be treated as a final version. In other words, not all of the unique features of different TN-based authorisation systems are supported in this protocol. Instead, it can only support the most representative functionalities required by TN, along with the communication messages inherently required in such processes. The reason that this protocol can only support the necessary internal structures (i.e. functionalities) and protocol messages (i.e. communication messages) is rather straightforward: it may take many years for protocol designers to develop a protocol that can not only provide all of the relevant functionalities, but also be implemented in practice. In addition, many protocols are developed by large collaborative teams, but in this instance, the number of individuals contributing to the design and review of the protocol is relatively limited. In addition, attributes or elements for expressing the generation time or issuer etc. required in each protocol message are not considered in the protocol specification, since they can be adopted from existing standard protocols or specifications (see section 2.4).

### 4.4.2 An improved TN protocol

Following the conceptual multi-layered interoperability-solution design shown in table 4.1 and requirements developed in section 4.3, the proposed protocol borrows some messages defined within the SAML specification and the XACML specification. Two stages are designed within this protocol. Stage one is called the preparation stage, which is a novel stage, and stage two is called the negotiation stage. The negotiation stage is a stage allowing two Web Services to use TN for achieving authorisation. The general process of the protocol is shown in figure 4.2.



Stage I: Preparation Stage
Step one: <TNPrepareRequest>
[Message received]
Step two: <TNPrepareResponse>
[TN can be triggered]
[TN cannot be triggered]
No more messages exchanged
Stage II: Negotiation Stage
Step one: <AuthzDecisionQuery>
[Requested resource is insensitive, a service provider claims successful authorisation]
[Requested resource is sensitive, and the non-policy-exchanged strategy is used]
[Requested resource is sensitive and the policy-exchanged strategy is used]
Intermediate step(s): <CredentialSet> used in the non-policy-exchanged strategy
[Received credentials can unlock more local sensitive credentials]
[Requested local credentials are sensitive]
Intermediate step(s): <PolicySet> used in the policy-exchanged strategy
[No local credentials can fulfil received policies, a service provider claims failed authorisation]
[More local sensitive policies can be sent out]
[Local insensitive credentials can fulfil remote policies]
[Local sensitive credentials cannot be unlocked, and all the local insensitive credentials have been sent out]
[No local sensitive credentials can be unlocked]
Intermediate step(s): <CredentialSet> used in the policy-exchanged strategy
[Sensitive resource can/cannot be unlocked]
[Received credentials cannot fulfil local policies]
[More local sensitive credentials can be unlocked]
[A service requester claims that TN has failed]
Last step: <AuthzDecision Statement>
Second last step: <Response>

Figure 4.2. Overview of an improved TN protocol

This preparation stage is designed according to requirements 1 to 6. As stated in (Zartman and Berman, 1982) a preparation stage can help to achieve a higher probability of successful negotiation. With the addition of this preparation stage within the protocol, the real commencement of TN in this protocol belongs to the second stage, namely, the negotiation stage.

The preparation stage allows two Web Services that are not maliciously designed (e.g. they are not designed to attack other Web Services) to consult with each other to reach an agreement on the use of a common strategy (interoperability at layer 6) and a common language combination (interoperability at layer 5). The agreement on a common strategy and a common language combination can help the TN approach avoid failure caused by the strategic interoperability issue (layer 6) or the capability interoperability issue (layer 5). The relevant protocol messages and internal structures achieving the communication at this step can provide syntactic (layer 2), semantic (layer 3) and functionality (layer 4) interoperability.

The negotiation stage is designed according to requirements 7 to 12. It provides syntactic (layer 2), semantic (layer 3) and functionality (layer 4) in relation to TN. In other words, it can enable two authorisation systems within Web Services to conduct TN by using the credentials and policies written in the agreed language combination and the agreed strategy. The processes within the negotiation stage by using the two types of strategies are different. With the use of the eager strategy (a typical representative of non-policy-exchanged strategies), only credentials will be exchanged between two Web Services, whereas message exchange will include both policies and credentials, when the parsimonious strategy (a typical representative of policy-exchanged strategies) is used.

## 4.5 Preparation Stage

**Service specification:** The design of the preparation stage is based on **Protocol Design Requirements 1-6 and Functionality Design Requirements 1-6.** The main purpose of the preparation stage is to allow two unknown Web Services to communicate with each other about consulting the capability interoperability (layer 5) and the strategic interoperability (layer 6). More specifically, it allows the two Web Services to consult with each other to identify whether there exists at least one common strategy and one common language combination (for expressing policies and credentials). Within this protocol, inbuilt language interpretation functionality along with a policy compliance checker based on an object-oriented design for comparing local policies(credentials) against received remote credentials(policies) are treated as the capability owned by each entity. If there is more than one common strategy and one common language combination, a Web Service Provider will then determine to opt for one ideal language combination for expressing policies and credentials before TN commences. In order for the Web Service Provider to decide on whether a common strategy or a common language combination can be found, the keyword-based approach and the semantic-based approach (e.g. by using approaches such as Resource Description Framework (Klyne and Carroll, 2004), Web Ontology Language (Hitzler et al, 2012) and SPARQL (Prud and Seaborne, 2008) etc.) used in Web Services discovery (Garofalakis et al., 2006) can be applied to this protocol. Two steps are included in this stage. They are designed for a Web Service Provider to determine a possible common strategy and a possible common language combination.

To aid clarity, the description of the preparation stage is divided into separate steps. At each step, the internal structure of the step is described first; this outlines which messages can be sent or received at which time points. An UML activity diagram is then presented to exemplify the relevant processes of the internal structure. Following the clarification of the internal structure, the relevant protocol message is simply described. Detailed syntax and semantics of the protocol message is presented in Appendix A.

### 4.5.1 Step one – Sends out a <TNPrepareRequest> message

**Internal Structure:** Whenever a user of a Web Service Requester decides to access a resource located on a Web Service Provider, the message communication will be triggered. The Web Service Requester then initiates a request by sending a <TNPrepareRequest> message (described later) to inform a Web Service Provider that it is intending to access a targeted resource or service (the process of the internal structure is shown in figure 4.3).

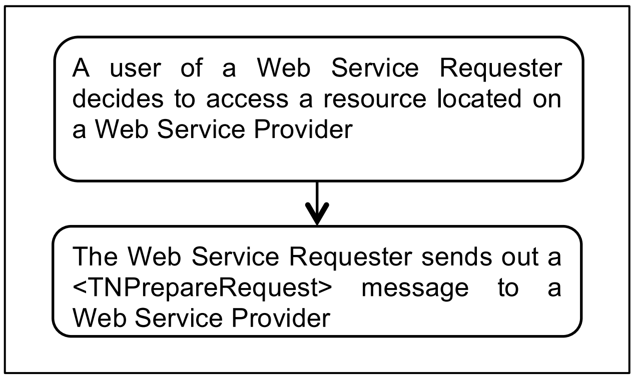

A user of a Web Service Requester decides to access a resource located on a Web Service Provider
The Web Service Requester sends out a <TNPrepareRequest> message to a Web Service Provider

Figure 4.3. A Web Service Requester sends out a <TNPrepareRequest> message to a Web Service Provider

**Protocol message description:** a <TNPrepareRequest> message includes a list of all of the supported access control methods enabling a Web Service Provider to match one of them to their accepted method list.

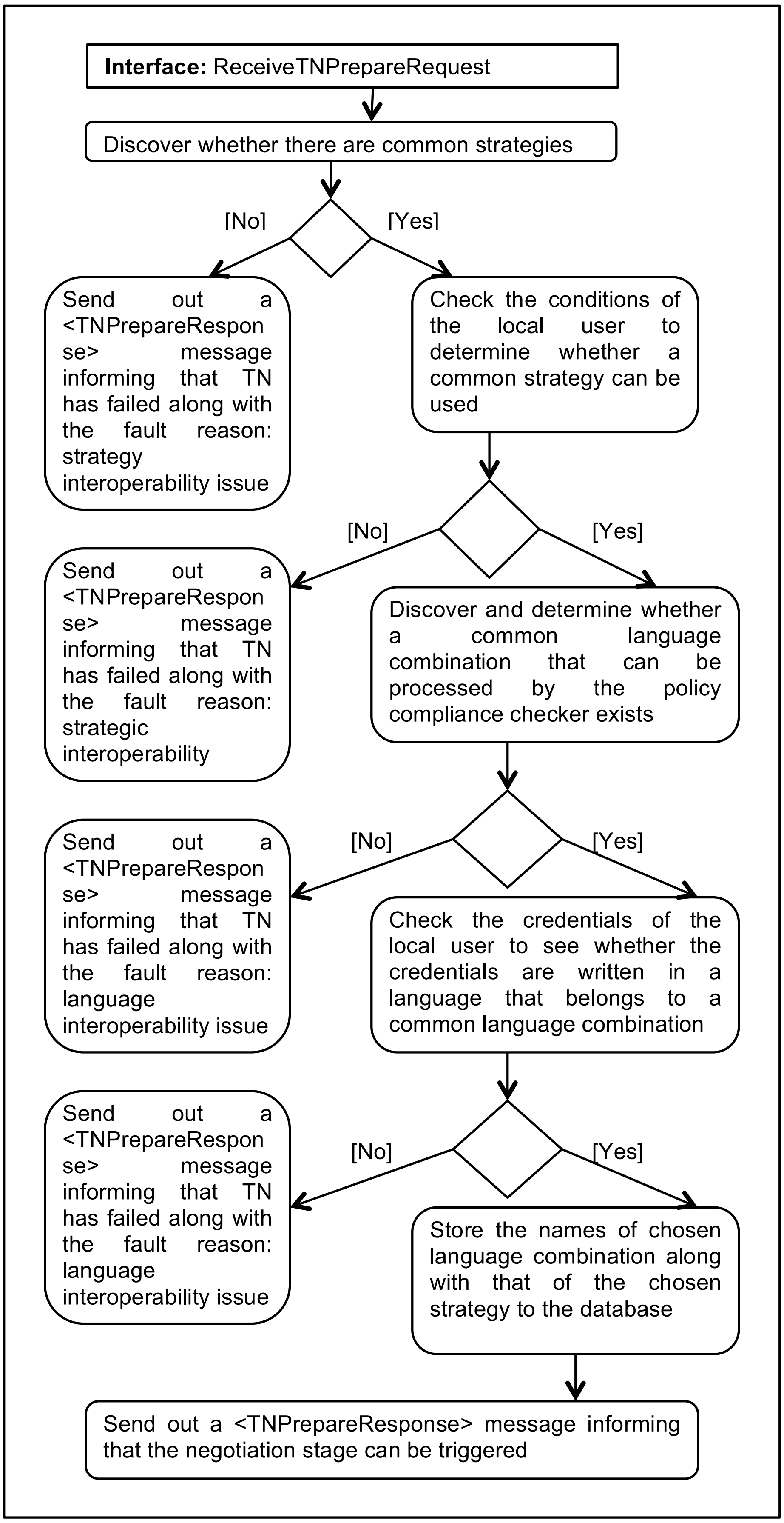
The detailed syntax and semantics of a <TNPrepareRequest> message is presented in section A.1 in Appendix A.

### 4.5.2 Step Two – Receives an incoming <TNPrepareRequest> and sends out an outgoing <TNPrepareResponse> message

**Internal Structure:** When the Web Service Provider receives a <TNPrepareRequest> message from the Web Service Requester at step one, it will determine the information in each list to discover whether there are common strategies. If there are common strategies, the Web Service Provider should check the conditions of the local user, which possesses the requested resource to determine whether the common strategies can be used according to the local user's condition. For instance, the eager strategy cannot be used, when all of the credentials owned by the local user are treated as sensitive.

If none of the common strategies is suitable to the conditions of the local user, the Web Service Provider will respond with a <TNPrepareResponse> message (described later) providing information that TN cannot be used along with the fault reason that the strategy interoperability issue has been discovered. If there is more than one common strategy that is suitable to the conditions of the local user, the Web Service Provider should choose an ideal one according to a locally determined preference order. After ensuring there is no strategy interoperability issue, the Web Service

Provider will discover whether there is any common language combination (expressing credentials and policies) within the list. If no common language combination can be found, the Web Service will respond with a <TNPrepareResponse> message providing information that TN cannot be used along with the fault reason that a language interoperability issue has been discovered. If there is more than one common language combination, the Web Service Provider will check whether or not the credentials owned by the user or the organisation are written in the language belonging to the common language combination to finally decide on whether a language combination can be used. Credential language checking is required, since credential languages may be out of the control of the Web Service Provider, whereas the policy languages used in the Web Service are normally under the control of the Web Service Provider, so there is no need to check the policy language. If a common language combination can be discovered, but the credentials owned by the user are not written in a credential language that can be understood, the Web Service Provider will send out a <TNPrepareResponse> message informing that TN cannot be used along with the fault reason that a language interoperability issue has been discovered. Finally, if both a common strategy and a common language combination can be discovered, the Web Service Provider will respond with a <TNPrepareResponse> message providing information that the negotiation stage can be triggered. In addition, the common strategy name and common language combination name along with the method name should be stored in a database for use in the negotiation stage (The process of the internal structure is shown in figure 4.4).



**Interface:** ReceiveTNPrepareRequest
Discover whether there are common strategies
[No]
[Yes]
Send out a <TNPrepareResponse> message informing that TN has failed along with the fault reason: strategy interoperability issue
Check the conditions of the local user to determine whether a common strategy can be used
[No]
[Yes]
Send out a <TNPrepareResponse> message informing that TN has failed along with the fault reason: strategic interoperability
Discover and determine whether a common language combination that can be processed by the policy compliance checker exists
[No]
[Yes]
Send out a <TNPrepareResponse> message informing that TN has failed along with the fault reason: language interoperability issue
Check the credentials of the local user to see whether the credentials are written in a language that belongs to a common language combination
[No]
[Yes]
Send out a <TNPrepareResponse> message informing that TN has failed along with the fault reason: language interoperability issue
Store the names of chosen language combination along with that of the chosen strategy to the database
Send out a <TNPrepareResponse> message informing that the negotiation stage can be triggered

Figure 4.4. A Web Service Requester processes a <TNPrepareRequest> message

**Protocol message description:** A <TNPrepareResponse> message contains information about whether or not the negotiation stage can be triggered. If the negotiation stage can be triggered, information about the common strategy and language combination supported by the Web Service Provider is contained; otherwise, the fault information explaining the reason is contained in this message informing that the negotiation stage cannot be triggered.

The detailed syntax and semantics of a <TNPrepareResponse> message is presented in section A.2 in Appendix A.

## 4.6 Negotiation Stage

**Service specification:** The design of the preparation stage is based on **Protocol Design Requirements 7-12 and Functionality Design Requirements 7-12.**

If two Web Services have agreed to use a common strategy and a common language combination (for expressing policies and credentials) by using TN to establish a trust relationship in the preparation stage, the negotiation stage is then triggered to perform real TN. In theory, the protocol design aims to support the use of different strategies. However, some policy-exchanged strategies such as PRUNES and DFANS are not available within this protocol, as they are not as superior as claimed by their designers (see assessments in section 3.3.1). By contrast, as the eager strategy and parsimonious strategy have been identified as the most representative strategies of the non-policy-exchanged strategies and policy-exchanged strategies respectively through the assessment of their feasibility in Web Services as stated in section 3.3.1, they are incorporated in this protocol at a higher priority. In addition, the architecture design within TrustBuilder2 is borrowed for serving this protocol, since the components designed within TrustBuilder2 can support different strategies and different policy compliance checkers for processing different language combinations (see section 3.3.4). This design can help two Web Services raise the possibility of interoperable communication for TN.

### 4.6.1 Step one – Receives an incoming <TNPrepareResponse> message and sends out an outgoing <AuthzDecisionQuery> message

**Internal Structure:** When the Web Service Requester receives a <TNPrepareResponse> message, it will check whether the negotiation stage can be triggered. If the negotiation can be triggered, the relevant information such as a common strategy and a common language combination will be stored in a database. It will then send an <AuthzDecisionQuery> message (described later) to the Web Service Provider; otherwise, the commencement of the negotiation stage cannot be triggered due to a specific interoperability issue (the logical process of the internal structure is shown in figure 4.5).

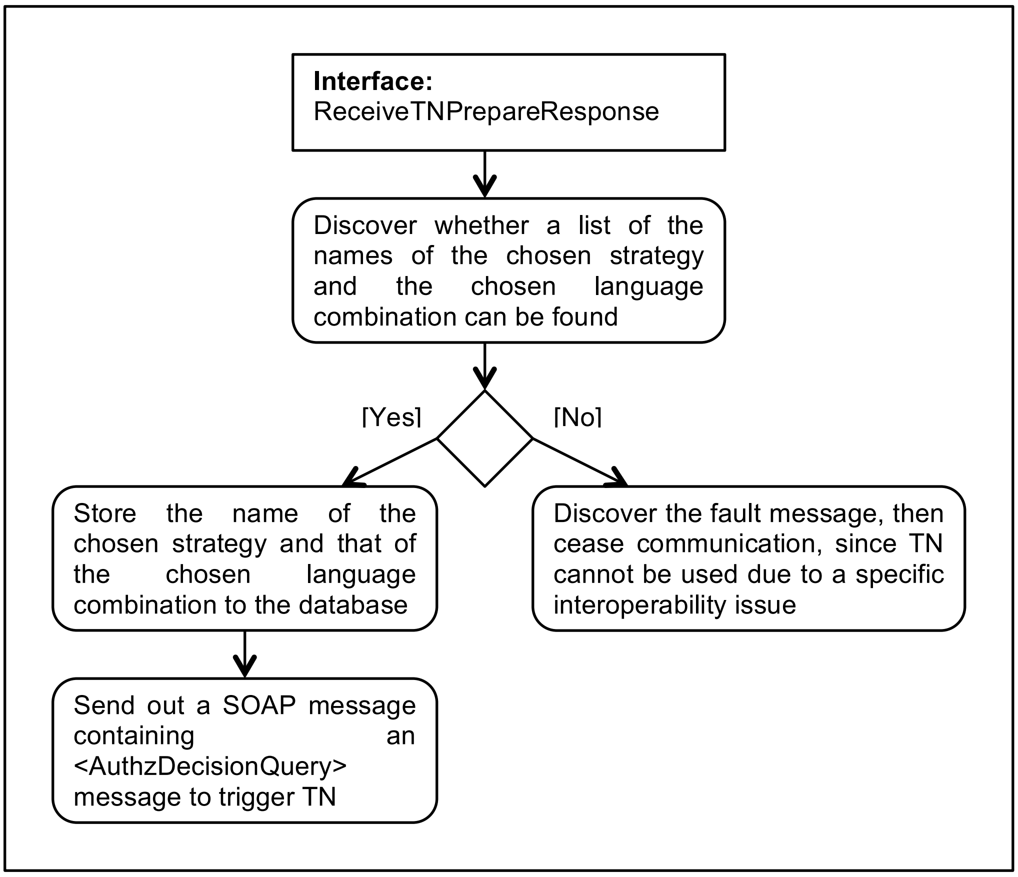

Interface:
ReceiveTNPrepareResponse
Discover whether a list of the names of the chosen strategy and the chosen language combination can be found
[Yes]
[No]
Store the name of the chosen strategy and that of the chosen language combination to the database
Discover the fault message, then cease communication, since TN cannot be used due to a specific interoperability issue
Send out a SOAP message containing an <AuthzDecisionQuery> message to trigger TN

Figure 4.5. A Web Service Requester processes a <TNPrepareResponse> message

The existing SAML specification defines the syntax and semantics of messages for authorisation between two Web Services. As some of their syntax and semantics are suitable to be used to fulfil the protocol requirements for TN, they are applied to this protocol to enable this protocol to be compatible with the existing Web Service specifications.

**Protocol message description:** an <AuthzDecisionQuery> message contains information about the user of the Web Service Requester, the Web Service Requester itself and the requested information located on the Web Service Provider.

The detailed syntax and semantics of an <AuthzDecisionQuery> message is presented in section A.3 in Appendix A.

### 4.6.2 Step two – Receives an incoming <AuthzDecisionQuery> message and decides to sends a relevant outgoing message

**Internal Structure:** If a Web Service has received an <AuthzDecisionQuery> message, it must be a Web Service Provider. It will obtain the values of the relevant attributes and store them in the database. In order to know what message should be sent out, the Web Service Provider will check the database to obtain the chosen strategy decided in the preparation stage. Information on the chosen strategy will be sent to a component called the "Negotiation Strategy Repository Component" (as a plugin component in TrustBuilder2, which can contain different implemented strategies) to trigger the real strategy. As mentioned earlier, since this protocol can only support the use of the eager strategy and parsimonious strategy, the discussed scenarios are only relevant to the use of the two strategies. If the Web Service Provider uses the eager strategy, it will send out a <CredentialSet> message (described in section 4.6.4) containing local insensitive credentials. There should exist local insensitive credentials, since the existence of the available insensitive credentials has been ensured at step 2 in the preparation stage (see section 4.5.2). If the Web Service Provider uses the parsimonious strategy, it should check whether there are relevant policies protecting the resources. If there are, then the policies are placed into a <PolicySet> message (described in section 4.6.3) to be sent out. If there is no relevant policy protecting the resources, a particular <AuthzDecisionStatement> message (described in section 4.6.6) will be sent out. Although this case may rarely occur in TN, it is still taken into consideration for ensuring the completeness of the protocol (The logical process of the internal structure is shown in figure 4.6).

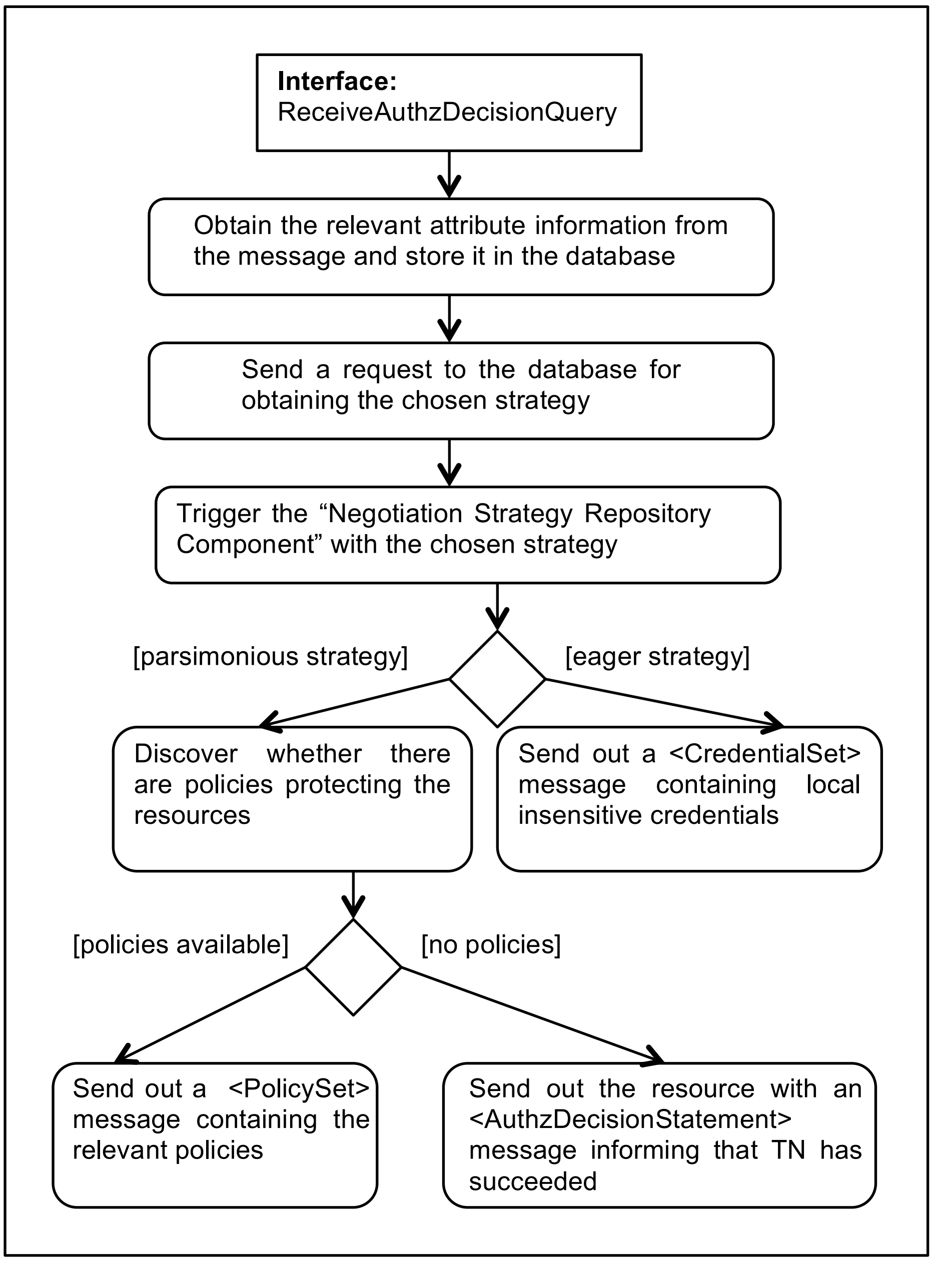

Interface:
ReceiveAuthzDecisionQuery
Obtain the relevant attribute information from the message and store it in the database
Send a request to the database for obtaining the chosen strategy
Trigger the "Negotiation Strategy Repository Component" with the chosen strategy
[parsimonious strategy]
[eager strategy]
Discover whether there are policies protecting the resources
Send out a <CredentialSet> message containing local insensitive credentials
[policies available]
[no policies]
Send out a <PolicySet> message containing the relevant policies
Send out the resource with an <AuthzDecisionStatement> message informing that TN has succeeded

Figure 4.6. A Web Service Provider processes an <AuthzDecisionQuery> message

### 4.6.3 Possible intermediate steps – Receives an incoming <PolicySet> message and sends out a relevant outgoing message

**Internal Structure:** Whenever a Web Service (can be either a Web Service Requester or a Web Service Provider) receives a <PolicySet> message (the syntax and semantics is described later), a Web Service needs to read the attribute information within the <PolicySet> message and store this information in the database. This functionality ensures that the Web Service is able to send an appropriate outgoing message in response to the correct counterpart. In fact, this functionality is even required within the existing Web Services, which receive the request and send the relevant response information. Following this functionality, the Web Service will read the name of the chosen strategy and language combination from the database to trigger the chosen strategy within the “Negotiation Strategy Repository Component” and the chosen policy compliance checker within a component called “Policy Compliance Checker Component”. Similar to the “Negotiation Strategy Repository Component”, a “Policy Compliance Checker Component” can contain an implementation of different policy compliance checkers, where each of them can process the specific language combination (the logical process of the aforementioned internal structure is shown in figure 4.7).

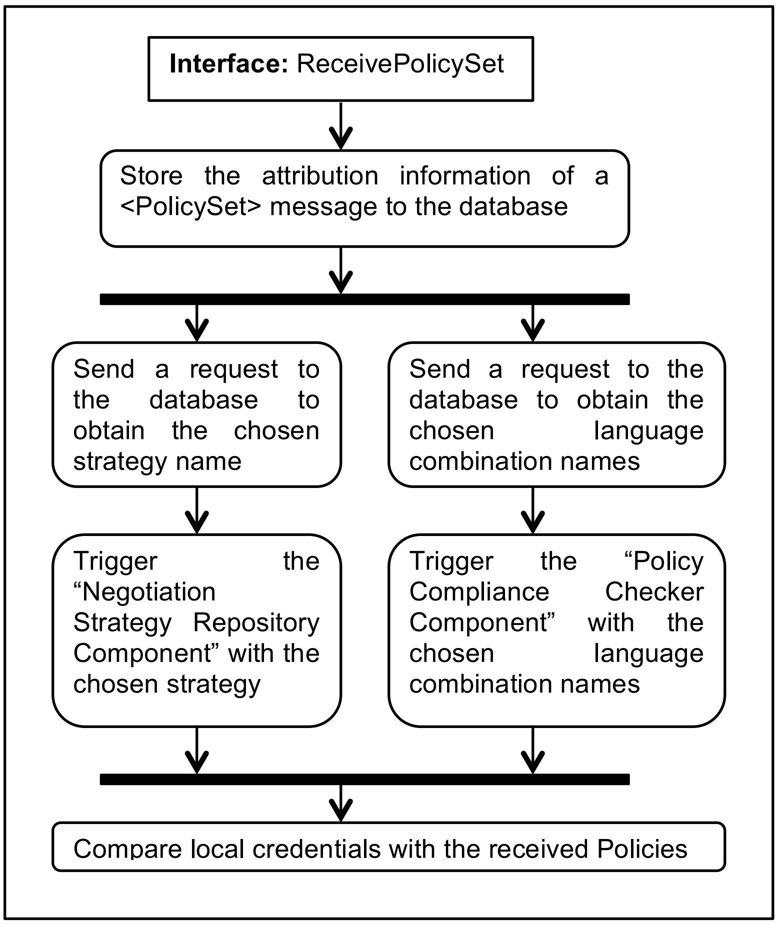

**Interface:** ReceivePolicySet
Store the attribution information of a <PolicySet> message to the database
Send a request to the database to obtain the chosen strategy name
Send a request to the database to obtain the chosen language combination names
Trigger the “Negotiation Strategy Repository Component” with the chosen strategy
Trigger the “Policy Compliance Checker Component” with the chosen language combination names
Compare local credentials with the received Policies

Figure 4.7. A Web Service processes a <PolicySet> message and decides to compare local credentials with the received policies

With the cooperation of the chosen strategy and chosen policy compliance checker, a Web Service is able to compare local credentials with the received policies. As the TN approach is actually an extension of ABAC, the purpose of the policy compliance checker is to identify whether the attributes within local credentials can fulfil the policies. Credential types or names mentioned in the policies are only used as examples. In other words, the policy compliance checker should not be designed to search a credential repository relevant to a user or organisation (can be either a requester or a provider) for whether there are specific local credentials along with the attribute information mentioned in the policies. To achieve this purpose, an ontology based approach as proposed in Squicciarini et al. (2006), should be integrated in each policy compliance checker. This method can ensure that the received policies are indeed fulfilled by the correct attribute information of credentials rather than a specific credential type or name.

Once the policy compliance checker has determined whether the local credentials can fulfil all of the rules of the received policies, it will decide what message should be sent out. The result can be divided into four cases. The logical process of the internal structure is shown in figure 4.8

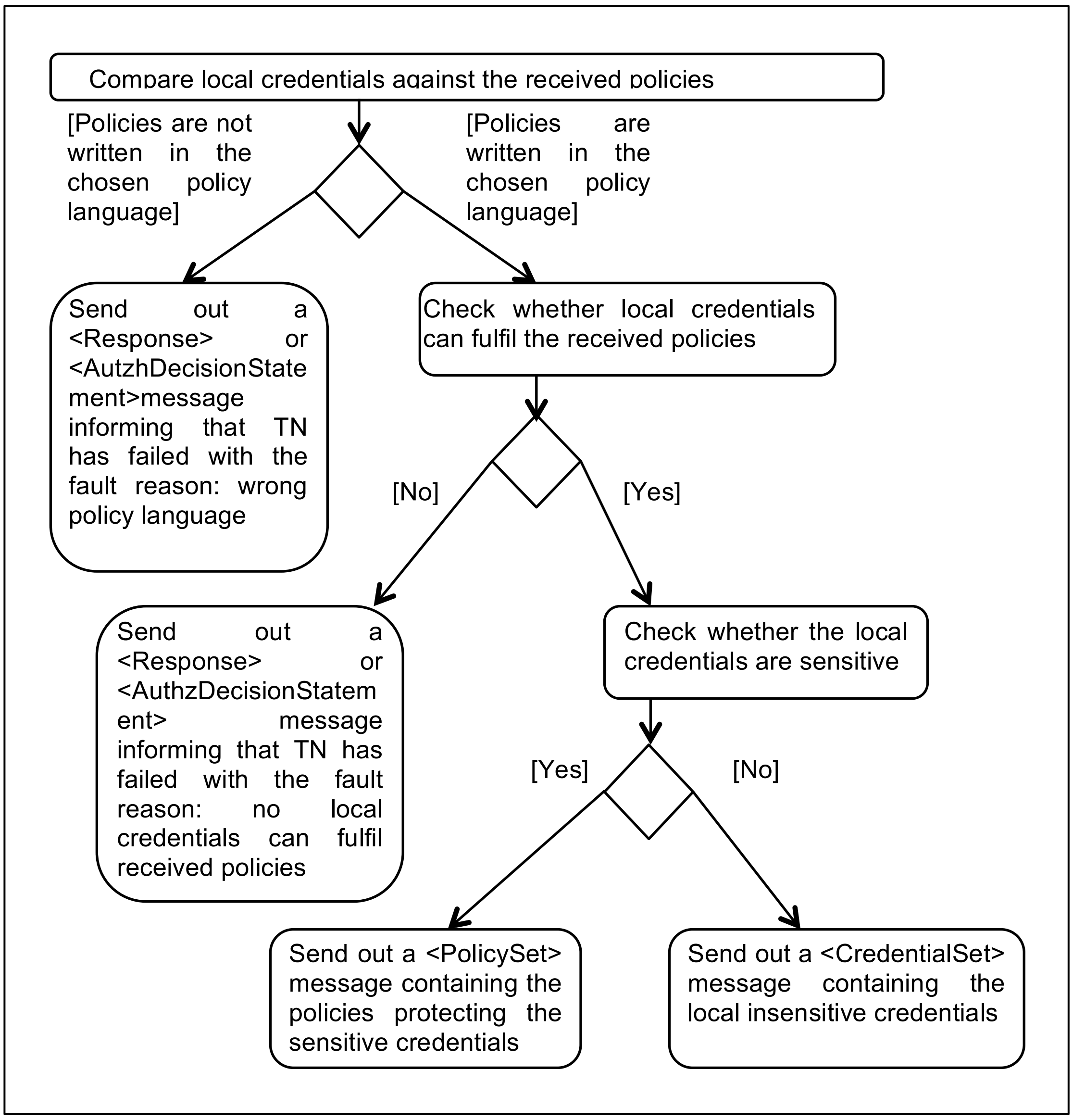

Compare local credentials against the received policies
[Policies are not written in the chosen policy language]
[Policies are written in the chosen policy language]
Send out a <Response> or <AutzhDecisionStatement>message informing that TN has failed with the fault reason: wrong policy language
Check whether local credentials can fulfil the received policies
[No]
[Yes]
Send out a <Response> or <AuthzDecisionStatement> message informing that TN has failed with the fault reason: no local credentials can fulfil received policies
Check whether the local credentials are sensitive
[Yes]
[No]
Send out a <PolicySet> message containing the policies protecting the sensitive credentials
Send out a <CredentialSet> message containing the local insensitive credentials

Figure 4.8. A process illustrating how the policy compliance checker makes a decision

Case 1: The received policies are not written in the policy language of the chosen language combination, so comparison of them against local credentials is impossible. This may occur, when the counterpart is a malicious Web Service. As a result, the Web Service will send out a <Response> message (as a Web Service Requester, syntax and semantics are described in section 4.6.5) or an <AuthzDecisionStatement> message (as a Web Service Provider) informing the counterpart that TN has failed along with the reason that an unknown policy language has been discovered.

Case 2: If the received policies are written in the recognised policy language, after comparison with the local credentials, and it is determined that no local credentials can fulfil all of the received policies, then a <Response> message (as a Web Service

Requester) or an <AuthzDecisionStatement> message (as a Web Service Provider) will be sent out with the fault reason that no local credentials can fulfil the received policies. This message informs the counterpart that TN has failed.

Case 3: In cases that the received policies are written in the recognised policy language, after comparing them against local sensitive credentials, if local sensitive credentials can fulfil them, and local sensitive credentials have not been unlocked (there are local policies protecting their disclosure), the Web Service should send out these local policies in a <PolicySet> message first.

Case 4: In cases that the received policies are written in the recognised policy language, after comparing them against local insensitive/sensitive credentials, if local insensitive credentials or sensitive credentials (the relevant policies have been fulfilled) can fulfil them, the Web Service should send out these local insensitive/sensitive credentials in a <CredentialSet> message (discussed in section 4.6.4).

The existing XACML specification has provided a well-designed language, which can specify one or more policies for the access control between Web Services. As they can meet the protocol design requirements mentioned above, their syntax and semantics are applied to this protocol.

**Protocol message Description:** a <PolicySet> message can contain one or more policies. Unlike the original syntax and semantics defined in the XACML specification that can only contain policies written in XACML, the modified semantics of this message, in this protocol, is flexible enough to contain policies written in any other existing policy language(s). If the existing policy language(s) such as the Ack policy language (Winsborough and Li, 2002a) has defined its own syntax and semantics to support the use of multiple policies, they can be contained directly in this message. If policy languages such as the X-TNL policy language (Bertino, Ferrari and Squicciarini, 2003b) do not provide such functionality in the message, this protocol can provide optional syntax and semantics to support multiple policy provision. If needed, the policies written in other policy languages contained within this message can be encoded in a BASE64 format, which is one of the popular encoding schemes used to store or transfer data within SOAP messages in the context of Web Services (Josefsson, 2006).

The detailed syntax and semantics of a <PolicySet> message is presented in section A.4 in Appendix A.

### 4.6.4 Possible intermediate steps – Receives an incoming <CredentialSet> message and sends out a relevant outgoing message

**Internal Structure:** When a Web Service (can be either a Web Service Requester or a Web Service Provider) receives a <CredentialSet> message (described later), the Web Service must verify the authenticity of the credential chains, before it is ready to compare them with the local policies. In order to verify the credentials, the Web Service will search the database to know the credential language within the chosen language combination. If the received credential cannot pass the verification (e.g. the credentials are not written in the credential language of the chosen language combination or other issues etc.), a <Response> message (as a Web Service Requester) or an <AuthzDecisionStatement> message (as a Web Service Provider) will be sent out informing that TN has failed along with the fault reason that the wrong credentials were received. If the received credential can pass the verification, a query sent from the database will trigger the chosen strategy and the chosen policy compliance checker to compare the received credentials with local policies. The process of the partial internal structure is shown in figure 4.9 below.

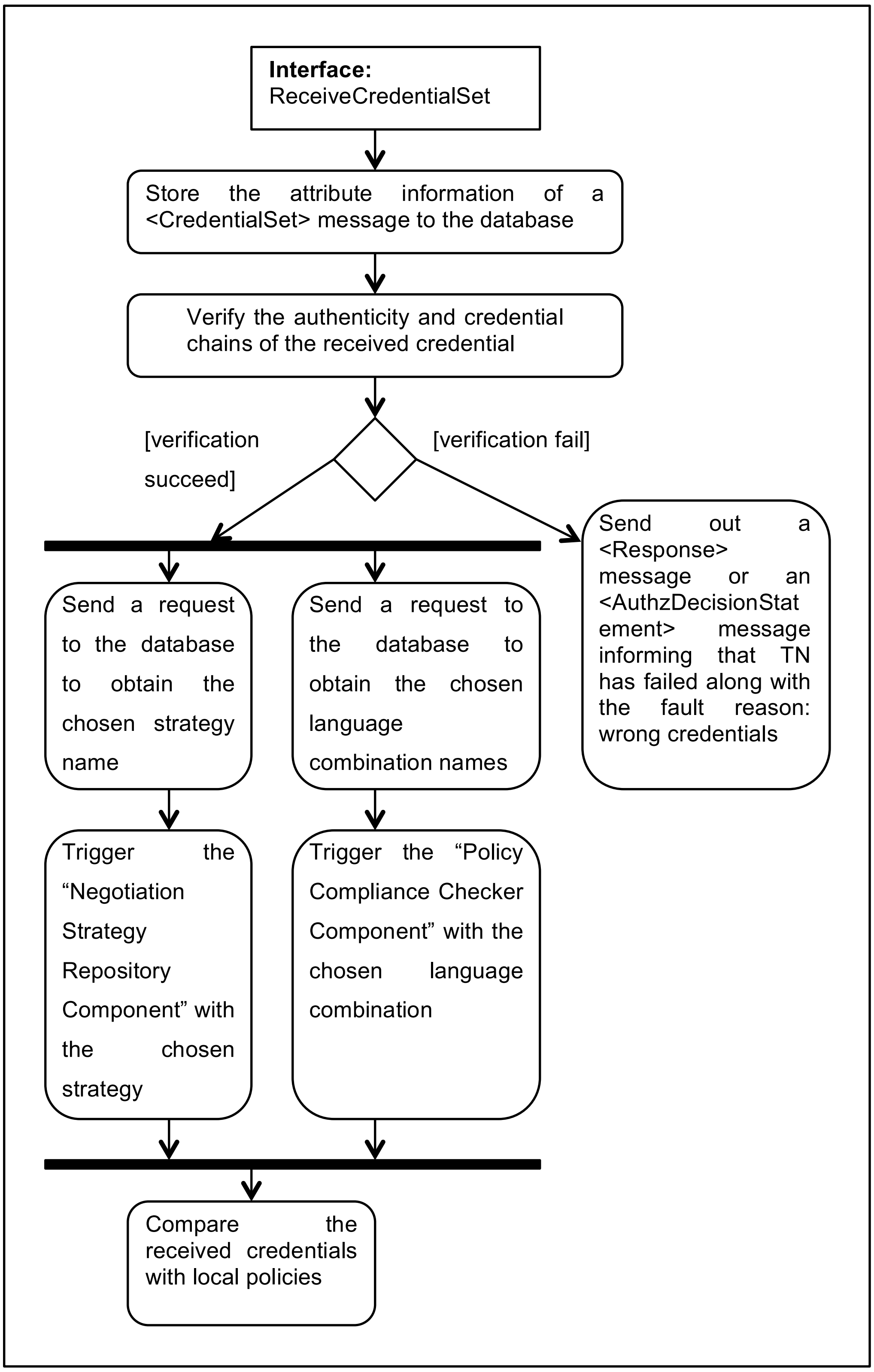

Interface: ReceiveCredentialSet
Store the attribute information of a <CredentialSet> message to the database
Verify the authenticity and credential chains of the received credential
[verification succeed]
[verification fail]
Send out a <Response> message or an <AuthzDecisionStatement> message informing that TN has failed along with the fault reason: wrong credentials
Send a request to the database to obtain the chosen strategy name
Send a request to the database to obtain the chosen language combination names
Trigger the "Negotiation Strategy Repository Component" with the chosen strategy
Trigger the "Policy Compliance Checker Component" with the chosen language combination
Compare the received credentials with local policies

Figure 4.9. A process of verifying the authenticity and credential chains of the received credentials to decide what to do

As the process of comparing the received credentials and local policies through the use of the eager strategy is different from that of using the parsimonious strategy, the following describes the process by using the eager strategy first. If the Web Service is the Web Service Provider, it will check whether or not the received credentials can fulfil the policies that protect the requested resource. If they can, the Web Service Provider will send the resource and an <AuthzDecisionStatement> message informing that TN has succeeded. If the received credentials cannot fulfil the policies, it will check whether the received credentials can unlock any local sensitive credentials that have not been sent by comparing them with local policies (from this step, the process is the same for the Web Service Requester). If there are no further local sensitive credentials that can be unlocked, it will check whether there are any local insensitive credentials that have not been disclosed. If there are, these insensitive credentials need to be sent out in a <CredentialSet> message; otherwise, the Web Service will send a <Response> message (as a Web Service Requester) or an <AuthzDecisionStatement> message (as a Web Service Provider) informing the counterpart that TN has failed along with the fault reason that no local credentials can be unlocked. If there are other local sensitive credentials that can be unlocked, the new set of credentials will be sent out in a <CredentialSet> message. The logical process of the internal structure in the case of using the eager strategy is shown in figure 4.10 below.

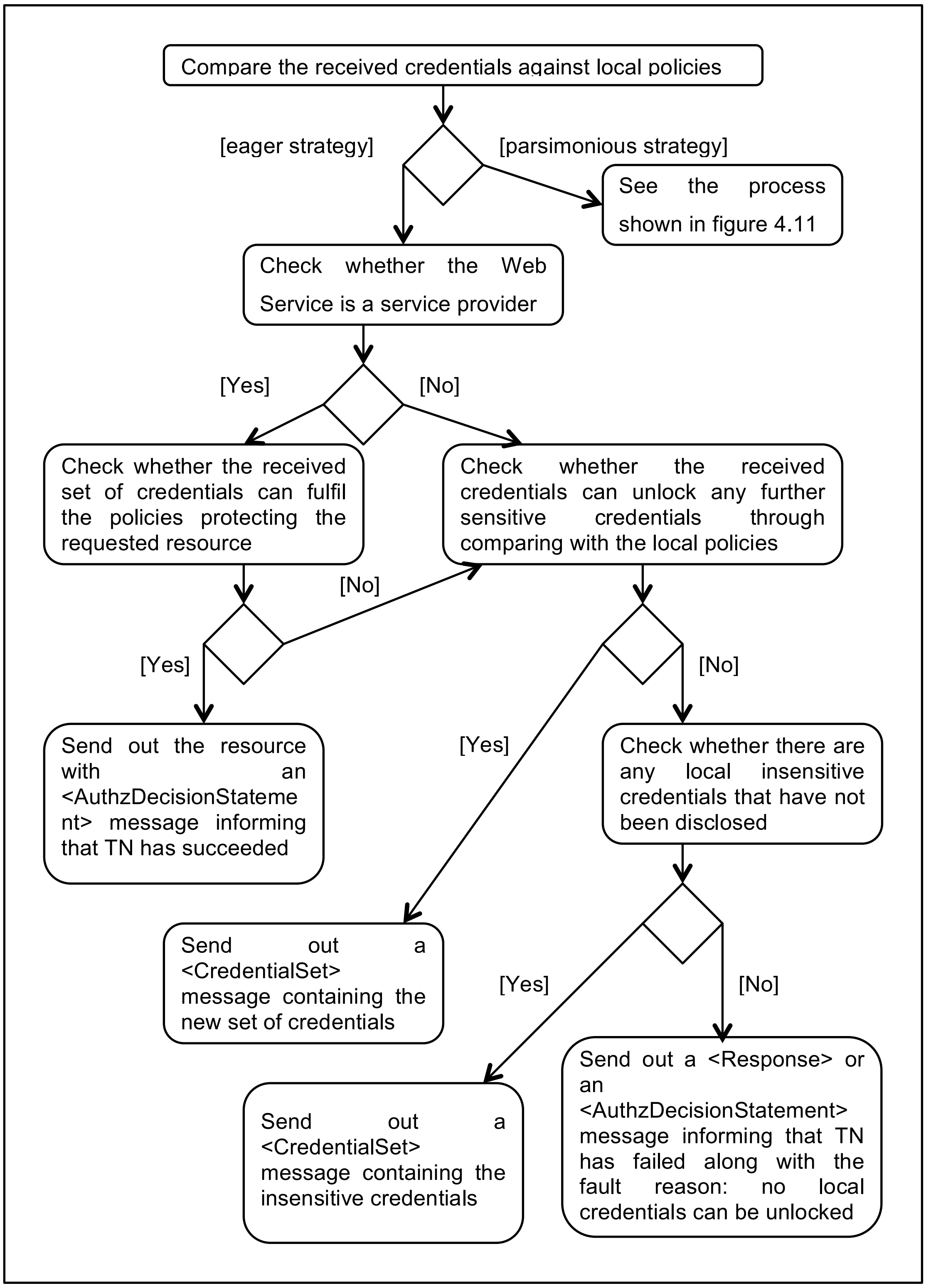

Compare the received credentials against local policies
[eager strategy]
[parsimonious strategy]
See the process shown in figure 4.11
Check whether the Web Service is a service provider
[Yes]
[No]
Check whether the received set of credentials can fulfil the policies protecting the requested resource
Check whether the received credentials can unlock any further sensitive credentials through comparing with the local policies
[No]
[Yes]
[No]
Send out the resource with an <AuthzDecisionStatement> message informing that TN has succeeded
[Yes]
Check whether there are any local insensitive credentials that have not been disclosed
Send out a <CredentialSet> message containing the new set of credentials
[Yes]
[No]
Send out a <CredentialSet> message containing the insensitive credentials
Send out a <Response> or an <AuthzDecisionStatement> message informing that TN has failed along with the fault reason: no local credentials can be unlocked

Figure 4.10. A process of comparing the received credentials against local policies by using the eager strategy

The following describes the process of comparing the received credentials and local policies, when the parsimonious strategy is used. If the received credentials cannot fulfil the specific local policies, the Web Service will send out a <Response> message (as a Web Service Requester) or an <AuthzDecisionStatement> message (as a Web Service Provider) informing that TN has failed along with the reason that the wrong credentials have been received. If the credentials can fulfil the specific policies, it will check whether there are any further local policies that need to be sent out. If there are, these policies will be sent out in a <PolicySet> message. This circumstance may occur, when the received credentials have fulfilled the policies protecting other sensitive policies. As the sensitive policies have been unlocked, they can be sent out to the counterpart.

If the Web Service has discovered that there are no further policies that need to be sent out, the next action to be performed by the two Web Services is different. If the Web Service is the Web Service Provider, it will check whether there are any local sensitive credentials that have been unlocked. If there are, these unlocked sensitive credentials will be sent out in a <CredentialSet> message. If there are not, it will send the resource and an <AuthzDecisionStatement> message informing that TN has succeeded, since all of the policies relevant to the requested resource have been fulfilled. If the Web Service is a Web Service Requester, it will send out unlocked sensitive credentials in a <CredentialSet> message to the counterpart. The relevant process of the internal process by using the parsimonious strategy is shown in figure 4.11 below.

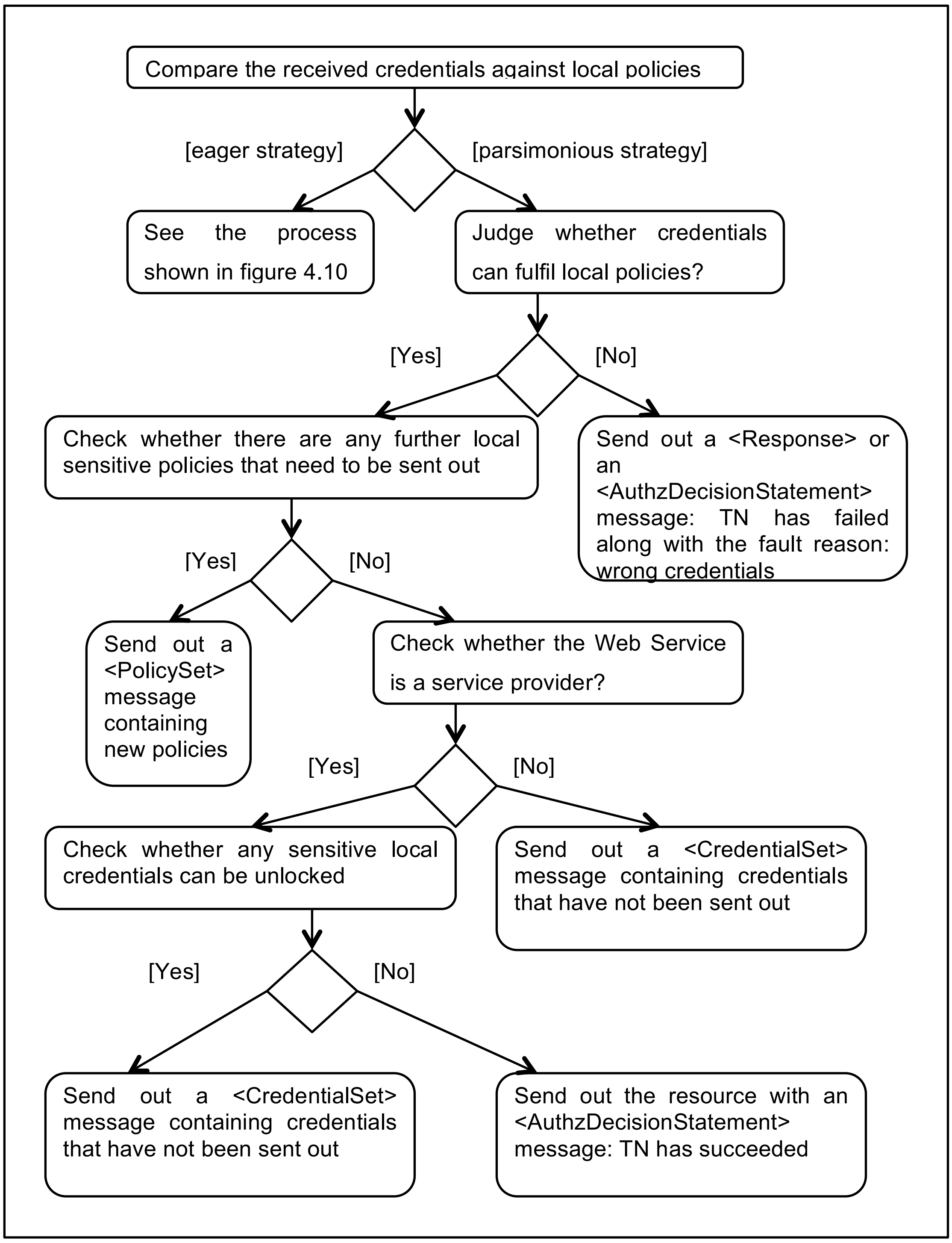

Compare the received credentials against local policies
[eager strategy]
[parsimonious strategy]
See the process shown in figure 4.10
Judge whether credentials can fulfil local policies?
[Yes]
[No]
Check whether there are any further local sensitive policies that need to be sent out
Send out a <Response> or an <AuthzDecisionStatement> message: TN has failed along with the fault reason: wrong credentials
[Yes]
[No]
Send out a <PolicySet> message containing new policies
Check whether the Web Service is a service provider?
[Yes]
[No]
Check whether any sensitive local credentials can be unlocked
Send out a <CredentialSet> message containing credentials that have not been sent out
[Yes]
[No]
Send out a <CredentialSet> message containing credentials that have not been sent out
Send out the resource with an <AuthzDecisionStatement> message: TN has succeeded

Figure 4.11. A process of comparing the received credentials against local policies by using the parsimonious strategy

**Protocol message description:** a <CredentialSet> message contains one or more credentials disclosed by a Web Service. The structure of this message is similar to that in a <PolicySet> message, which is designed as a container to include credentials written in different credential languages.

The detailed syntax and semantics of a <CredentialSet> message is presented in section A.5 in Appendix A.

### 4.6.5 Possible second last step – Receives a <Response> message from a Web Service Requester and sends out an <AuthzDecisionStatement> message

**Internal Structure:** If a Web Service Provider receives this message, it means that the TN has failed due to some reasons. It then just checks the fault reason and sends out an <AuthzDecisionStatement> message (see section 4.6.6). The logical process of the internal structure is shown in figure 4.12.

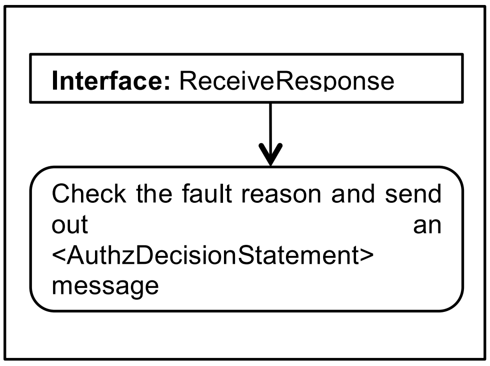

**Interface:** ReceiveResponse
Check the fault reason and send out an <AuthzDecisionStatement> message

Figure 4.12. A Web Service Provider processes a <Response> message

A <Response> message has been defined within the SAML specification. As it can meet the protocol design requirements as mentioned in section 4.3, it is applied to this protocol.

**Protocol message description:** a <Response> message contains a failed result, and this message can only generated by a Web Service Requester.

The detailed syntax and semantics of a <Response> message is presented in section A.6 in Appendix A.

### 4.6.6 Last step – Receives an <AuthzDecisionStatement> message from a Web Service Provider

**Internal Structure:** whenever a Web Service Requester receives this message, it should stop the communication with the counterpart (the logical process of the internal structure is shown in figure 4.13).

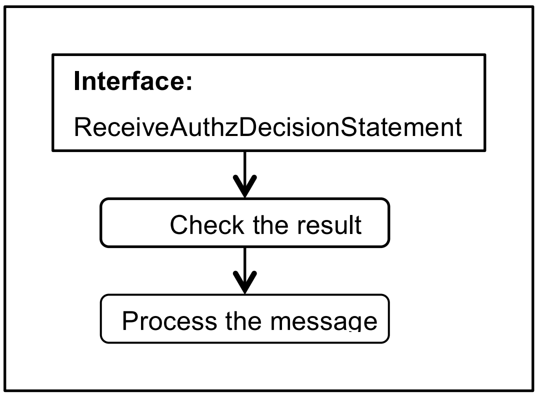


Figure 4.13. A Web Service Requester processes an <AuthzDecisionStatement> message

An <AuthzDecisionStatement> message has been defined within the SAML specification. As it can meet the protocol design requirements as mentioned in section 4.3, it is applied to this protocol.

**Protocol message description:** an <AuthzDecisionStatement> message contains the result information about TN. If TN has failed, a fault reason is also contained in this message.

The detailed syntax and semantics of an <AuthzDecisionStatement> message is presented in section A.7 in Appendix A.

## 4.7 Chapter Summary

This chapter initially presents a novel conceptual multi-layered interoperability-solution design for illustrating the use of a protocol-based approach providing interoperability for authorisation systems within Web Services taking into consideration interoperability at multiple layers (i.e. layers 2 to 6). A proposed protocol for addressing interoperability between authorisation systems within Web Services is provided as an example of utilising the interoperability-solution design. To ensure the correctness of the protocol, the process of design and development of this protocol strictly follows the interoperability-solution design and the process of the protocol design methodology as stated in section 4.1. In particular, the process includes step one and step two. According to step one of the methodology, the relevant protocol requirements are elicited based on a critical assessment of the literature review stated (Chapter 2 and Chapter 3) and the solution proposed in an interoperability-solution design as shown in table 4.1.

Following the elicited requirements, a novel stage “the preparation stage” is added to the proposed protocol. This stage can effectively aid two Web Services in reaching an agreement on a common strategy and a common language combination for the preparation of conducting a TN approach in the second stage. This agreement can guarantee that the second stage will not fail due to the strategic interoperability issue (interoperability at layer 6) or the capability interoperability issue (interoperability at layer 5). In addition, the protocol is designed to take into consideration its compatibility with the SAML specification and the XACML specification for achieving authorisation.

Within the second stage as called “the negotiation stage”, TN-related functionalities can be used for two unknown Web Services. In particular, the Web Service Provider can make an access control decision based on the TN process, whereas the Web Service Requester can also stop the authorisation process, if it discovers that it cannot submit any credentials to fulfil the policies disclosed by the Web Service Provider.

Additionally, in both stages, the service specification and protocol specification are presented. More specifically, the protocol specification constituting the relevant protocol messages (syntax and semantics of each communication messages) and internal structures (logical functionalities) are specified to provide syntactic (layer 2), semantic (layer 3) and functional (layer 4) interoperability for the two stages designed in the protocol.

Following step three of the protocol design and development methodology, protocol verification is needed. The next chapter discusses the reason for the need for a protocol verification test and provides a difference between the purpose of conducting protocol verification and that of proposed interoperability-solution evaluation. In addition, a discussion of selecting the most appropriate protocol verification method is also presented.

# Chapter 5. Protocol Verification

## 5.1 Introduction

Chapter 4 has presented a conceptual multi-layered interoperability-solution design (presented in table 4.1) illustrating how to use a protocol to provide interoperability from layers 2 to 6 between two communicating systems. Following the guidance of this design, an improved TN protocol for addressing interoperability issues between authorisation systems within Web Services is then proposed. The protocol design includes the service specification and protocol specification (internal structures and protocol messages).

As the protocol design and development strictly followed the guidance of the interoperability-solution design, if the effectiveness of the protocol can be proved, the usefulness of the design will in turn be proved as well. The proof of the effectiveness of the protocol is through evaluation of the protocol (presented in the chapter 7). Before presenting the protocol evaluation, protocol verification is indispensable according to step three of the protocol design and development methodology identified in section 4.1. In essence, protocol verification forms a foundation for protocol evaluation, as the effectiveness of a protocol can only be proved, if the protocol has been properly verified (Matsuo et al., 2010).

There might be confusion between protocol verification and protocol validation. An official IEEE guide (2011) explicitly points out a distinct difference between verification and validation. Validation is used to test whether a product, service etc. can meet the needs of customers or stakeholders (an external process), whereas verification is mainly used to test whether a product, service etc. can conform to the predefined requirements, regulations and so on (an internal process). As the improved TN protocol proposed in Chapter 4 is not designed following the requirements of specific customers or stakeholders, protocol validation is not suitable to be used in this research.

Protocol verification normally includes a completeness test (Sunshine et al., 1982; Chevalier and Vigneron, 2002) and a correctness test (Bhargavan, Obradovic and

Gunter, 2002; Yolum, 2004). A completeness test is used to verify whether a protocol has any design flaws, as any design flaws may affect the correctness of a protocol (Martin, Hill and Wood, 2003). In terms of the definition of protocol correctness, a variety of its definitions have been identified through a review of related work. For instance, Fabrega and Herzog (1998) state that protocol correctness can only be ensured, if communicating entities have agreed on the required information for the protocol. Rein and Fokkinga (1999) regard the definition of protocol correctness as interoperability. In other words, if each entity can successfully process a received message according to the protocol, protocol correctness can hold. Researchers such as Huang and Hsu (1994) and Yolum (2004) treat the definition of the protocol correctness as the same as the definition of protocol completeness. Chkliaev, Hooman and Stok (2000) take serializability as protocol correctness.

Although there are differences in the definition of protocol correctness listed above, a widely accepted definition of protocol correctness does exist. Debbabi (2004) recommends that protocol correctness can be proved, if the relevant properties can be supported, where the relevant properties are designed to achieve the goal of designing a protocol. Bella (2008) states, “A security protocol is correct if it lives up to the goals that its designer stated against specific threats”. There are other researchers who also support this definition of protocol correctness (e.g. Pironti, Pozza and Sisto, 2011; Jamroga and Melissen, 2014).

Reflecting on the purpose of proposing the improved TN protocol, it is designed following step one of the proposed interoperability-solution design to address the raised second research problem (see sections 1.2.1 and 3.7). Therefore, following the accepted definition of protocol correctness, correctness of the proposed protocol can hold, as long as successful or failed authorisation can occur according to the specific conditions pre-set to the two unknown entities. Each authorisation result by using this TN protocol should follow the rules of TN (i.e. the general concept of TN presented in section 3.2). Following this purpose, it can be identified that the purpose of the protocol correctness test in this research is a sub part included in the protocol evaluation in this research, which aims to verify whether the second research problem can be addressed. This chapter only details the protocol completeness test, with the interoperability-solution evaluation including the protocol correctness test presented in Chapter 7.

It should be noted that although a performance test (Linn and JR., 1989) is sometimes required in protocol verification, it is not delivered in this research. This decision was made taking into consideration that a performance test result cannot answer the second research question (see section 1.2.2).

To prove the completeness of the protocol specification, a discussion of selecting an appropriate method is presented first followed by an example for the illustration of the use of the selected method. An overview of the completeness proof of the protocol specification is then presented before a more detailed description of the proof is provided.

## 5.2 Discussion of Methods for the Completeness Proof

To demonstrate the completeness of the logical design of the protocol specifications, there are different optional formal methods that can be selected to use. To make a decision to select the most appropriate method, there is a need to analyse their benefits and limitations. Hidden Markov model (Baum and Petrie. 1966) is a method that can be used to discover hidden states in a system; this is often used to obtain the probability of each hidden state. However, in terms of the completeness proof in this research, the researcher of this Thesis only wanted to prove that all of the hidden states in this protocol could be identified, but had no interest in knowing the probability of reaching these states. From this perspective, this method was not appropriate to be used for proving completeness.

Another formal method is called the Finite State Machine (referred to as FSM hereafter). This enables a check to be completed on the reachability of different states (Bochmann and Gecsei, 1977; Bochmann, 1978; Sunshine, 1979; Sunshine et al., 1982). Reachability can be checked with the use of a reachability tree diagram (Peterson, 1977) called a State Transition Diagram (referred to as STD hereafter) (Danthine, 1980); A STD contains all of the possible states generated from the protocol specifications. The purpose of using this formalism is to check whether such a diagram is complete.

By using FSM, the discovery of potential deadlocks, assessment of the liveness of a state, and the identification of loops is feasible. Drawbacks exist to this method, and must be avoided to ensure the completeness (Merlin, 1976; Bochmann, 1978; Merlin, 1979; Palmer and Sabnani, 1986). A deadlock means that no further transition can occur from the current state. Liveness of a state requires that all of the possible states can be reached from the initial state. Loops, sometimes are necessarily required by the logical design of the protocol specifications, but may result in an infinite iteration. Different solutions are required to avoid unexpected infinite iterations.

There also exist other formal methods that can be used to test the completeness of a protocol. Example methods include: enumeration of finite shapes (Doghmi, Guttman and Thayer, 2007) and constraint satisfaction procedure (Millen and Shmatikov, 2001) etc. After an analysis of their features, they provide similar effects as supplied by FSM in terms of a protocol completeness test, as their conceptual ideas are quite similar to the idea used in FSM, where terms used in these methods are different. In other words, they can be treated as variations of FSM. Therefore, given the limited differences between methods at a conceptual level, this Thesis uses FSM as the method to provide a completeness test.

After a discussion of the selected method for the completeness proof, the next section presents an introduction of an example FSM to illustrate how this method is used in this research.

## 5.3 Introduction of An Example FSM

As mentioned earlier, the formalism called FSM has been selected to demonstrate the completeness of the protocol. However, before the demonstration, a clear definition of the completeness should be given. To define the completeness of the diagram, an example STD (shown in figure 5.1) is used to help clarify the definitions given below:

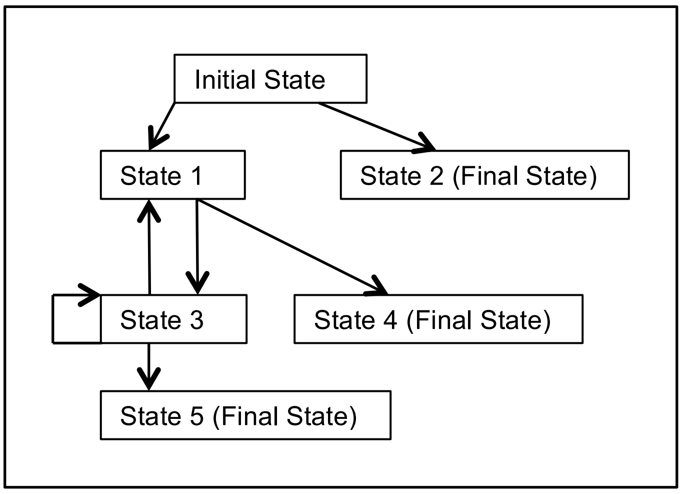


Figure 5.1. An example STD

**Definition 1: Transition:** is a link from a current state to the next state.

**Definition 2: One-way transition:** is a transition from a current state to the next state, where a reverse transition is not possible. The symbol → expresses a one-way transition. An example one-way transition in figure 5.1 is the first transition starting from the initial state pointing to state 1. This example is denoted as: initial state → state 1.

**Definition 3: Two-way transition:** is a transition from a current state to the next state, where a reverse transition is possible. The symbol ↔ expresses a two-way transition. An example two-way transition in figure 5.1 is the transition from state 1 to state 3 and from state 3 back to state 1. This example is denoted as: state 1↔ state 3.

**Definition 4: Self-Transition:** is a transition from a current state to itself directly. The symbol $\lrcorner_{ST}$ expresses a self-transition. An example self-transition in figure 5.1 is the transition from state 3 to itself directly. This example is denoted as: state 3$\lrcorner_{ST}$.

**Definition 5: Transition Chain:** is a set of continuous transitions. An example transition chain in figure 5.1 is that the first transition starts from the initial state pointing to state 1, from which a second continuous transition pointing to state 3. This example is denoted as: initial state → state 1 → state 3.

**Definition 6: Initial State**: is the first state triggered in a STD.

**Definition 7: Final State**: is the last state, where no more transitions will occur. For instance, in figure 5.1, states 2, 4 and 5 are all final states.

**Definition 8: Intermediate State:** is a state at which a transition can arrive from a previous state, and from which a path can reach the next state. For instance, in figure 5.1, states 1 and 3 are intermediate states.

**Definition 9: Sibling States**: are the optional states, through which a transition can arrive from the same previous state. In the STD, a sibling state can be either an intermediate state or a final state. The decision to reach which sibling state depends on the specific context variables. For instance, states 1 and 2 are sibling states to each other, and states 3 and 4 are also sibling states to each other.

**Definition 10: Self-Transition State:** is a state on which a self-transition occurs. State 3 in figure 5.1 is an example self-transition state.

**Definition 11: Path**: is a transition chain starting from the initial state reaching a final state or a duplicate state. Within the transition chain, the number of the transitions is equivalent to or greater than one. In particular, when the number equals to one, only one transition exists in the set of transitions. It links from the initial state directly to the final state. When the number is greater than one, the transitions in the transition chain must be linked to or from at least one intermediate state. For instance, in figure 5.1, an example path can be a path either starting from the initial state directly pointing to state 2, or from the initial state pointing to state 1 and in turn pointing to state 4. This example is denoted as: Initial state → state 2 or Initial State → State 1→ State 4.

**Definition 12: Infinite Transitions:** are transitions from two or more repeated states to form infinite loops, so a transition to a final state is impossible. For instance, the third path above may cause infinite transitions, if there is no approach that can enable the transition from state 3 to itself or state 5 other than state 1. Thus, if there exist infinite transitions in a STD, and no pertinent approach can break them, the completeness of the STD cannot hold.

**Definition 13: Diagram Completeness**: Starting from the initial state, whenever traversing through an intermediate state, a final state can always be reached to guarantee the existence of a path. If all of the different paths can be found, when all of the sibling states are involved according to the service specification, and if there exists infinite transitions that can be broken by relevant approaches, the diagram completeness will hold.

Researchers such as Merlin (1976), Peterson (1977) and Merlin (1979) recommend one formal method called Petri Nets, which can be used for presenting all of the different paths of a protocol. However, after an analysis of this method, a conclusion was drawn that, unfortunately, this method was insufficient to support the presentation such as two-way transition or self-transition, it is therefore not used in this chapter. Instead, all of the different paths of the example STD shown in figure 5.1 are expressed by using the chosen symbols illustrated in the examples above. Figure 5.2 lists all of the paths of the example STD by using the chosen symbols.

1. Initial State $\rightarrow$ State 2.
2. Initial State $\rightarrow$ State 1 $\rightarrow$ State 4.
3. Initial State $\rightarrow$ State 1 $\leftrightarrow$ State 3$_{\hookleftarrow|ST}$ $\rightarrow$ State 5.

Figure 5.2. Completeness of the example STD

The completeness of the example STD as shown in figure 5.2 can only be proved, if there is at least one approach that can break the infinite transitions between states 1 and 3 and between the self-transition of state 3. Otherwise, the diagram completeness cannot be proved. After the introduction of an example STD along with the definition of diagram completeness, the completeness of the STD of the protocol is illustrated in the next section.

## 5.4 FSM-based Completeness Proof of the Protocol

### 5.4.1 Overview of the states of the protocol

An overview of the states of the protocol is presented in table 5.1. According to the protocol specification presented in Chapter 4, there are two stages designed in this protocol. Stage one is called “the preparation stage”, and stage two is called “the negotiation stage”. In each stage, there are several protocol messages defined within it. Within the completeness test by using the FSM, a state is represented as the time when a protocol message is completely processed according to its relevant internal structure and a decision has to be made whether to send out a message or not.

| **Preparation Stage** |
|---|
| After processing a <TNPrepareRequest> message and sending out a <TNPrepareResponse> message (state 1 and state 2) |
| **Negotiation Stage (Optional 2)** |
| After processing an <AuthzDecisionQuery> message and sending out an <AuthzDecisionStatement> message (state 3) or a <PolicySet> message (state 4) |
| After processing a <PolicySet> message (state 4) and sending out a possible message |
| After processing a <CredentialSet> message (state 5, state 6, state 8 and state 9) and sending out a possible message |
| After processing a <Response> message (state 7 and state 10) and sending out an <AuthzDecisionStatement> message |
| After processing an <AuthzDecisionStatement> message (state 7, state 10 and state 11) |

Table 5.1. An overview of states designed in the protocol

An overview of the preparation stage using the FSM is presented in figure 5.3. As the process of the negotiation stage will differ, when the parsimonious strategy and the eager strategy are used, an overview of the negotiation stage of the two optional strategies is presented in figures 5.4 and 5.5. Figures 5.3, 5.4 and 5.5 relate directly to figure 4.2 primarily through the condition statements displayed on each of the links. For instance, in figure 5.3, a <TNPrepareRequest> message and a <TNPrepareResponse> message shown in the condition between the initial state and state 1 or state 2 can also be found in step one and step two of stage 1 of the protocol as shown in figure 4.2.

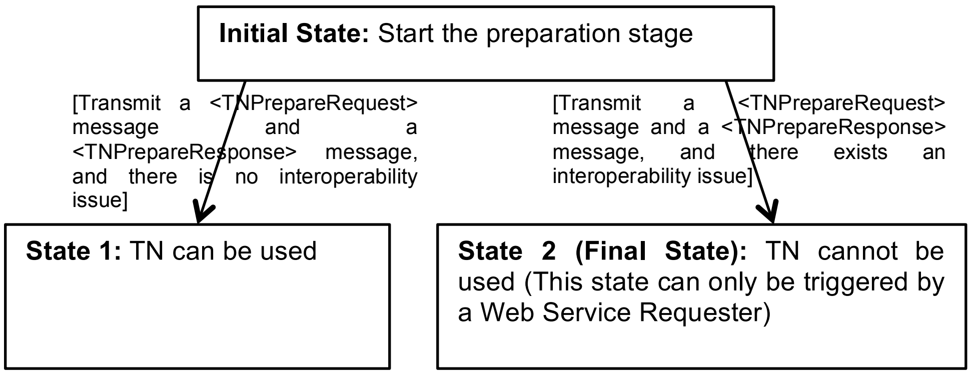

**Initial State:** Start the preparation stage
[Transmit a <TNPrepareRequest> message and a <TNPrepareResponse> message, and there is no interoperability issue]
[Transmit a <TNPrepareRequest> message and a <TNPrepareResponse> message, and there exists an interoperability issue]
**State 1:** TN can be used
**State 2 (Final State):** TN cannot be used (This state can only be triggered by a Web Service Requester)

Figure 5.3. The entire STD of the preparation stage of the protocol

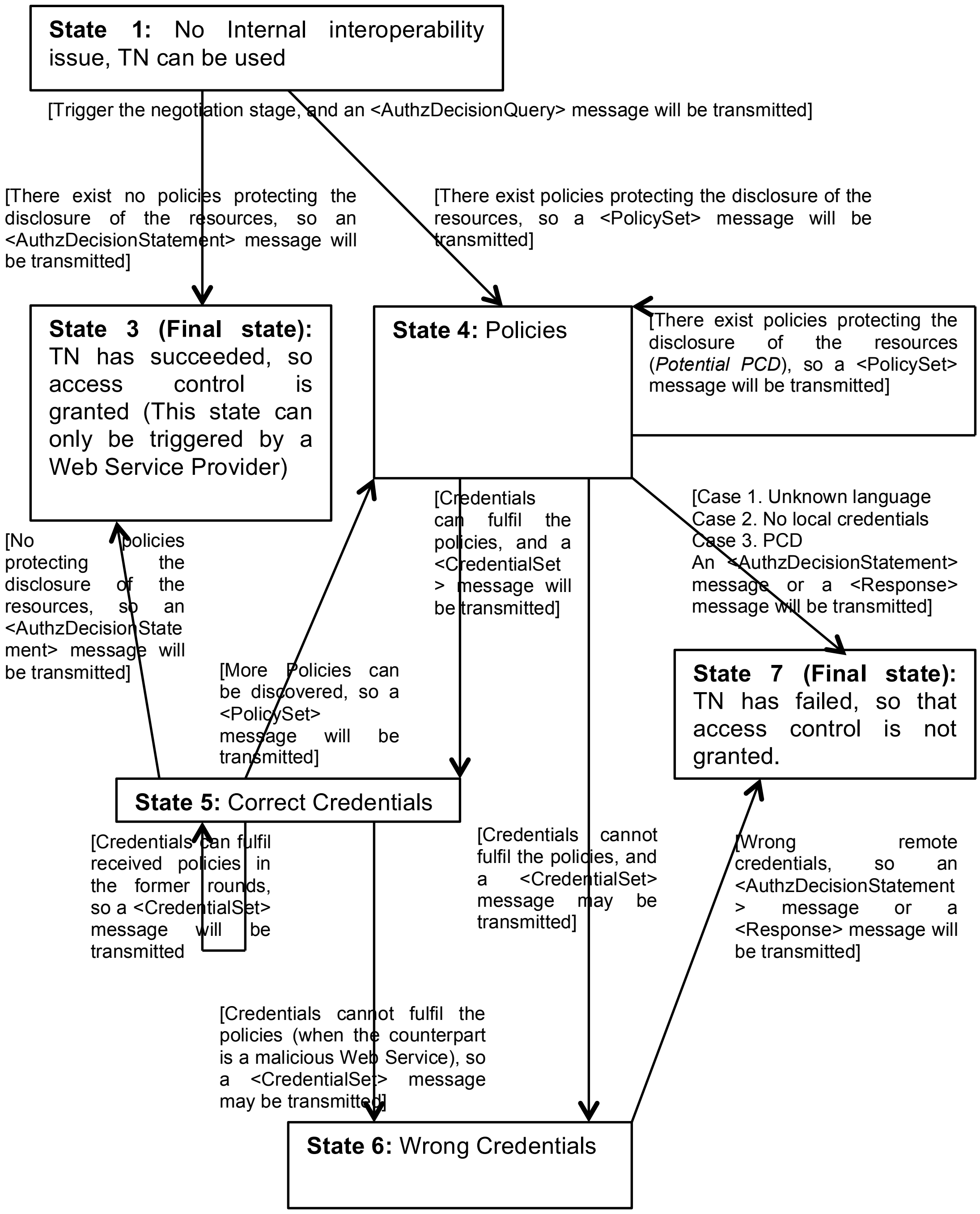

State 1: No Internal interoperability issue, TN can be used
[Trigger the negotiation stage, and an <AuthzDecisionQuery> message will be transmitted]
[There exist no policies protecting the disclosure of the resources, so an <AuthzDecisionStatement> message will be transmitted]
[There exist policies protecting the disclosure of the resources, so a <PolicySet> message will be transmitted]
State 3 (Final state): TN has succeeded, so access control is granted (This state can only be triggered by a Web Service Provider)
State 4: Policies
[There exist policies protecting the disclosure of the resources (Potential PCD), so a <PolicySet> message will be transmitted]
[Credentials can fulfil the policies, and a <CredentialSet> message will be transmitted]
[Case 1. Unknown language Case 2. No local credentials Case 3. PCD An <AuthzDecisionStatement> message or a <Response> message will be transmitted]
[No policies protecting the disclosure of the resources, so an <AuthzDecisionStatement> message will be transmitted]
[More Policies can be discovered, so a <PolicySet> message will be transmitted]
State 7 (Final state): TN has failed, so that access control is not granted.
State 5: Correct Credentials
[Credentials can fulfil received policies in the former rounds, so a <CredentialSet> message will be transmitted
[Credentials cannot fulfil the policies, and a <CredentialSet> message may be transmitted]
[Wrong remote credentials, so an <AuthzDecisionStatement> message or a <Response> message will be transmitted]
[Credentials cannot fulfil the policies (when the counterpart is a malicious Web Service), so a <CredentialSet> message may be transmitted]
State 6: Wrong Credentials

Figure 5.4. The entire STD of the negotiation stage of the protocol, while the parsimonious strategy is used

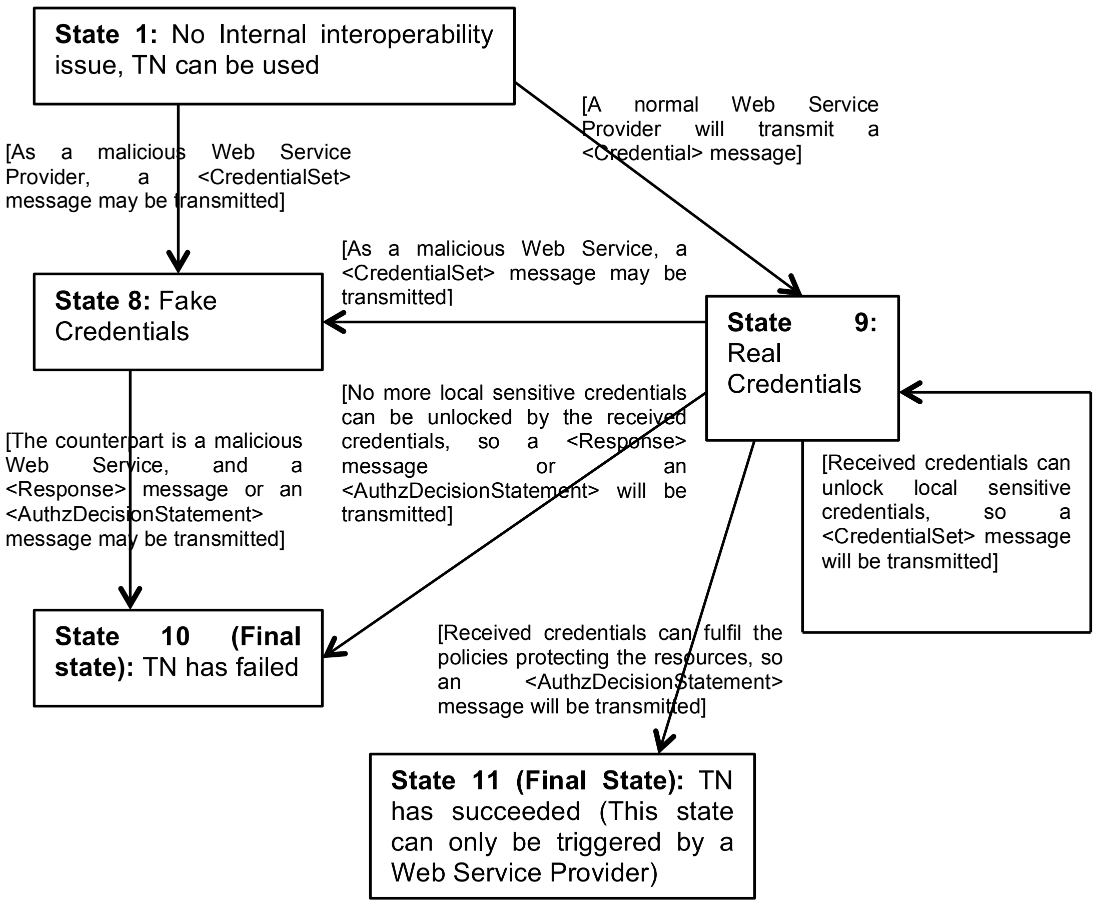

State 1: No Internal interoperability issue, TN can be used
[A normal Web Service Provider will transmit a <Credential> message]
[As a malicious Web Service Provider, a <CredentialSet> message may be transmitted]
[As a malicious Web Service, a <CredentialSet> message may be transmitted]
State 8: Fake Credentials
State 9: Real Credentials
[No more local sensitive credentials can be unlocked by the received credentials, so a <Response> message or an <AuthzDecisionStatement> will be transmitted]
[The counterpart is a malicious Web Service, and a <Response> message or an <AuthzDecisionStatement> message may be transmitted]
[Received credentials can unlock local sensitive credentials, so a <CredentialSet> message will be transmitted]
State 10 (Final state): TN has failed
[Received credentials can fulfil the policies protecting the resources, so an <AuthzDecisionStatement> message will be transmitted]
State 11 (Final State): TN has succeeded (This state can only be triggered by a Web Service Provider)

Figure 5.5. The entire STD of the negotiation stage of the protocol, while the eager strategy is used

Combining the FSM in figures 5.3, 5.4 and 5.5 together, all of the paths generated from the protocol are presented in figure 5.6.

Initial State → state 2 (final state).

Initial State → state 1 → state 3 (final state).

Initial State → state 1 → state 4↲$_{ST}$ → state 7 (final state).

Initial State → state 1 → state 4↲$_{ST}$ ↔ state 5↲$_{ST}$ → state 3 (final state).

Initial State → state 1 → state 4↲$_{ST}$ → state 6 → state 7 (final state).

Initial State → state 1 → state 4↲$_{ST}$ ↔ state 5↲$_{ST}$ → state 6 → state 7 (final state).

Initial State → state 1 → state 8 → state 10 (final state).

Initial State → state 1 → state 9↲$_{ST}$ → state 8 → state 10 (final state).

Initial State → state 1 → state 9↲$_{ST}$ → state 10 (final state).

Initial State → state 1 → state 9↲$_{ST}$ → state 11 (final state).

Figure 5.6. All of the paths of STD of the protocol

It can be observed from figure 5.6 that each path is finished at a state as a final state. According to definition 13 above, this is the first condition required by the diagram completeness. As the second condition, to verify the diagram completeness, a proof that all of the potential infinite transition points can be broken has to be provided. Clarification of the second condition and the correctness of the construction of the FSM of the protocol presented in figures 5.3, 5.4 and 5.5 is discussed in detail in the following sections.

### 5.4.2 FSM-based completeness proof of the preparation stage

According to the proposed protocol, the preparation stage is triggered first to allow two Web Services to reach an agreement about whether TN can be used. The initial state is triggered, when a Web Service Requester sends a <TNPrepareRequest> message to let the Web Service Provider check whether or not there is an interoperability issue at layers 5 or 6 (i.e. strategic or capability layer). Once the Web Service Provider has made a decision based on the information provided in the <TNPrepareRequest> message, it will inform the Web Service Requester of the result in a <TNPrepareResponse> message. This result includes two possibilities: (1) TN cannot be used due to an identified interoperability issue (e.g. strategy or language interoperability issue) and (2) TN can be used without any interoperability issues. Thus, starting from the initial state, two sibling states can be reached. The first possible state (referred to as state 1) is that TN can be used, since no interoperability issue has been discovered. State 1 is an intermediate state, since it will trigger the negotiation stage. The second possible state  (referred to as state 2) is that TN cannot be used due to the existence of an identified interoperability issue. State 2 is a final state, since the negotiation stage will not be triggered. The construction of this partial STD presenting the possible transitions from the initial state is shown in figure 5.7 below.

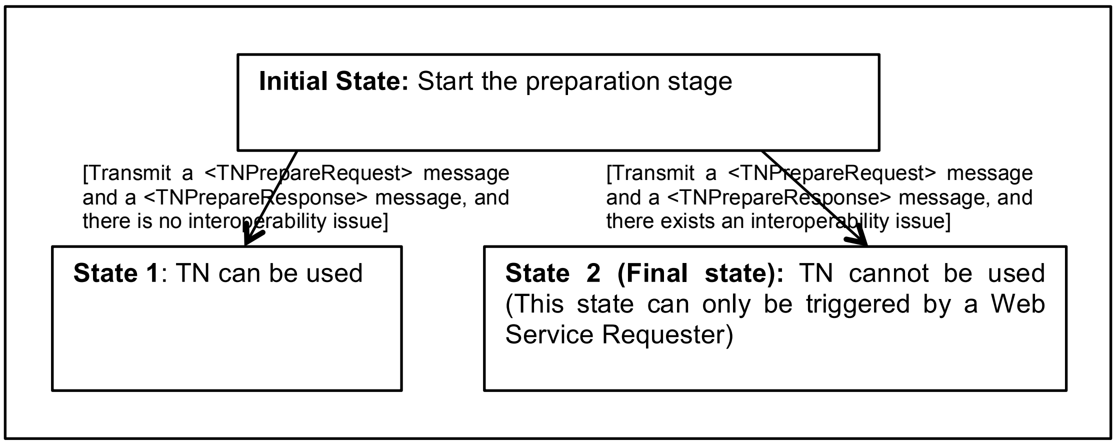

**Initial State:** Start the preparation stage
[Transmit a <TNPrepareRequest> message and a <TNPrepareResponse> message, and there is no interoperability issue]
[Transmit a <TNPrepareRequest> message and a <TNPrepareResponse> message, and there exists an interoperability issue]
**State 1**: TN can be used
**State 2 (Final state):** TN cannot be used (This state can only be triggered by a Web Service Requester)

Figure 5.7. Two transitions from the initial state

Starting from state 1, the second stage called “the negotiation stage” will be triggered, once the Web Service Requester sends an <AuthzDecisionQuery> message detailing information about the requested resource to the Web Service Provider. According to the internal structures of the negotiation stage of the protocol, internal structures for processing various incoming messages are different, when different strategies (i.e. a eager strategy and a parsimonious strategy) are used. This phenomenon requires a discussion of state transitions in the eager strategy to be separated from that in the parsimonious strategy. The scenario of using a parsimonious strategy is discussed first in the next section.

### 5.4.3 FSM-based completeness proof of the negotiation stage with the use of a parsimonious strategy

With the use of the parsimonious strategy, two transitions are possible. If the requested resource is not protected by any policies, a possible state (referred to as state 3) is that the resource can be disclosed with the last message (the <AuthzDecisionStatement> message). In other words, TN has succeeded, since the resource is set as a non-sensitive resource that is publicly accessible. Thus, state 3 is a final state. If the requested resource is protected by its policies, a state (referred to as state 4) that will be reached is an intermediate state, in which the entity will disclose the policies protecting the resources. The construction of this partial STD presenting the possible transitions from state 1 with the use of the parsimonious strategy, is shown in figure 5.8 below:

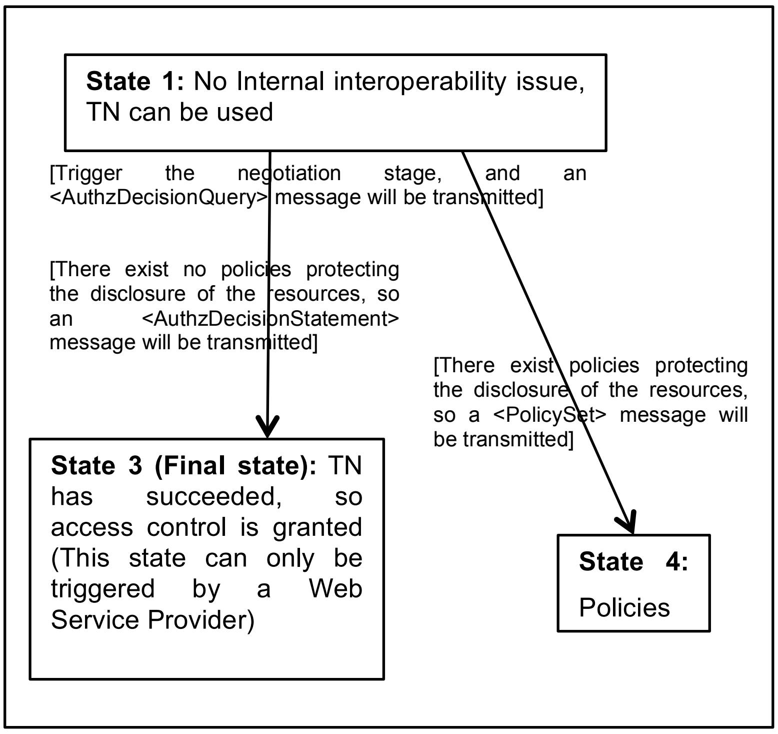

State 1: No Internal interoperability issue, TN can be used
[Trigger the negotiation stage, and an <AuthzDecisionQuery> message will be transmitted]
[There exist no policies protecting the disclosure of the resources, so an <AuthzDecisionStatement> message will be transmitted]
[There exist policies protecting the disclosure of the resources, so a <PolicySet> message will be transmitted]
State 3 (Final state): TN has succeeded, so access control is granted (This state can only be triggered by a Web Service Provider)
State 4: Policies

Figure 5.8. Transitions from state 1, when the parsimonious strategy is used

As discussed in section 4.6.3, according to the internal structures of the negotiation stage with the use of the parsimonious strategy, once a Web Service receives remote policies requiring local credentials, four cases will occur. Case 1: The local credentials that can fulfil the remote policies are sensitive. They are protected by other policies, so the entity will disclose policies in a <PolicySet> message. Case 2: The local credentials that can fulfil remote policies are not sensitive, so the entity will disclose credentials in a <CredentialSet> message. Case 3**:** No local required credentials can be found, so the entity will send a <Response> message (sent by a Web Service Requester) or an <AuthzDecisionStatement> message (sent by a Web Service Provider) containing a fault reason. Case 4: PCD may be found, so the entity will send a <Response> message or an <AuthzDecisionStatement> message containing a fault reason (see figure 6.1 shown in section 6.3.3). However, as mentioned in section 3.3.3, a malicious Web Service may be involved. Thus, another two cases may occur. Case 5: If a Web Service does not hold any local credentials that can fulfil the remote policies, but it still sends some unrelated credentials to an entity. The entity will send a <Response> message (as a Web Service Requester) or an <AuthzDecisionStatement>message containing (as a Web Service Provider) a fault reason. Case 6: The received remote policies are written in unknown languages, so the entity will send a <Response> message or an <AuthzDecisionStatement>message containing a fault reason.

As some of the cases mentioned above can be represented as a single state, the six cases form four sibling states from state 4. The first possible state is a self-transition state of state 6 that the entity will send policies. This self-transition of state 6 may cause infinite transitions. In particular, if iteration exists in this process, the two entities may keep sending the same policies to each other. This means that the transition from the current state (disclosing policies) to the next state (disclosing policies) will cause PCD (see section 3.3.1.2) between two entities. Such PCD occurring in the improved TN protocol are typical infinite transitions, which can break the completeness of the improved TN protocol. Unfortunately, there is no relevant solution designed within the protocol to enable an entity to automatically detect the occurrence of PCD. Therefore, it requires a novel solution that can be embedded into the protocol to resolve this issue; otherwise, the completeness of the protocol cannot hold.

The second state (referred to as state 5) is an intermediate state that the entity will send correct credentials in a <CredentialSet> message that can fulfil remote policies. The third state (referred to as state 6) is an intermediate state that the entity will send wrong credentials in a <CredentialSet> message, if this entity is a malicious Web Service, or the logical design of its policy compliance checker is wrong. The fourth state (referred to as state 7) is a final state that TN has failed. Three reasons can lead to the transition from state 6 to reach this state. Reason 1: remote policies are written in an unknown language (see reason two contained in a <Fault> element of an <AuthzDecisionStatement> message in section A.7). Reason 2: there is no local credential that can fulfil remote policies (see reason three contained in a <Fault> element of an <AuthzDecisionStatement> message in section A.7). Reason 3: potential PCD may be identified (see reason four contained in a <Fault> element of an <AuthzDecisionStatement> message in section A.7). The construction of this partial STD presenting possible transitions from state 4 is shown in figure 5.9 below:

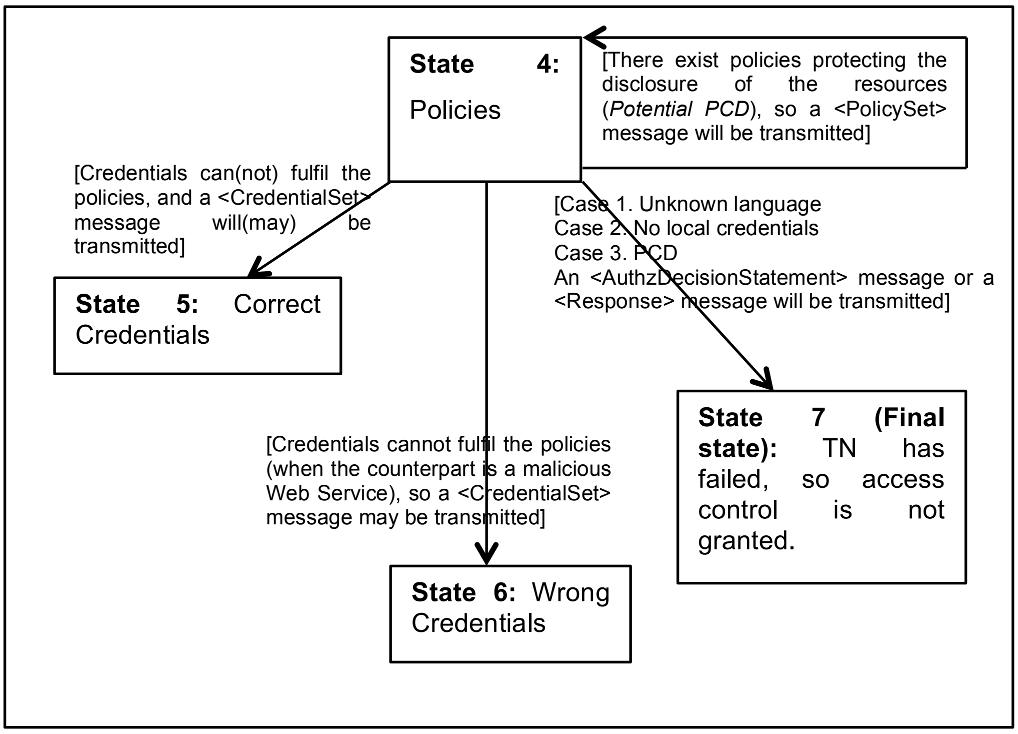

State 4: Policies
[There exist policies protecting the disclosure of the resources (*Potential PCD*), so a <PolicySet> message will be transmitted]
[Credentials can(not) fulfil the policies, and a <CredentialSet> message will(may) be transmitted]
[Case 1. Unknown language
Case 2. No local credentials
Case 3. PCD
An <AuthzDecisionStatement> message or a <Response> message will be transmitted]
State 5: Correct Credentials
[Credentials cannot fulfil the policies (when the counterpart is a malicious Web Service), so a <CredentialSet> message may be transmitted]
State 7 (Final state): TN has failed, so access control is not granted.
State 6: Wrong Credentials

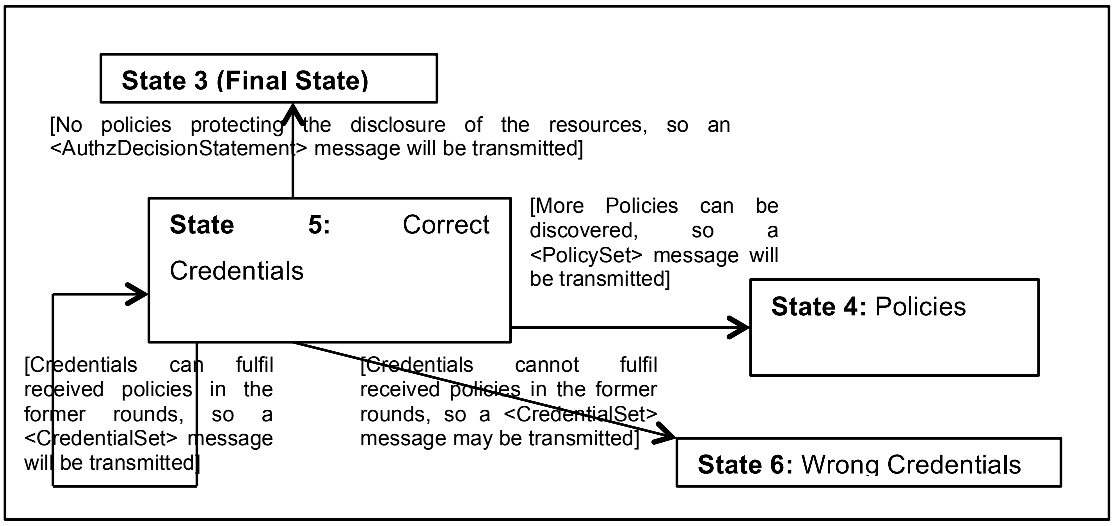
Figure 5.9. Four possible transitions from state 4

State 5 indicates the circumstance that an entity has received the correct remote credentials. From this state, there are four states that can be reached (see figure 4.11). The first possible state is state 4 that remote credentials have fulfilled local policies that have been sent to the counterpart, at which point the entity may send further local sensitive policies in a <PolicySet> message. The transition from state 5 back to state 4 indicates that the transition between the two states is a two-way transition, which explores a potential infinite iteration that has never been identified in the state-of-the-art TN-based authorisation systems. For instance, if one Web Service is maliciously designed, it may send out infinite different policies requesting the same credentials from the counterpart. Unfortunately, there is no solution designed within the protocol that can enable an honest Web Service to automatically detect such an attack. The identification of this weakness breaks the completeness of the protocol for a second time (no automatic detection for PCD identified above is the first time breaking the completeness of the protocol).

The second possible state is itself - a self-transition state that remote credentials have fulfilled local policies that have been sent to the counterpart, so the entity may send credentials that are required by the remote policies in a recent round in a <CredentialSet> message. It seems that another infinite transition may occur between the self-transition state 5 and itself, but this infinite transition will not occur in the process of the negotiation stage, if at least one of the Web Service is honestly designed (the circumstance that two Web Services are both maliciously designed is outside consideration, as TN is mainly used to help two unknown entities to establish a bilateral trust relationship rather than to enable two maliciously designed Web Services to attack each other). The reason is illustrated as follows.

The transition from state 5 to itself can occur, only if the received credentials can fulfil local policies. If the two Web Services are both honestly designed, infinite exchange of policies should not occur, which will in turn result in a finite exchange of credentials in this first instance. If one Web Service is maliciously designed and the other one is honestly designed, at least the number of credentials sent by the honestly designed Web Service is finite. Therefore, in this second instance, infinite exchange of credentials should not occur either.

The third possible state is state 6 that wrong credentials may be sent out in a <CredentialSet> message. The fourth possible state is state 3 as a final state that remote credentials cannot fulfil local policies, so the entity will send a <Response> message (as a Web Service Requester) or an <AuthzDecisionStatement> message (as a Web Service Provider) containing a fault reason. The construction of this partial STD presenting the possible transitions from state 5 is shown in figure 5.10 below.



State 3 (Final State)
[No policies protecting the disclosure of the resources, so an <AuthzDecisionStatement> message will be transmitted]
State 5: Correct Credentials
[More Policies can be discovered, so a <PolicySet> message will be transmitted]
State 4: Policies
[Credentials can fulfil received policies in the former rounds, so a <CredentialSet> message will be transmitted]
[Credentials cannot fulfil received policies in the former rounds, so a <CredentialSet> message may be transmitted]
State 6: Wrong Credentials

Figure 5.10. Three possible transitions from state 5

In figure 5.10, from state 6, there exists only one transition to the next state that TN is not successful, since the received credentials cannot fulfil local policies (see reason one contained in a <Fault> element in an <AuthzDecisionStatement> message in section A.7). Thus, the reached state is state 7 as a final state. The construction of this partial STD presenting the possible transitions from state 6 is shown in figure 5.11 below.

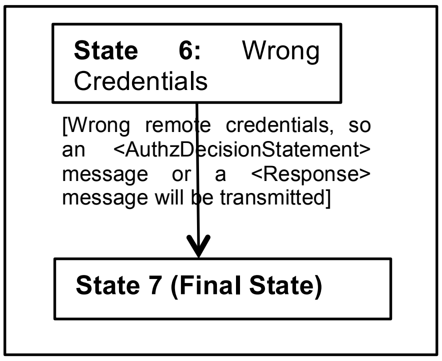

State 6: Wrong Credentials
[Wrong remote credentials, so an <AuthzDecisionStatement> message or a <Response> message will be transmitted]
State 7 (Final State)

Figure 5.11. One transition from state 6

As the state transitions with the use of the parsimonious strategy have been discussed above, the next section discusses the state transitions with the use of an eager strategy.

### 5.4.4 FSM-based completeness proof of the negotiation stage with the use of an eager strategy

When the eager strategy is used (see figures 4.6, 4.9 and 4.10), two transitions will be produced from state 1 (as the first state in the negotiation stage, see figure 5.3 and 5.7). The first possible state (referred to as state 8) is that fake credentials (cannot pass the credential authenticity checking as mentioned in section 4.6.4, see figure 4.12) may be disclosed in a <CredentialSet> message. This state may occur, when the Web Service Provider is maliciously designed, or the internal structure is wrongly designed. The second possible state (referred to as state 9) is that real credentials will be disclosed in a <CredentialSet> message. The construction of this partial STD presenting the possible transitions from state 1 with the use of the eager strategy is shown in figure 5.12 below.

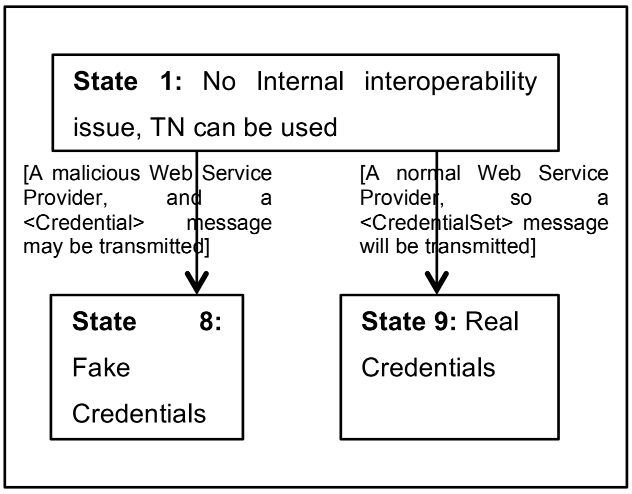

State 1: No Internal interoperability issue, TN can be used
[A malicious Web Service Provider, and a <Credential> message may be transmitted]
[A normal Web Service Provider, so a <CredentialSet> message will be transmitted]
State 8: Fake Credentials
State 9: Real Credentials

Figure 5.12. Transitions from state 1, when the eager strategy is used

When state 8 has been reached, a final state (referred to as state 10) is the only state that can be transitioned from state 8. The fake credentials can be detected by using the "Credential Verification Component" in section 4.6.4 (see figure 4.9). The Web Service Requester/Provider must terminate TN by sending a <Response> /<AuthzDecisionStatement> message containing a fault reason (see reason one contained in a <Fault> element of a <Response>/<AuthzDecisionStatement> message in section A.6/A.7). The construction of this partial STD presenting the possible transitions from state 8 is shown in figure 5.13 below.

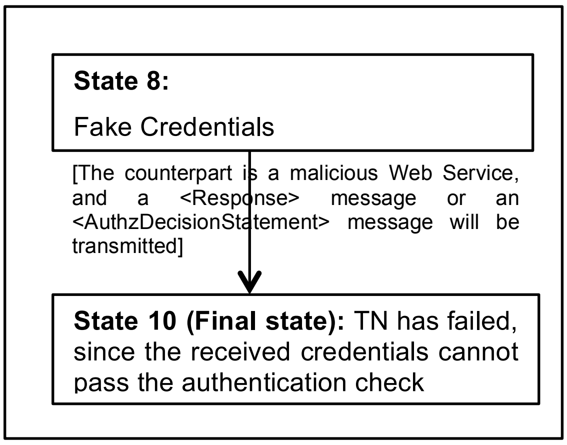

State 8: Fake Credentials
[The counterpart is a malicious Web Service, and a <Response> message or an <AuthzDecisionStatement> message will be transmitted]
State 10 (Final state): TN has failed, since the received credentials cannot pass the authentication check

Figure 5.13. A transition from state 8

If real credentials have been received as indicated in state 9, there are four possible transitions generating different states. The first possible state is state 8, where fake credentials contained in a <CredentialSet> message may be sent out. The condition of the occurrence of the two cases is the same as that mentioned in state 8, in which the counterpart is a maliciously designed Web Service or the policy compliance checkers are wrongly designed; otherwise, this state cannot be reached, if both Web Services are correctly designed following the guidance of the eager strategy. The second possible state is itself as a self-transition state. It seems that there is another infinite iteration occurring on the self-transition state 9, but this infinite iteration will be avoided due to the intrinsic nature of the eager strategy. For instance, if one of the Web Services discloses credentials strictly following the eager strategy, it will eventually terminate TN, when it discovers that no more local sensitive credentials can be unlocked. This result will not be changed regardless of whether the counterpart is maliciously designed or not. The third possible state (referred to as state 11) is a final state indicating that TN has succeeded. This state can only be triggered by a Web Service Provider, which has discovered that the received credentials have fulfilled all of the policies protecting the requested resource. The fourth possible state is state 10 as a final state. At this state, a Web Service, as a Service Requester will send a <Response> message, or as a Service Provider, will send an <AuthzDecisionStatement> message containing a fault reason (see reason one contained in a <Fault> element of a <Response>/<AuthzDecisionStatement> message in section A.6/A.7.) informing that TN has failed. This state will be reached, if a Web Service (can be either a Web Service Requester and a Web Service Provider) discovers that no more local sensitive credentials can be unlocked by the received credentials. The construction of this partial STD presenting the possible transitions from state 9 is shown in figure 5.14 below.

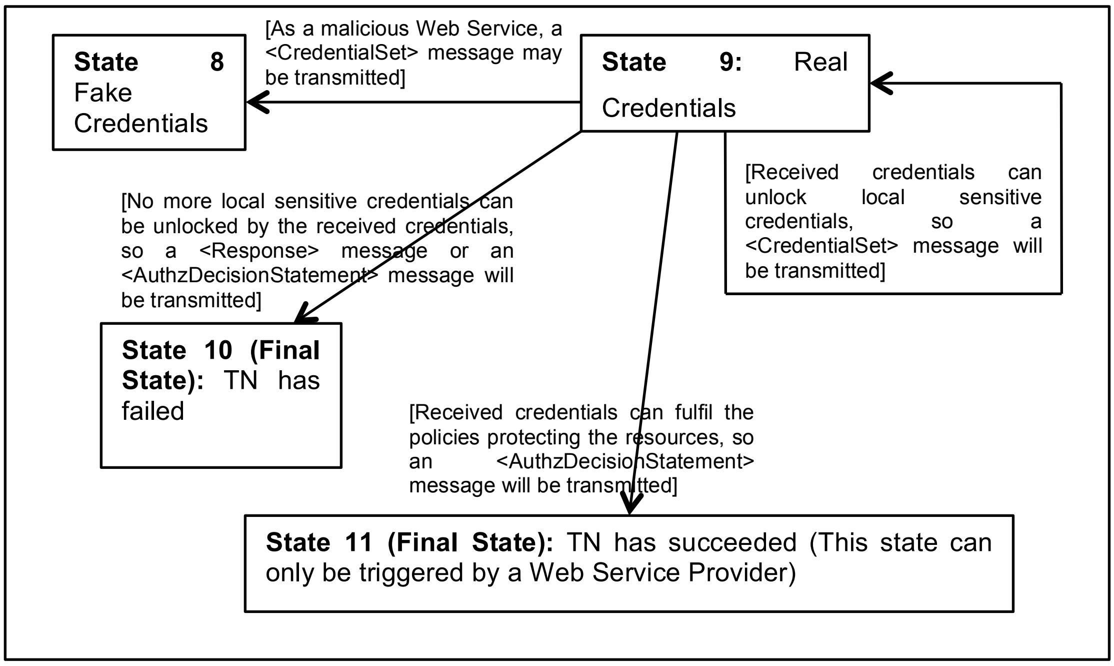

State 8 Fake Credentials
[As a malicious Web Service, a <CredentialSet> message may be transmitted]
State 9: Real Credentials
[Received credentials can unlock local sensitive credentials, so a <CredentialSet> message will be transmitted]
[No more local sensitive credentials can be unlocked by the received credentials, so a <Response> message or an <AuthzDecisionStatement> message will be transmitted]
State 10 (Final State): TN has failed
[Received credentials can fulfil the policies protecting the resources, so an <AuthzDecisionStatement> message will be transmitted]
State 11 (Final State): TN has succeeded (This state can only be triggered by a Web Service Provider)

Figure 5.14. Four transitions from state 9

## 5.5 Identified Innate Vulnerabilities

Through the detailed process of the protocol completeness verification, it has been identified that the improved TN protocol is not complete, since two types of infinite iteration may exist during the conduction of TN. As the current functionalities designed within the protocol cannot break the infinite iteration, the second condition of the protocol completeness cannot be fulfilled. The two types of infinite iteration as innate vulnerabilities have been explored within figure 5.10. The first vulnerability is that no functionality is designed within this protocol to enable two Web Services to automatically detect the occurrence of potential PCD. The second vulnerability is that no functionality is supported by this protocol to enable a honestly designed Web Service to defend against attacks, if its counterpart keeps sending different policies requesting the same local credentials.

## 5.6 Impact of the FSM Approach

The FSM as a formalism was initially used to verify the completeness of a protocol. When the protocol was being verified through the process, the initial context of the protocol was that the two Web Services were honest Web Services. In other words, if developers implemented this protocol strictly following the protocol specification, none of the potential infinite iteration discovered in the verification results would occur. For instance, the occurrence of PCD may occur between the honest Web Services in practice, but the occurrence of the attacks as the above-identified second vulnerability issue should not occur in practice. However, discovery of unexpected infinite iteration through the use of the FSM on the protocol completeness test gave rise to consideration other possible protocol application contexts (i.e. TN communication may not only occur between two honest Web Services, but also occur between one malicious Web Service and one honest Web Service). In other words, FSM can not only be utilised to effectively identify potential infinite iteration, but can also be leveraged to enable protocol developers to explore implicit vulnerability issues existing within a protocol. Therefore, the effect of applying FSM on a protocol to explore these potential vulnerability issues might be widely used in other communication protocol designs.

## 5.7 Chapter Summary

This chapter presents protocol verification for demonstration of the protocol completeness according to step three of the protocol design and development methodology. After an analysis of the available approaches, FSM has been selected as the most appropriate approach for the demonstration of the protocol completeness. An overview and detailed completeness proof are presented and discussed. The completeness proof is necessary, as it forms a foundation for the correctness test or the protocol evaluation, which aims to prove whether or not the proposed protocol can provide interoperability at the relevant layers as announced in the interoperability-solution design presented in table 4.1. Unfortunately, through the completeness verification process, two innate vulnerabilities have been identified within the protocol. Without relevant solutions, the protocol cannot be proved to be complete. Exploration of a potential solution for resolving the two vulnerabilities is discussed in the next chapter.

# Chapter 6. Remembrance of Local Information Status for Enforcing Robustness of Policy-Exchanged Strategies for Trust Negotiation

## 6.1 Introduction

Detailed protocol verification has been presented in Chapter 5. As identified in the verification results, two innate vulnerability issues still exist within the improved TN protocol as well as state-of-the-art TN-based authorisation systems. The first vulnerability issue is that there is no approach at present for the automatic detection of PCD (detailed in section 6.2). Li et al. (2003) have proposed an approach called OSBE (see section 3.3.1.2), which partially resolves this problem, but the approach works on the basis that PCD has been identified. There is however no current approach to enable two entities to automatically detect the occurrence of PCD. The second vulnerability is that there is no approach at present, which enables an honest Service Requester/Service Provider to defend against Repetitive Credential Request Attacks (detailed in section 6.2 and referred to as RCRA hereafter) potentially causing a DoS impact on the Web Service. If an honest Service Requester/Service Provider is communicating with a malicious unknown Service Provider/Service Requester, the malicious entity may continue sending requests, which impact on the other entity's resources.

As reviewed in Chapter 3, TN is an alternative to ABAC-based authorisation approach for two unknown systems to make access control decisions, so any flaws existing within TN may cause it to be less useful to provide security. Therefore, it is important to explore relevant solutions for addressing the two innate vulnerability issues to improve TN as soon as possible.

This chapter aims to identify possible solutions for the two identified vulnerability issues within TN. To enable a reader to have an insight into the two vulnerability issues with ease, two new case scenarios are presented to illustrate how the two vulnerability issues may occur within the TN process.

A conceptual solution design is proposed along with detailed clarification of its realisation by means of a specific technique in practice. After the description of the solution, relevant evaluation is discussed for demonstrating the effectiveness of this proposed solution. The next section starts with the description of the two case scenarios.

## 6.2 Innate Vulnerability Issues in TN

Two case scenarios in the context of Web Services are presented in this section. Case scenario 1 aims to illustrate how PCD may occur within a TN process between two honest unknown entities, and where the relevant vulnerability issue remains. Case scenario 2 aims to illustrate how an honest Service Provider/Service Requester may suffer RCRA from an unknown malicious Service Requester/Service Provider, which keeps repeatedly requesting the same credentials an undetermined number of times. As both vulnerability issues exist only in policy-exchanged strategies and the parsimonious strategy has been assessed as the most typical representation of this category (see section 3.3.1.2), the case studies below will use this strategy.

Without loss of generality, it is assumed that each resource, credential, policy set and policy will be assigned a unique id during the process. Symbols are used to represent the detailed information exchanged between them. More specifically, the meaning of symbols are explained as follows:

- A Resource is denoted as R.
- A Credential is denoted as C, and $C_i$ is different from $C_j$, $0<i<j<n$, where n is a finite natural number.
- A credential may contain a number of attributes. Each attribute name is denoted as $AttName_i$, and each attribute value is denoted as $AttValue_i$, $0<i<n$, where n is a finite natural number.
- According to the XACML policy language designed by Parducci and Locakhart (2010), a policy message can contain a policy set (denoted as $PS_i$, $0<i<n$, where n is a finite natural number), which in turn can contain multiple policies, which in turn can contain multiple rules. Each policy is denoted as P, and $P_i$ is different from $P_j$, $0<i<j<n$, where n is a finite natural number.

For clarity, all of the identifications assigned to policies and credentials used in WSA on behalf of Alice are odd numbers (e.g. $P_1$, $P_3$, $C_1$, $C_3$), whereas all of the identifications assigned to policies and credentials used in WSB on behalf of Bob are even numbers (e.g. $P_2$, $P_4$, $C_2$, $C_4$).

**Description of Case Scenario 1:** There are two Web Services A and B, which are unknown to each other (referred to WSA and WSB respectively). Alice is a user of WSA, and Bob is a user of WSB. Alice intends to access a resource R from Bob. As Bob treats R as a sensitive resource, he declares a policy $P_2$ protecting R. $P_2$ has a rule requiring a credential $C_1$ containing $AttName_1$=$AttValue_1$ to unlock it. Alice has the credential $C_1$, but she also treats $C_1$ as sensitive. Therefore, she declares a policy $P_1$ protecting its disclosure. $P_1$ requires a credential $C_2$ containing $AttName_2$=$AttValue_2$. Bob has the credential $C_2$, but he also treats this credential as sensitive. Therefore, he also sets $P_2$ to protect the disclosure of $C_2$.

When Alice uses WSA sending a request to WSB for accessing Bob's resource R, the communication process of TN is shown as follows:

**Step one:** WSA sends a request to WSB to inform that Alice wants to access Bob's resource R.

**Step two:** WSB discovers that Bob has declared the policy $P_2$ for protecting the disclosure of R, so it sets $P_2$ in $PS_2$ and sends the $PS_2$ to WSA.

**Step three:** After WSA analyses $P_2$ in $PS_2$, it knows that Alice's $C_1$ can fulfil $P_2$. However, as Alice has declared the policy $P_1$ for protecting the disclosure of $C_1$, it then sends $PS_1$ containing the $P_1$ to WSB.

**Step four:** Upon analysing $P_1$ in $PS_1$, WSB identifies that Bob's $C_2$ can meet the $P_1$. Unfortunately, as $C_2$ is also protected by $P_2$, $C_2$ cannot be disclosed. Therefore, $P_2$ should be contained in a $PS_4$, which will be sent out to WSA again. This phenomenon is called PCD as identified by Li et al. (2003).

At step four, it can be identified that PCD has occurred and WSB should use some approaches (e.g. OSBE) to stop resending $PS_4$ containing $P_2$ to WSA. Unfortunately, as there are no existing approaches that can help WSB automatically identify the occurrence of PCD, WSB may continue sending $P_2$ contained in $PS_4$ to WSA again.

**Description of Case Scenario 2:** There are two Web Services A and B, which are unknown to each other (referred to WSA and WSB respectively). WSA is an honest Web Service and WSB is a malicious Web Service. Alice is a user of WSA, who wants to access a resource R located in WSB held by a malicious owner Bob. Bob declares different policies $P_i$, $i>=1$, protecting this resource R, but all of these policies require a different set of attribute information within a credential $C_1$ containing m attribute information, $1<m<n$, where n is a finite natural number. Alice possesses $C_1$.

When Alice uses WSA sending a request to access Bob's resource R, the communication process of TN is shown as follows:

**Step one:** WSA sends a request to WSB to inform that Alice wants to access Bob's resource R.

**Step two:** WSB sends WSA $PS_1$ containing $P_1$ requesting $C_1$ containing $AttName_1=AttValue_1$.

**Step three:** After WSA analyses $P_1$, it submits Alice's $C_1$, as $C_1$ is not sensitive.

**Step four:** WSB sends WSA $PS_2$ containing $P_2$ requesting $C_1$ containing $AttName_2=AttValue_2$.

According to $P_2$ in $PS_2$ at step four, it can be identified that step five will be the same as step three, where WSA will send $C_1$ to WSB again. This kind of attack is explored in the protocol verification presented in Chapter 5. In other words, it has never been identified in state-of-the-art TN-based authorisation systems. So a term is given in this Thesis to refer to this kind of attacks as Repetitive Credential Request Attacks (RCRA). If WSB keeps sending its policies $P_i$ requesting different attribute information contained in $C_1$, $i>2$, where i is a natural number, WSA will keep sending $C_1$ to WSB, which eventually forms long-term or infinite communication.

## 6.3 A Proposed Solution Design

### 6.3.1 Discussion of one potential solution

In case scenario 2, the reason that WSA will suffer RCRA is due to the fact that there is no existing approach enabling WSA to detect that all $P_i$ requests aim to obtain the same local credential(s). A potential solution might be to enable WSA to remember the content of each $P_i$ disclosed by WSB so that it could stop communication, if the same content within the remote policy had been identified again. This solution could avoid infinite communication, but could not avoid long-term communication attack. For instance, a $P_3$ may be designed to request $C_1$ containing $AttName_1=AttValue_1$ and $AttName_2=AttValue_2$ and a $P_4$ may be designed to request the $C_1$ containing $AttName_1=AttValue_1$ and $AttName_3=AttValue_3$. According to the rule of combination, there can be $2^m-1$ kinds of policies requesting the same $C_1$, where m is the number of the attribute information within the $C_1$. Furthermore, in case scenario 2, it is assumed that each entity only possesses one credential. However, in reality, there can be a multitude of credentials owned by one entity. Thus, it is possible that longer-term-based RCRA may occur. From this perspective, this solution is not ideal.

### 6.3.2 Conceptual idea of a proposed solution

In order to address the two vulnerability issues, the core conceptual idea of the proposed solution is to enable an honest entity to remember the local information status in relation to policies that have been sent out and that of the requested local credentials. In terms of the context of the first vulnerability issue, it normally occurs between two honest unknown entities. To address the first vulnerability issue, as long as an entity detects that there exists one local policy that needs to be sent out to the counterpart during TN for the second time, a decision will be made that the occurrence of PCD has been detected. With respect to the context of the second vulnerability issue, it normally occurs between an honest entity and a malicious entity. To resolve the second vulnerability issue, as long as an entity detects that there exists one local credential that the number of times for requesting the credential is greater than a maximum value predefined in the local database, a decision will be made that the occurrence of RCRA has been detected.

The main benefit of this conceptual solution is that clues used by the honest entity to detect the occurrence of either two vulnerability issues are completely collected from the information stored in its local database; therefore, the reliability and veracity of the clues can be ensured. In other words, detection of the occurrence of either one of the two vulnerability issues does not rely on the information provided by the counterpart at all. Therefore, this solution is useful regardless of whether the counterpart is maliciously designed or not.

### 6.3.3 Realisation technique

To enable the realisation of the proposed conceptual solution, technically, more than one method can be used, as long as the methods can support the addition, modification and removal of data. Within this research, the selected specific technique to deliver the functionality is through relational database technology. The reason for selecting this technique is due to the power of Structured Query Language (SQL) used within the relational database technology.

Following the core idea of the conceptual solution – remembrance of local information status, this database is designed to automatically detect the occurrence of the first vulnerability issue by enabling an entity to check whether information in relation to each local policy has been stored in a local table. For detecting the occurrence of the second vulnerability issue, it allows an honest entity to remember the number of times that local credentials have been requested rather than to remember the content of each remote policy.

This database is designed by taking into consideration an easy integration into the improved protocol and the majority of existing TN-based authorisation systems such as TrustBuilder2 (Lee, Winslett and Perano, 2009). As a TN-based authorisation system should have its own policy compliance checker as a core component (Seamons et al., 2002a) for comparing local/remote credentials against remote/local policies in order to make a decision on whether or not any local policies or local credentials should be sent out. Before sending out local policies or local credentials, a request should be sent to this database for checking the local copy of the hitherto conducted process of TN including what policies and credentials have been sent out. A decision will be made only after the entity has checked out the local information within the database (see figure 6.1).

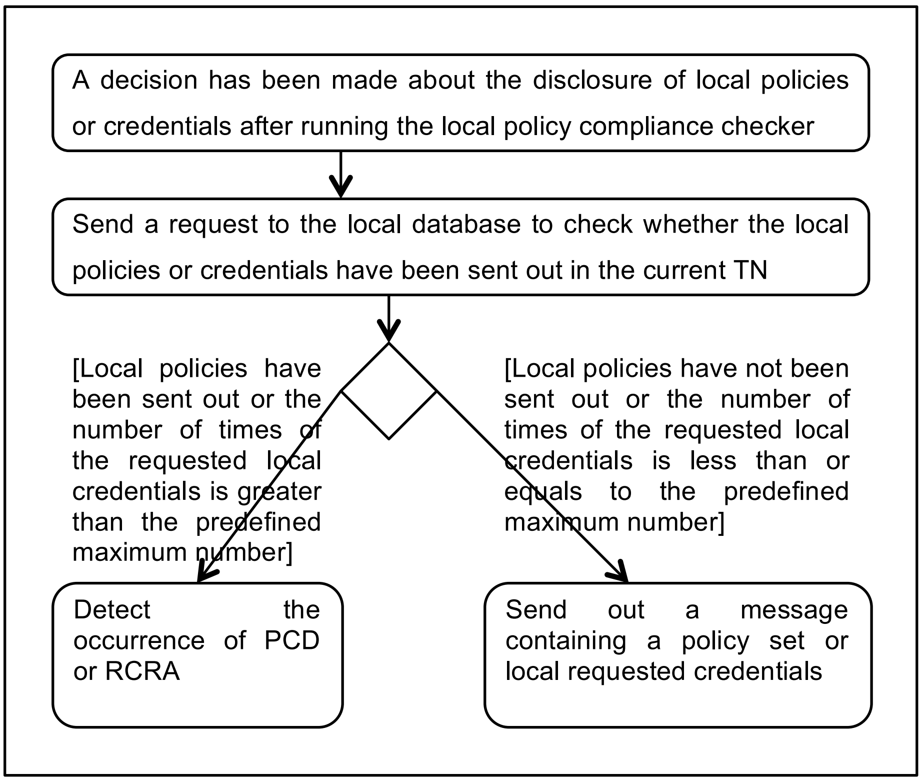

A decision has been made about the disclosure of local policies or credentials after running the local policy compliance checker
Send a request to the local database to check whether the local policies or credentials have been sent out in the current TN
[Local policies have been sent out or the number of times of the requested local credentials is greater than the predefined maximum number]
[Local policies have not been sent out or the number of times of the requested local credentials is less than or equals to the predefined maximum number]
Detect the occurrence of PCD or RCRA
Send out a message containing a policy set or local requested credentials

Figure 6.1. The position of the database within the process of TN

Within this database design, there are only two tables named "Local Policy" and "Local Credential" respectively, which are not relevant to each other. The "Local Policy" table is used to enable an honest Web Service to detect the occurrence of the first vulnerability issue, whereas the "Local Credential" table is used to aid an honest Web Service in detecting the occurrence of the second vulnerability issue. It is assumed that each local policy and local credential stored in a Web Service will be assigned a unique id. This assumption should be valid following the current WS specifications such as XACML (Parducci and Lockhart, 2010), WS-Policy (Vedamuthu et al., 2007), WS-SecurityPolicy (Lawrence and Kaler, 2009c) and SAML (Philpott et al., 2009).

Regarding the use of both tables, at the start of TN, both of them should be empty. Relevant data will be only added to this table when TN is in the process. Whenever TN finishes, each of the two tables should be emptied again regardless of the TN result (i.e. successful or failed). It should be noted that the two tables are not designed by taking into consideration other functionalities (e.g. logging). For scenarios where repeated malicious requests may be sent from the same malicious entity, a back up of both tables may be needed, where the stored information can be used as logging so that an honest entity is able to defend against such attacks.

The detailed description of each table are clarified as follows:

Table 1: In the “Local Credential” table, the attribute LCID is used as an identifier and is the primary key of this table. There are two other attributes called as NumberOfTimesOfBeingRequested (NOTOBR) and PredefinedMaximumNumberOfTimesOfBeingRequested (PMNOTOBR). The NOTOBR attribute is to count the number of times that a specific local credential has been requested by the counterpart in TN. Whenever a local credential is to be sent out upon being requested, the value of this attribute should be updated by adding one to its current value. The PMNOTOBR attribute is a predefined maximum number of times that a specific local credential has been requested. Its value is a fixed number, which cannot be changed. The value of this attribute is used to compare against the value of the NOTOBR attribute to help an entity decide whether a specific local credential should be disclosed.

Table 2: In the “Local Policy” table, the attribute LPID is used as an identifier of each local policy and is the primary key of this table.

After presenting the database design, pseudo code of an algorithm is presented in figure 6.2 below to illustrate how an honest entity can use the two tables to detect the two vulnerability issues.

```
PCD.detected=false; //judge whether PCD has been detected
RCRA.detected=false; //judge whether RCRA has been detected
if (a decision of the policy compliance checker is made to send
out policies (Pi) within a policy set (PSj)) {
 for each Pi{
  if(Select data from Table “Local Policy” where LPID=“Pi” can
   be found){
    PCD.detected=true;// if Pi has been stored in Table “Local
                //Policy”, it means that Pi has been sent out
    break;
  }else{
    add data in table “Local Policy” where LPID=“Pi”;
  }
 }
 if(PCD.detected.equals(false)){
  send out a PSi containing the Pi;
 else{
  detect the occurrence of PCD;
  Use any possible solution (e.g. OSBE);
 }
}else if(a decision of the policy compliance checker is made to
  send out local credentials (Ci) within a credential Set
  (CSj)) {
    if(Select data from Table "Local Credential"
      where LCID="Ci" can be found){
      for each Ci {
       if(Ci.NOTOBR<Ci.PMNOTOBR){
         update Ci.NOTOBR by increasing 1;
       }else{
         RCRA.detected=true;
         break;
       }
    }else{
      add data in table "Local Credential" where LPID="Ci",
  NOTOBR="1", PMNOTOBR="a predefined value";
    }
    if (RCRA.detected.equals(false)){
     send out the Ci in a CSj;
    }else{
     detect the occurrence of RCRA;
     send out a last message and stop TN; empty Tables;
    }
}
```

Figure 6.2. An algorithm for addressing the two vulnerability issues

One point that needs to be highlighted is that the functionality of emptying the two tables is not presented in the algorithm, when PCD is detected. As OSBE might be used after PCD is detected, TN could continue rather than stop. Nevertheless, the two tables still need to be emptied when TN finishes.

Upon clarifying the proposed solution design, the next section presents detailed description of evaluation result analysis of the solution design.

## 6.4 Evaluation Result Analysis

The case studies can be used to evaluate the effectiveness of the conceptual solution design through the realisation of the relational database design as well as the algorithm presented in figure 6.2. As evaluation of two case studies was carried out strictly following the process of the algorithm, a description of the evaluation process is not stated. If a reader is interested in the evaluation process, detailed information is referred to Appendix B. This section directly presents an evaluation result analysis.

Through the evaluation result analysis, the effectiveness of this proposed solution for the reviewed policy-exchanged strategies (see section 3.3.1.2) are concluded in table 6.1. The two vulnerability issues are used as row titles, and the names of the five policy-exchanged strategies are used as column titles in this table. The “tick” symbol means effective, whereas the “cross” symbol means ineffective.

| | **Parsimonious** | **PRUNES** | **DFANS** | **Adaptive** | **SRNS** |
|---|---|---|---|---|---|
| **Issue 1** | ✔ | ✖ | ✖ | ✔ | Uncertain |
| **Issue 2** | ✔ | ✖ | ✔ | ✔ | ✔ |

Table 6.1. Conclusion of the effectiveness of the database design for policy-exchanged strategies

The solution is effective for addressing the two vulnerability issues, when either the parsimonious strategy or the adaptive strategy is used. The reason for its effectiveness in resolving the first vulnerability issue – automatic detection of the occurrence of PCD for the two strategies – is due to the similar characteristic owned by these strategies. The similar characteristic is: whenever a local sensitive resource is requested, each local relevant policy protecting the sensitive resource will be sent out in an outgoing policy message immediately in the next round. Therefore, with the use of this proposed solution, it is convenient for an entity to check whether any local policies have been transmitted.

In terms of the assessment of the proposed solution for addressing the second vulnerability issue – defending against RCRA, it is very effective when any one of the policy-exchanged strategies is used except the PRUNES. The reason for its effectiveness is that these policy-exchanged strategies share one characteristic: whenever a local non-sensitive credential is requested, the honest entity will transmit it to the counterpart. The proposed solution is designed to enable an honest entity to record and analyse the local status of the requested credentials. Discovery of the occurrence of potential RCRA is therefore based on the local record rather than based on the memory of the same remote policies being disclosed by the counterpart. As long as the system of the honest entity is correctly designed (developers with no malicious intent can ensure this design), the occurrence of RCRA can always be detected regardless of whether the counterpart is maliciously designed or not.

In terms of the effectiveness of the proposed solution for the DFANS, it is an ineffective solution for addressing the first vulnerability issue – detection of the occurrence of PCD. The reason for its ineffectiveness is due to the nature of this strategy. With the use of this strategy, whenever there is a disclosure of a policy message, an entity will only disclose a request for one required credential (as mentioned earlier and in section 3.3.1.2, a policy can have multiple rules, and each rule can require different combinations of multiple credentials by using the logic symbols such as AND/OR) rather than the whole policy. As the proposed solution uses a "Local Policy" table to store the data for a whole policy rather than each atomic element (requesting one credential, see section 3.3.1.2) of the policy, the entity treats the action of sending out different atomic elements of the policy in different rounds as the action of sending out the same policy multiple times. Therefore, when applying this solution to the DFANS, detecting potential PCD is not correct. What is even worse is that potential successful TN without the occurrence of PCD will be misjudged as an occurrence of PCD.

When applying the database design for the PRUNES, it is completely ineffective for PCD and RCRA. Regarding PCD, the reason for its ineffectiveness is due to the fact that the characteristic of its disclosure of a policy message is the same as that of using the DFANS. Regarding RCRA, PRUNES is not designed to disclose each local credential shortly after discovering that the local credential can fulfil the received policy. The credential-exchange phase (see section 3.3.1.2) designed in the PRUNES will only occur, when the WSB as a service provider discloses the information that the resource R has been unlocked at the end of the policy-exchange phase. Unfortunately, in scenario 2, WSB is a malicious Web Service, so it does not provide such information to WSA during the policy-exchange phase. The proposed solution is designed as an addition to the result of a compliance checker, which decides to send out local credentials. Without performing the action of sending out a local credential, the proposed solution cannot even be triggered. In SRNS (Liu et al., 2013), policy disclosure is not that clearly specified; therefore "uncertain" is stipulated in the table.

Upon clarifying the benefits and limitations of the solution design, the next section discusses the impact of the proposed solution.

## 6.5 Impact of the Proposed Solution

Through protocol verification tests in Chapter 5, the proposed improved TN protocol has been verified to be incomplete, as two vulnerability issues – PCD and RCRA –

still exist within policy-exchanged strategies without any solutions. There is therefore a need to address them as soon as possible.

This chapter proposes a conceptual solution design based on the idea of "remembrance of local information status" through the realisation of relational database technology aiming to address the two vulnerability issues, thereby filling the gaps within the relevant field. Through the evaluation tests, the database design has been demonstrated to be effective to address the two vulnerability issues for the parsimonious strategy, adaptive strategy, and to resolve the second vulnerability issue for DFANS and SRNS. Its ineffectiveness for PRUNES and partial ineffectiveness for the DFANS is due to the unique nature of the two strategies, wherein the way an entity sends out policies and credentials by using PRUNES and DFANS is different from that of the other three strategies. Nevertheless, as the reason for the ineffectiveness of the solution design in these scenarios has been identified, it should be useful as guidance for exploring potential solutions. In addition, through the critical review of the TN strategies in section 3.3.1, the analysis of the inappropriate use of PRUNES for TN is clarified. This analysis implies that PRUNES is not suitable enough to be used within the context of TN. DFANS is also not superior in comparison with the parsimonious strategy. Therefore, taking into consideration the widespread use of the parsimonious strategy in practice, the solution design proposed in this chapter should be helpful for developers, who are intending to implement TN along with the parsimonious strategy within their systems.

This Thesis strongly recommends the use of the parsimonious strategy in the improved TN protocol, and the proposed solution in this chapter can effectively address the two vulnerability issues, when the parsimonious strategy is used within the protocol. Therefore, this solution can be treated as a complementary approach to ensure that the completeness of the protocol as tested in Chapter 5 can hold, as the infinite iterations caused by the two vulnerability issues can be broken with the use of this solution.

In addition, with respect to the nature of RCRA that an honest entity always replies to seemingly different normal requests without checking whether each request in essence is the same, RCRA can be considered as a variation of Denial of Services

(DoS) attacks. As the proposed solution is effective to the majority of the policy-exchanged strategies, it might also be applied to defend against other variations of DoS attacks.

## 6.6 Chapter Summary

In this Chapter, a conceptual solution design is proposed to address the vulnerability issues identified in the protocol completeness test shown in Chapter 5, and its effectiveness has been evaluated in case studies. This solution can be added to the improved TN protocol, so the completeness of the protocol can be proved. The next chapter details a protocol evaluation for evaluating the effectiveness of the interoperability-solution design, which in turn can prove the protocol correctness.

# Chapter 7. Protocol Evaluation

## 7.1 Introduction

Chapter 5 has detailed the protocol verification for the completeness of the improved TN protocol. Through the verification process, two innate vulnerability issues have been identified in the proposed protocol, wherein the completeness cannot be ensured. To address the two vulnerability issues, Chapter 6 has proposed a conceptual solution design based on the idea of "remembrance of local information status" through the realisation of the relational database technology. This solution can complement the proof of the completeness of the proposed TN protocol (see section 4.4). This chapter aims to evaluate the proposed interoperability-solution design (see table 4.1) to assess its effectiveness for addressing the second research problem. In addition, a protocol correctness test is also included within this evaluation as a sub part.

To evaluate the effectiveness of the proposed interoperability-solution design and the correctness of the protocol, an appropriate research method should be selected to carry out the evaluation. Therefore, the rationale for selecting the most appropriate research method is clarified. In particular, a discussion of the benefits and limitations of the selected research method is presented. In addition, a discussion around the inappropriateness of other potential usable research methods is also given. Once the evaluation research method has been decided, a discussion is provided regarding the evaluation method procedure. Following this discussion, a data collection method is used for collecting the relevant data for conducting data analysis. Based on the data analysis results, the answers for the effectiveness of the protocol are described in detail, which in turn answers the effectiveness of the interoperability-solution design. In addition, a further discussion of the impact of the improved TN protocol is also provided.

## 7.2 Evaluation Research Method

The research method selected within this Thesis to enable evaluation of the degree of interoperability provided by the proposed solution is one of experimental case study. The use of case study evaluation is commonly used in different fields such as the social sciences (Neale, Thapa and Boyce, 2006; Baxter and Jack, 2008). Robson

(2002) states “The flexibility in design and execution of the case study, together with the fact that most evaluations are concerned with the effectiveness and appropriateness of an innovation or programme in a specific setting …, make the case study strategy appropriate for many evaluations”. Yin (2013) suggests that a case study is usually used when (1) there is a need to answer “how” and “why” questions, (2) the investigated target is “a contemporary set of events” and (3) a researcher has little or no control over the investigated phenomenon. Yin (2013) also states that other research methods (e.g. surveys, interviews, documents) can be used within case studies for achieving data collection. Qualitative and quantitative methods can also be used within case studies for accomplishing data analysis. These features enable case studies to be a comprehensive research method. Another unique feature of using case studies is that researchers can develop a deeper insight into the investigated phenomenon, whereas other research methods struggle to achieve this goal (Thapa and Boyce, 2006).

In the field of Information Systems (referred to IS hereafter), a case study approach has been accepted as a main research method used by researchers (Benbasat, Goldstein and Mead, 1987; Lee, 1989, Klein and Myers, 1999; Runeson and Host, 2009). For instance, Benbasat, Goldstein and Mead (1987) provide a reason for using the case study as a reasonable research method, “a case approach is an appropriate way to research an area in which few previous studies have been carried out”. They also argue that two categories of case study namely “application descriptions” and “action research” respectively used in the field of IS should be excluded from case study. They give the reasons as follows.

- With the use of the “application descriptions” method, researchers mainly focus on the implementation of a specific system rather than on conducting research. Reasons for the inappropriateness of implementation used within this evaluation are detailed in section 7.3.4.
- Following the process of “action research”, researchers are proposed to become participants rather than observers.

Referring back to the second research question (see section 1.2.2), it is raised in the way of “how”. By observing the context of the second research problem (see section

1.2.1), integrating different authorisation systems into one Web Service for achieving authorisation is a ubiquitous phenomenon.

Limitations of using a case study approach exist, for example as stated by Neale, Thapa and Boyce (2006) (1) they can be lengthy (presenting the case in a narrative form), (2) lacking rigor and (3) can also not be generalizable (limiting scope). These limitations cause the case study method to fall in the category of qualitative research, which cannot provide definitive scientific results due to the lack of systematic data collection. In addition, criteria can be hard to set for assessing and measuring the evaluation results from the data analysis with the use of case study. However, as mentioned earlier, Yin (2013) argues that a case study can include quantitative methods for data collection and data analysis; therefore, a case study can include a mixed methodological response and as such should not simply be categorised into qualitative research. In addition, Yin (2013) also argues that features such as case studies lacking rigor and being non-generalizable can also occur in other research methods such as experiments.

Given the analysis above, it can be identified that the majority of the features of this research (e.g. the way the second research question asked, it is a contemporary phenomenon) can match the rationale for using case studies mentioned earlier. In addition, taking into consideration the benefits and limitations of case studies, a decision was made that the use of case studies should be the most appropriate research method for conducting evaluation.

## 7.3 Use of Case Studies

### 7.3.1 A general procedure of conducting case studies

Yin (2013) introduces a concept of "case study protocol", which aims to help researchers to design a general procedure before conducting case study research so as to increase the reliability of the research. Following this concept, Runeson and Host (2009) propose a simplified procedure of conducting case studies, which is similar to that introduced by Kitchenham et al. (2002) as follows:

"(1) Case study design: objectives are defined and the case study is planned.

(2) Preparation for data collection: procedures and protocols for data collection are defined.
(3) Collecting evidence: execution with data collection on the studied case.
(4) Analysis of collected data
(5) Reporting."

All of the above researchers suggest that conducting case study-based research should follow the path of a predefined procedure so that a researcher is able to identify the proper answers for the original raised research questions. Therefore, this procedure is used in this evaluation as general guidance.

### 7.3.2 Types of case studies

In terms of the types of case studies, different researchers provide their recommended types from different perspectives. Amongst them, types introduced by Yin (2013) and Stake (1995) are widely accepted by researchers nowadays (Baxter and Jack, 2008; Runeson and Host, 2009). Yin (2013) introduces three types: explanatory, exploratory and descriptive. Explanatory case studies are mainly used to identify the causal links between causes and effects. Use of exploratory case studies can be used to seek to define research questions, as stated by Stake (1995) " [Exploratory designs] are often a prelude to additional research efforts and involve fieldwork and information collection prior to the definition of a research question". Descriptive case studies are used to describe contemporary phenomenon. Apart from the three categories above, Yin (2013) also recommends the use of single-case designs and multiple-case designs, and points out their specific advantages and disadvantages. Taking into consideration the probability of the impact of detailed units of case studies, both single-case study and multiple-case studies can then be classified into embedded case studies and holistic case studies. Categories of case studies are presented in a matrix as shown in figure 7.1. With the application of this matrix, any one of the explanatory, exploratory or descriptive case studies can be used in any one of the four types as shown in the matrix.

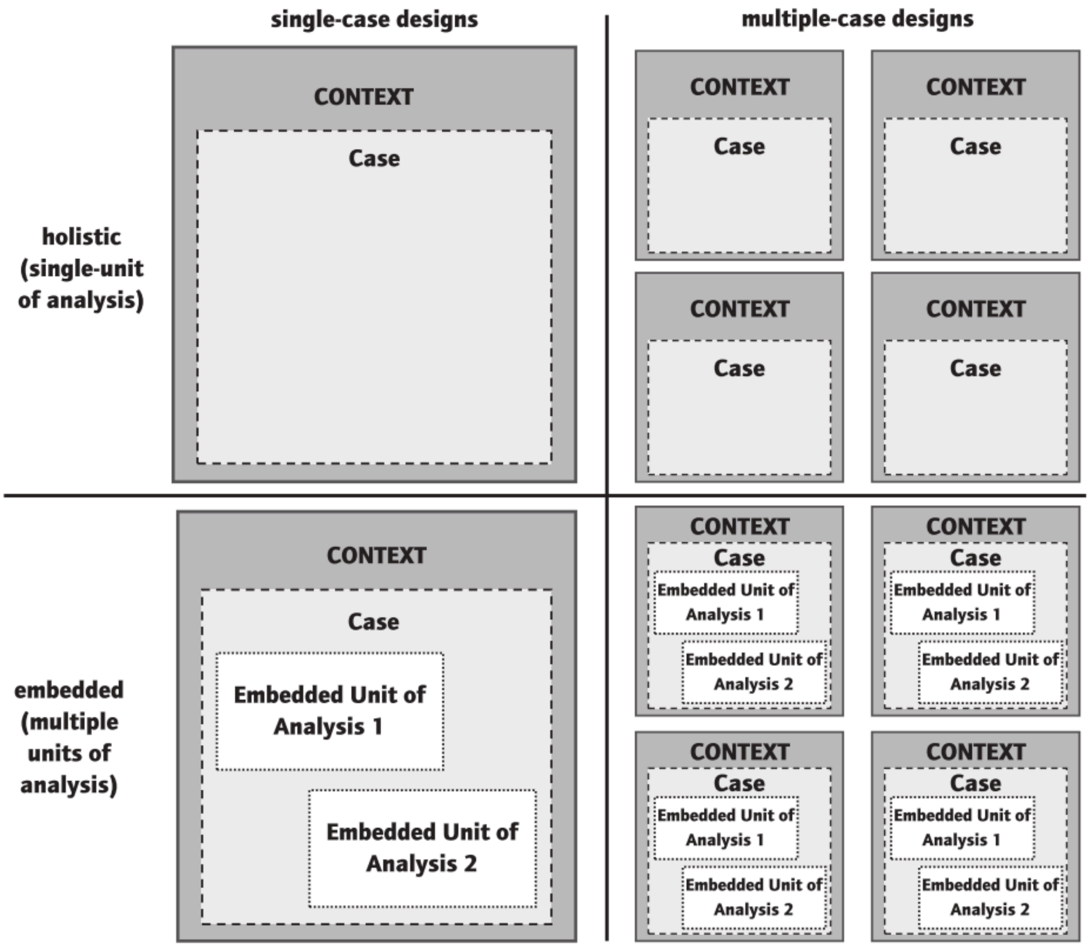

single-case designs
multiple-case designs
holistic (single-unit of analysis)
embedded (multiple units of analysis)
CONTEXT
Case
Embedded Unit of Analysis 1
Embedded Unit of Analysis 2

Figure 7.1. A category of case designs (Yin, 2013)

In contrast to Yin's types, Stake (1995) also introduces three categories of case studies, which are intrinsic, instrumental or collective. Intrinsic case studies are selected, if researchers concentrate more on the details of the case studies rather than the general context. Instrumental case studies are used, if researchers aim to gain an insight into the issues occurring in the general context of case studies for understanding or refining a theory. The notion of collective case studies is similar to that of multiple case studies as described by Yin (2013), in which a multitude of case studies are used.

Observing the characteristics of the types provided by Yin (2013) and Stake (1995), it can be identified that there are similarities between them as listed below.

- Holistic single case studies (Yin, 2013) are similar to instrumental case studies (Stake, 1995);

- Embedded single case studies (Yin, 2013) are similar to intrinsic case studies (Stake, 1995);
- Holistic multiple-case studies (Yin, 2013) are similar to collective case studies (Stake, 1995).

According to the nature of each type of case study mentioned above, it can be identified that the case scenario consisting of different circumstances as presented in section 3.4 for illustrating the first research problem belong to the category of being an embedded explorative single case study as mentioned by Yin (2013). As these explorative circumstances are only used to help the researcher seek research questions, they are not properly designed for evaluation. Following the guidance of the case study design mentioned above, case studies should be designed closely related to the raised research questions (see step 1 of the case study protocol shown in section 7.3.1).

Before conducting the case study design within the research, Yin (2013) states that a researcher should have decided to use a single-case design or a multiple-case design. To help a researcher make such a decision, Yin (2013) also provides propositional rationales for conducting single-case designs and multiple-case designs. In short, a decision of using a single-case study should be made, when a case is (1) critical, (2) extreme, (3) common, (4) revelatory and (5) longitudinal. However, critiques arise for using the single-case study, if other potential important conditions are overlooked; therefore, a theory concluded from single-case designs may not be able to provide generality. In such a circumstance, multiple-case designs should be used instead to enforce the robustness of a generalised theory.

In terms of the use of multiple-case designs, questions may also occur such as how many case studies should be used or what case studies should be used? To help researchers make decisions, researchers such as Yin (2013), Stake (1995) and Baxter and Jack (2008) propose a guideline that the use of multiple-case studies should be able to demonstrate the literal replication (similar results) or theoretical replication (contrast results). The decision of choosing the appropriate case studies requires a researcher to predict possible results before the multiple-case designs. The researcher's prediction is based on obtained knowledge through the literature review.

### 7.3.3 Construction of case studies

Observing the second research question in section 1.2.2, "authorisation systems" and "Web Services" are identified as two keywords for the construction of case studies. Through the literature review shown in Chapter 2 and Chapter 3, authorisation systems used within the current context of Web Services mainly include ABAC-based authorisation systems such as PERMIS, Akenti (see section 2.6). By integrating the improved TN protocol into authorisation systems within Web Services, communication between authorisation systems can be classified into three categories: (1) exchange between one ABAC-based authorisation system and one ABAC-based authorisation system (e.g. communication between PERMIS and Akenti), (2) exchange between one ABAC-based authorisation system and one TN-based authorisation system (e.g. communication between PERMIS and TrustBuilder2) and (3) exchange between one TN-based authorisation system and another TN-based authorisation system (e.g. communication between TrustBuilder2 and TrustServ).

The above features are only obtained from authorisation systems, not from Web Services as a general context for investigation. Therefore, there is a need to identify the necessary features of Web Services. As reviewed in Chapter 2, Schlager et al. (2006) propose an AAI for e-commerce by using Web Services technologies, which are widely used in further research (e.g. Erber, Schlager and Pernul, 2007; Schlager et al., 2007). Globus proposed by Foster (2006) is another widely accepted framework, which is a toolkit normally used for embedding authorisation systems within the context of Web Services. Within the AAI and Globus, SAML messages are utilised as protocol messages for achieving authorisation. Therefore, the use of SAML messages is one feature of authorisation systems in Web Services. The process for making an access control decision is defined within the XACML specification. This process has been widely acknowledged within Web Services, so the use of this process is another feature of authorisation systems in Web Services.

In order to help developers design real Web Services with ease, Daigneau and Robinson (2011) introduce three common API styles called "Remote Procedure Call

(RPC) API", "Message API" and "Resource API" respectively. Amongst them, the "Resource API" style is especially utilised, when a service requester needs to access a requested resource provided by a service provider. So the "Resource API" is the most appropriate API style to be used in scenarios where requests to access specific resources occur. Thus, the "Resource API" style is also treated as a feature within the context of Web Service authorisation systems.

A basic and common design pattern for communication between Web Services is called "Request/Response". As this design pattern is easy to be implemented, it is treated as the default pattern used within existing Web Services (Daigneau and Robinson, 2011).

As mentioned in Chapter 2, whenever a requested resource has been treated as sensitive, authorisation has to be involved to enable a service provider to make an access control decision on it. Therefore, to enable the authorisation to be used within the "Resource API" style, Daigneau and Robinson (2011) suggest the use of the "Request/Response" design pattern and that of the "Resource API" style together as a basic technique. Therefore, the use of them is also treated as a feature within the context of Web Service authorisation systems.

It should be noted that other features such as communication breakage, broker services might occur in all types of communication between two commutating systems in distributed systems environments. As they cannot be treated as the unique features existing in the communication between authorisation systems, these features are not included in the context of the case study design.

Taking into account the "common" characteristic (see reason 3 of the rationales for the single-case design) of the context of the research, the use of single-case designs is appropriate within this research. According to figure 7.1 as suggested by Yin (2013), the single-case designs category is twofold: holistic and embedded. Through the critical analysis as shown in Chapter 2 and Chapter 3, embedded units are discovered within authorisation systems including (1) multiple authorisation system types, (2) multiple strategy types, (3) multiple policy languages, (4) multiple credential languages, (5) predefined sensitivity of policies and (6) predefined sensitivity of credentials. Any change of each embedded unit may cause a case study to produce a different result. Hence, a decision was made to use embedded single-case designs for evaluation based on the above features.

Combining the features of authorisation systems and the relevant Web Service API style and design pattern, a general overview of embedded single-case designs is shown in figure 7.2 below.

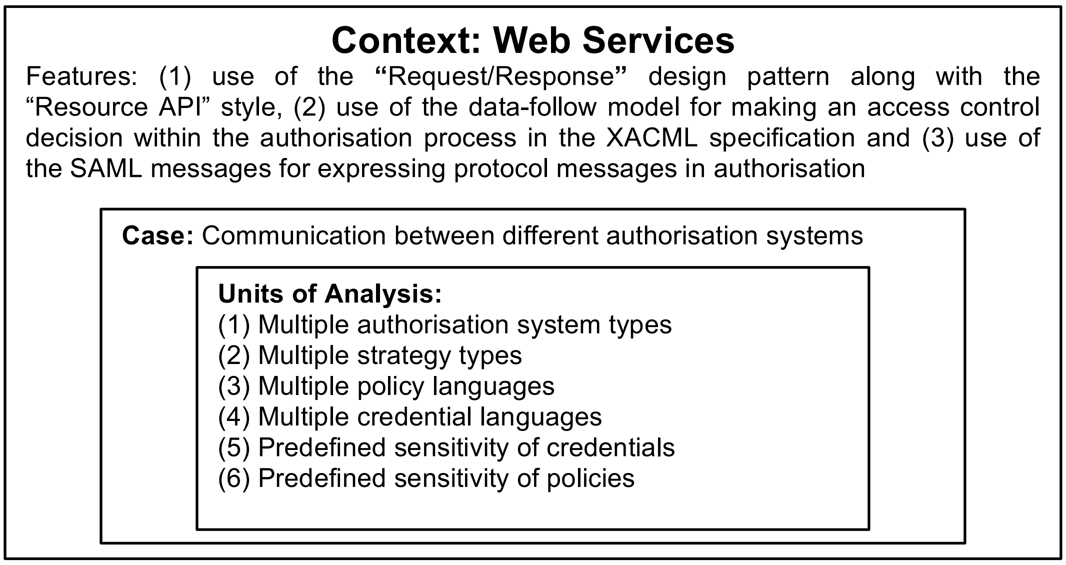

Context: Web Services
Features: (1) use of the "Request/Response" design pattern along with the "Resource API" style, (2) use of the data-follow model for making an access control decision within the authorisation process in the XACML specification and (3) use of the SAML messages for expressing protocol messages in authorisation
Case: Communication between different authorisation systems
Units of Analysis:
(1) Multiple authorisation system types
(2) Multiple strategy types
(3) Multiple policy languages
(4) Multiple credential languages
(5) Predefined sensitivity of credentials
(6) Predefined sensitivity of policies

Figure 7.2. A general overview of embedded single-case designs

### 7.3.4 Methods and process for data collection

In terms of the approach for the protocol application, a model-based testing of the protocol (Utting, and Legeard, 2006; Kull, 2009) is preferred to the implementation of the protocol along with real experiments. Model-based testing requires no need to perform a real experiment, and allows researchers to conduct an experiment through a mental experimental process for exploring the consequences of theory application. The use of model-based testing as an evaluation method has been applied in research such as Lu and Liu (2009). The reasons for using the model-based testing of the protocol are explained as follows.

- As protocol implementation should strictly follow the designed process of the theory of the protocol, a protocol process within a model-based testing environment and a real experiment should be the same, if the conditions set in both methods are the same.
- With the use of model-based testing, if any unpredicted logical design error occurs, it will be straightforward for the researcher to locate the error position within the

protocol. By contrast, this benefit might not be easily achieved through a practical experiment. For example, if there existed any unpredicted logical design error in a prototype system, the system might suddenly pause and print out technical error information (e.g. there are many in-built technical error information predefined in programming languages such as Java). In other words, the system could not explicitly indicate which part of the protocol the error comes from. This would cause the researcher to re-examine the theoretical process of the protocol to identify the error position.

- In contrast to a real experiment, with the use of model-based testing, technical obstacles in relation to implementation will not occur. It is straightforward for the researcher to apply the proposed protocol to each embedded case study to collect the relevant data from the descriptive process. Potential technical obstacles of the protocol implementation might not be discovered through this approach, which is one potential weakness of using model-based testing. Nevertheless, as these issues are not the major concerns within the evaluation of this research, they should not severely affect the evaluation result.
- In terms of a unique benefit of using a real experiment, it could prove the feasibility of the protocol implementation. However, as the majority of the components (e.g. compliance checkers, strategy components) designed within the proposed protocol had been implemented in the state-of-the-art TN-based authorisation systems, their feasibility was not an issue. The major difference between the protocol and the TN-based authorisation systems is the preparation stage, which is designed within the protocol, but is not supported by the TN-based authorisation systems. By observing the functionality designed in the preparation stage, there should be no technical issue for enabling two entities to agree on a common capability and a common strategy, as the keyword-based and semantic-based approaches (e.g. OWL language) are readily available in Web Services. From this perspective, the use of a real experiment may not show any superiority to the use of model-based testing in this evaluation.

As a decision has been made that model-based testing will be used within the embedded single-case study, model-based testing should also be the most appropriate approach used for data collection. In case study evaluation, triangulation is a commonly used term meaning “taking different angles towards the studied object and thus providing a broader picture” (Runeson and Host, 2009). The researcher did realise that the use of multiple sources of evidence was often used within the data collection phase of a research to avoid potential bias on a generalised theory based on the collected data. However, due to the unique characteristic of the selected approach used for the case study within this research, other methods for data collection such as surveys, implementation-based experiments or historical analysis were considered to be not appropriate. Reasons for the inappropriateness of implementation-based experiments have been given above. Questionnaire-based surveys and interviews were also not appropriate to be used, as the main participants within this research are the authorisation systems rather than human beings. Historical analysis, as another data collection method, normally aims to establish data from previous data subsets. The investigated phenomenon in this research is communication between authorisation systems within Web Services, which is a relatively new phenomenon since the year around 2000; thus, the historical analysis method was also not appropriate to be used.

Having explained the reasons for selecting the case study method along with the use of model-based testing as the evaluation research methods and data collection methods respectively, the data collection process after applying the protocol within the case study for collecting the target data within the case study is presented as follows (see step 2 of the case study protocol shown in section 7.3.1).

**Step one:** Compare the improved TN protocol against the current protocols used in the two authorisation systems as a service requester and a service provider respectively in each embedded sub case. Differences between the improved TN protocol and the compared protocol used in each existing authorisation system (e.g. ABAC-based authorisation systems such as PERMIS, Akenti or TN-based authorisation systems such as TrustBuilder2, Trust-X) as participating systems will be generated. More precisely, any elements (e.g. protocol messages, components providing specific functionalities) that are defined within the improved TN protocol, but cannot be found in the current protocol used in the two participating authorisation systems will be listed (see tables 7.2 to 7.5). These elements are to be embedded in the protocols used in the two participating authorisation systems within each embedded sub case. Then the differences generated from the comparison between the improved TN protocol and the protocol used in the two participating authorisation systems is treated as the first data source.

**Step two:** Let two authorisation systems use the improved TN protocol. The detailed authorisation process will depend on the conditions pre-set in each embedded sub case. The detailed descriptive process of authorisation in each embedded sub case is treated as the second data source.

**Step three:** Whenever authorisation in each embedded sub case finishes without any interoperability issues (see the criterion for interoperability identified in section 2.3), that is, the reason causing authorisation communication to cease is not due to an inability to process received messages. Such an authorisation result is treated as the third data source regardless of its result type (i.e. success or failure).

### 7.3.5 Model-based testing of protocol application within embedded single-case study for data collection

In the case study design, several embedded sub cases are used, wherein each of the units of analysis are different. In order to differentiate each embedded sub case, the term "circumstance" is used to substitute the term "embedded sub case" for simplicity.

Without the loss of generality, it is assumed that each resource, credential, policy set and policy will be assigned a unique id during the process. Each credential and policy language should have its unique name. In other words, given a language name for credentials or policies, the syntax and semantics of this language should be unique.

Symbols are used to represent the detailed information exchanged between them. More specifically, the meaning of symbols are explained as follows:

- A Resource is denoted as R.
- A Credential is denoted as C, and $C_i$ is different from $C_j$, $0<i<j<n$, where n is a finite natural number.
- A Credential Language is denoted as CL, and $CL_i$ is different from $CL_j$, $0<i<j<n$, where n is a finite natural number.
- A policy is denoted as P, and $P_i$ is different from $P_j$, $0<i<j<n$, where n is a finite natural number.

• A Policy Language is denoted as PL, and $PL_i$ is different from $PL_j$, $0<i<j<n$, where n is a finite natural number.

For clarity, all of the identifiers assigned to policies and credentials used in WSA on behalf of Alice are odd numbers (e.g. $P_1$, $P_3$, $C_1$, $C_3$), whereas all of the identifiers assigned to policies and credentials set in WSB on behalf of Bob are even numbers (e.g. $P_2$, $P_4$, $C_2$, $C_4$).

To clarify the expression of the possessed information of each entity in each circumstance, specific symbols used in Yu, Ma and Winslett (2000) are presented in figures. The semantics of these symbols are explained as follows:

• A symbol "R" denotes a sensitive resource possessed by the WSB as a Web Service Provider.

• A symbol "$C_i$" denotes a credentials possessed by a Web Service, where $C_i$ is different from $C_{i,}$ $0<i<j<n$, where n is a finite natural number.

• A symbol "$P_i$" denotes a policy possessed by a Web Service, where $P_i$ is different from $P_{j,}$ $0<i<j<n$, where n is a finite natural number.

• To detail the content of a specific policy, a symbol "$\leftarrow$" means a requirement. The left parts of this symbol are the protected sensitive resources (e.g. a sensitive requested resource R, sensitive credentials $C_i$, sensitive policies $P_i$). If more than one sensitive resource is set in the left part of this symbol, a parenThesis symbol "()" along with a comma symbol "," dividing each sensitive resource is used. The right parts of this symbol are required credentials. In addition, a symbol "$\wedge$" denotes the logical conjunction of required credentials, and a symbol "$\vee$" denotes the logical disjunction of required credentials. For instance, a policy $P_1$ represented as "$P_1$: $(C_{4,} C_5)\leftarrow(C_1\wedge C_2)\vee C_3$" means that to request a sensitive credential $C_4$ and a sensitive credential $C_5$ held by one Web Service, a counterpart needs to submit a combination of a credential $C_1$ and a credential $C_2$ or a credential $C_3$.

There are ten circumstances designed in this case study for data collection (see step 3 of the case study protocol shown in section 7.3.1). An overview of basic information of each circumstance is presented in table 7.1.

| Circumstances | Circumstance 1: | Circumstance 2: |
|---|---|---|
| Communicating Systems: | ABAC-ABAC | ABAC-ABAC |
| Conditions: | • No common language combinations can be supported by both entities.<br>• Both entities use the parsimonious strategy. | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• WSA can only support the parsimonious strategy, whereas WSB can only support the eager strategy. |
| Result: | Finishes in stage 1, as the capability interoperability issue has been identified. | Finishes in stage 1, as the strategic interoperability issue has been identified. |
| **Circumstances** | **Circumstance 3:** | **Circumstance 4:** |
| **Communicating Systems:** | ABAC-ABAC | ABAC-TN |
| **Conditions:** | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the parsimonious strategy.<br>• WSA holds a non-sensitive $C_1$. | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the parsimonious strategy.<br>• WSA holds a sensitive $C_1$. |
| **Result:** | A successful authorisation result finishes in stage 2 | A failed authorisation result finishes in stage 2, as required credentials cannot be submitted |
| **Circumstances** | **Circumstance 5:** | **Circumstance 6:** |
| **Communicating Systems:** | ABAC-TN | ABAC-TN |
| **Conditions:** | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the eager strategy.<br>• WSA holds a sensitive $C_1$, and WSB does not hold any credentials to unlock $C_1$. | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the parsimonious strategy.<br>• WSA holds a sensitive $C_1$ and a non-sensitive $C_3$, and WSB holds a non-sensitive $C_2$ that can unlock $C_1$.<br>• WSB declares a sensitive $P_2$. |
| **Result:** | Authorisation failure finishes in stage 2, as required credentials cannot be | Authorisation success finishes in stage 2. |

| | | |
|---|---|---|
| | submitted. | |
| **Circumstances** | **Circumstance 7:** | **Circumstance 8:** |
| **Communicating Systems:** | ABAC-TN | TN-TN |
| **Conditions:** | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the eager strategy.<br>• WSA holds non-sensitive $C_1$ and $C_3$, and WSB holds a non-sensitive $C_2$.<br>• WSB declares a sensitive $P_2$. | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the eager strategy.<br>• WSA holds a sensitive $C_1$ and non-sensitive $C_3$, and WSB holds a non-sensitive $C_2$.<br>• WSB declares a sensitive $P_2$. |
| **Result:** | Authorisation success finishes in stage 2. | Authorisation success finishes in stage 2. |
| **Circumstances** | **Circumstance 9:** | **Circumstance 10:** |
| **Communicating Systems:** | TN-TN | TN-TN |
| **Strategy:** | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the parsimonious strategy.<br>• WSA holds sensitive $C_1$ and $C_3$, WSB holds a non-sensitive $C_2$.<br>• WSB declares a sensitive $P_2$. | • Both entities can compare policies written in $PL_1$ against credentials written in $CL_1$.<br>• Both entities use the parsimonious strategy.<br>WSA holds sensitive $C_1$ and $C_3$, and a non-sensitive $C_7$ and $C_9$, and WSB holds a non-sensitive $C_2$, and a sensitive $C_4$.<br>WSB declares a sensitive $P_2$. |
| **Result:** | Authorisation failure finishes in stage 2, as required credentials cannot be submitted. | Authorisation success finishes in stage 2. |

Table 7.1. An overview of ten circumstances in the case scenario

For simplicity, the general background information of the case study is the same as the case scenario described in section 2.8. It should be noted that as there are similarities among the ten circumstances, so the same detailed process is only described in its first occurrence in the relevant circumstance.

<u>Circumstance 1:</u>

Alice wants to access Bob's resource R. Bob declares a $P_2$ requiring attribution information in a $C_1$ to protect R. Alice has a $C_1$.

The possessed information presented in symbols is shown in figure 7.3 below.

WSA (Alice)
Credentials: $C_1$
Policies: nil

WSB (Bob):
Resource: R
Credentials: nil
Policies: $P_2$:R$\leftarrow C_1$

Figure 7.3. Possessed information in circumstance 1

<u>*Units of Analysis:*</u>

- Authorisation system types: both WSA and WSB use ABAC-based authorisation systems (PERMIS and Akenti for instance)
- Strategy types: both WSA and WSB support the use of the parsimonious strategy
- Policy/Credential language: WSA can compare policies written in $PL_1$ against credentials written in $CL_1$. WSB can compare policies written in $PL_2$ against credentials written in $CL_2$.
- Sensitivity of credentials: nil
- Sensitivity of policies: nil

Before applying the proposed protocol to this circumstance, a comparison between PERMIS, Akenti and the improved TN protocol is needed. The detailed comparison is shown in table 7.2. There are four columns within this table. The first column presents the interoperability layers from 2 to 6 defined within the interoperability models (see tables 3.2 and 4.1). The second, third and fourth columns present the relevant content identified within the proposed TN protocol, the protocol of PERMIS and that of Akenti used in Web Services respectively as used in the AAI (see section 2.7.1).

| **Interoperability Layer** | **Improved TN Protocol** | **PERMIS in Web Services** | **Akenti in Web Services** |
|---|---|---|---|
| Strategy | (1) Preparation stage and (2) Negotiation Strategy Repository Component | Nil | Nil |
| Capability | (1) Preparation stage and (2) Multiple languages for expressing credentials and Policies | SAML messages only | SAML messages only |
| Functionality | Policy Compliance Checker | Following the XACML data-flow | Following the XACML data- |

| | | diagram | flow diagram |
|---|---|---|---|
| Syntax and Semantics | XML-based languages, SAML languages and XACML languages | SAML messages only | SAML messages only |

Table 7.2. Comparison between PERMIS, Akenti and the improved TN protocol

Observing table 7.2, it can be identified that except the policy compliance checker (the XACML data-flow diagram used in PERMIS and Akenti provides the functionality of a policy compliance checker) as a common functionality owned by three protocols, other interoperability layers supported in the improved TN protocol cannot be supported in PERMIS and Akenti. So assuming that these parts of the improved TN protocol are to be implemented within the two systems.

By applying the proposed protocol to circumstance 1, the communication process is shown as follows.

**Step 1:** WSA sends a <TNPrepareRequest> message to WSB for requesting an access to Bob's resource R, and a list of supported strategies and language combinations contained within the message to WSB. The message is shown in figure 7.4 below.

```
<TNPrepareRequest Resource="http://WSB/Bob/Resource/R"
RemoteResourceOwner="http://WSB/Bob">
  <StrategyList Number="1">
    <Strategy ID="1">parsimonious</Strategy>
  </StrategyList>
  <LanguageCombinations Number="1">
    <LanguageCombination ID="1">
      <PolicyLanguage>PL1</PolicyLanguage>
      <CredentialLanguage>CL1</CredentialLanguage>
    </LanguageCombination>
  </LanguageCombinations>
</TNPrepareRequest>
```

Figure 7.4. Message 1 in the preparation stage in circumstance 1

**Step 2:** Following the process of dealing with the <TNPrepareRequest> message as shown in figure 4.4, WSB first tries to find out whether the strategy name contained within this message is also supported in its own system. As the received strategy name is "parsimonious" and this name is also supported by its own system, it then tries to discover whether the language combinations contained within this message are also supported in its own system. Within the received message, only one language combination “$PL_1$ and $CL_1$” has been found. As WSB can only compare policies written in $PL_2$ against credentials written in $CL_2$, it cannot discover a common language combination. So it responds with a <TNPrepareResponse> message that TN cannot be used due to an internal interoperability issue of TN. The message is shown in figure 7.5 below.

```
<TNPrepareResponse TNCanBeUsed=”no”>
  <Fault>Language interoperability issue</Fault>
</TNPrepareResponse>
```

Figure 7.5. Message 2 in the preparation stage in circumstance 1

As the <TNPrepareResponse> message contains a fault message, according to the process dealing with this response message (see figure 4.5), WSA will cease the following communication with WSB.

Circumstance 2:

Alice wants to access Bob’s resource R. Bob declares a $P_2$ requiring attribute information in a $C_1$ to protect R. Alice has a $C_1$.

The possessed information presented in symbols is shown in figure 7.6 below.

| | |
|---|---|
| WSA (Alice)<br>Credentials: $C_1$<br>Policies: nil | WSB (Bob):<br>Resource: R<br>Credentials: nil<br>Policies: $P_2$:R$\leftarrow C_1$ |

Figure 7.6. Possessed information in circumstance 2

*Units of Analysis:*

- Authorisation system types: both WSA and WSB use ABAC-based authorisation systems (CAS and VOMS for instance)
- Strategy types: WSA only supports the parsimonious strategy and WSB only supports the eager strategy
- Policy/Credential language: both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$. WSA can also compare policies written in $PL_2$ against credentials written in $CL_2$.
- Sensitivity of credentials: nil
- Sensitivity of policies: nil

As the protocols of CAS and VOMS used in Web Services are the same as that used in PERMIS and Akenti (see the AAI shown in figure 2.7.1), the comparison is not detailed here.

By applying the proposed protocol to circumstance 2, the communication process is shown as follows.

**Step 1:** WSA sends its list of supported language combinations in a <TNPrepareRequest> message to WSB. The message is shown in figure 7.7 below.

```
<TNPrepareRequest Resource="http://WSB/Bob/Resource/R"
RemoteResourceOwner="http://WSB/Bob">
  <StrategyList Number="1">
    <Strategy ID="1">parsimonious</Strategy>
  </StrategyList>
  <LanguageCombinations Number="2">
    <LanguageCombination ID="1">
      <PolicyLanguage>PL1</PolicyLanguage>
      <CredentialLanguage>CL1</CredentialLanguage>
    </LanguageCombination>
    <LanguageCombination ID="2">
      <PolicyLanguage>PL2</PolicyLanguage>
      <CredentialLanguage>CL2</CredentialLanguage>
    </LanguageCombination>
  </LanguageCombinations>
</TNPrepareRequest>
```

Figure 7.7. Message 1 in the preparation stage in circumstance 2

**Step 2:** Following the process dealing with the <TNPrepareRequest> message (see figure 4.4), WSB discovers that the strategy supported by the counterpart is the parsimonious strategy. It then checks available information about Bob's resource R, where the eager strategy is the only option. As WSB cannot discover a common strategy, it responds with a <TNPrepareResponse> message that TN cannot be used due to an internal interoperability issue of TN. The message is shown in figure 7.8 below.

```
<TNPrepareResponse TNCanBeUsed="no">
  <Fault>Strategic interoperability issue</Fault>
</TNPrepareResponse>
```

Figure 7.8. Message 2 in the preparation stage in circumstance 2

As the <TNPrepareResponse> message contains a fault message, according to the process dealing with this response message (see figure 4.5), WSA will cease the following communication with WSB.

Circumstance 3:

Alice wants to access Bob's resource R. Bob declares a $P_2$ requiring attribute information in a $C_1$ to protect R. Alice has a $C_1$.

The possessed information presented in symbols is shown in figure 7.9 below.

WSA (Alice)
Credentials: $C_1$
Policies: nil

WSB (Bob):
Resource: R
Credentials: nil
Policies: $P_2$:R$\leftarrow C_1$

Figure 7.9. Possessed information in circumstance 3

*Units of Analysis:*

- Authorisation system types: both WSA and WSB use ABAC-based authorisation systems (PERMIS and Akenti for instance)
- Strategy types: both WSA and WSB support the parsimonious strategy
- Policy/Credential language: both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$. WSA can also compare policies written in $PL_2$ against credentials written in $CL_2$.
- Sensitivity of credentials: nil
- Sensitivity of policies: nil

As the comparison between the improved TN protocol and the protocols of PERMIS and Akenti used in Web Services have been presented in table 7.2, it is omitted here.

By applying the proposed protocol to circumstance 3, the communication process is shown as follows.
As step 1 is the same as that shown in circumstance 2, it is omitted here.
**Step 2:** Following the process dealing with the <TNPrepareRequest> message (see figure 4.4), WSB discovers that the strategy supported by the counterpart is the parsimonious strategy. It then checks available information about Bob's resource R, where the parsimonious strategy is also available. As there is a common strategy,

WSB then discovers that the language combinations supported by the counterpart are "$PL_1$ and $CL_1$" and "$PL_2$ and $CL_2$". After this discovery, it searches the information about the policy languages used within its own system and knows that "$PL_1$" is supported. It also checks the language information about Bob's credentials to discover that "$CL_1$" is the language used for expressing Bob's credentials. It also checks the capability of its local policy compliance checker and discovers that it can compare policies written in $PL_1$ against credentials written in $CL_1$. Therefore, "$PL_1$ and $CL_1$" have been identified as the common language combination. Since WSB can discover a common strategy and a common language combination, it stores the information about the strategy name and language combination for expressing credentials and policies into its local database. After this, it agrees to use TN with WSA to trigger the negotiation stage by sending a <TNPrepareResponse> message. The message is shown in figure 7.10 below.

```
<TNPrepareResponse TNCanBeUsed="yes">
  <ChosenStrategy>Parsimonious</ChosenStrategy>
  <ChosenLanguageCombination>
    <PolicyLanguage>PL1</PolicyLanguage>
    <CredentialLanguage>CL1</CredentialLanguage>
  </ChosenLanguageCombination>
</TNPrepareResponse>
```

Figure 7.10. Message 2 in the preparation stage in circumstance 3

Applying the "Resource API" design pattern that a request should be composed of standardised server methods (i.e. GET, PUT, POST, DELETE) and a URI.

**Step 3:** When WSA receives the <TNPrepareReponse> message, following the process of dealing with this message (see figure 4.5), it can discover a list of names of the chosen strategy and the chosen language combination. It knows that the negotiation stage can be triggered. It then stores the chosen strategy name and the chosen language combination name into its local database. After that, WSA sends an <AuthzDecisionQuery> message as an authorisation request to WSB. The message is shown in figure 7.11 below.

```
<AuthzDecisionQuery ID="1"
Destination="http://WSB" Resource="http://WSB/Bob/Resource/R"
RemoteResourceOwner="http://WSB/Bob"
LocalRequesterName="http://WSA/Alice">
  <Subject>
```

```
    <SubjectConfirmation Method="TN">
  </Subject>
  <Action>Access</Action> //Access equals to the GET method
</AuthzDecisionQuery>
```

Figure 7.11. Message 3 in the negotiation stage in circumstance 3

**Step 4:** After WSB receives the <AuthzDecisionQuery> message, following the process dealing with this message (see figure 4.6), it obtains the relevant attribute information and stores this in the database. WSB then requests the local database to obtain the stored chosen strategy name – parsimonious strategy, and uses this strategy name to activate the relevant strategy in the "negotiation strategy repository component". It then checks the information about Bob's resource R and discovers that R has been protected by a $P_2$ requiring a $C_1$. As the parsimonious strategy has been chosen, it sets this policy in a <PolicySet> message to WSA. The message is shown in figure 7.12 below.

```
<PolicySet ID="bps1" LocalPolicyFileName="http://WSB/Bob/Policy/P2"
RemoteResourceOwner="http://WSA/Alice"
LocalPolicyOwnerName="http://WSB/Bob"
ProtectedLocalResource="http://WSB/Bob/Resource/R"
PolicyTotalNumber="1">
    <Policy ID="bp1">
      A P2 written in the PL1 language requiring attribute
      information in a C1 written in the CL1 language
    </Policy>
</PolicySet>
```

Figure 7.12. Message 4 in the negotiation stage in circumstance 3

**Step 5:** When WSA receives the <PolicySet> message, following the process of dealing with this message (see figures 4.7 and 4.8), it stores the attribute information into its local database. It then requests its local database for retrieving the chosen language combination "$PL_1$ and $CL_1$". When this information is provided from the local database, it then activates the specific functionality of its policy compliance checker for comparing credentials in $CL_1$ against policies in $PL_1$. By using this functionality, it compares Alice's $C_1$ against Bob's $P_2$ contained within this message. As Alice's $C_1$ is not sensitive, WSA decides to send the $C_1$ in a <CredentialSet> message to WSB. The message is shown in figure 7.13 below.

```
<CredentialSet ID="acs1" CredentialTotalNumber="1"
```

```
MeetRemotePolicy="http://WSB/Bob/Policy/P₂"
MeetRemotePolicyOwner="http://WSB/Bob"
LocalCredentialOwner="http://WSA/Alice">
   <Credential ID="ac1" CredentialType="C₁">
   Detailed attribute information within the C₁ written in the CL₁
   language
   </Credential>
</CredentialSet>
```

Figure 7.13. Message 5 in the negotiation stage in circumstance 3

**Step 6:** WSB receives the <CredentialSet> message, according to the process of dealing with the message (see figures 4.9 and 4.11); it stores the attribute information in the database. It then sends a request to the credential verification component for choosing the relevant functionality for verifying the authenticity of credentials written in $CL_1$ for the $C_1$. Once the authenticity of the $C_1$ has been verified, it requests the local database for the chosen strategy and the chosen language combination, and sends them to the negotiation strategy repository and policy compliance checker. As the parsimonious strategy has been chosen, it then compare the $C_1$ against the $P_2$ to identify that the $C_1$ can fulfil the $P_2$. After the comparison, WSB discovers that there are no further local sensitive credentials that can be disclosed. So, it sends an <AuthzDecisionStatement> message containing the successful result along with the resource back to WSA. The message is shown in figure 7.14 below.

```
<AuthzDecisionStatement ID="n" InResponseTo="1"
Resource="http://WSB/Bob/Resource/R"
ResourceOwner="http://WSB/Bob" Decision="Permit">
   <Action>read only</Action>
</AuthzDecisionStatement>
```

Figure 7.14. Message 6 in the negotiation stage in circumstance 3

When WSA receives the <AuthzDecisionStatement> message, according to the process of dealing with this message (see figure 4.13), it checks the result and knows that TN has succeeded, which means that Bob's resource R can be accessed.

Circumstance 4:

Alice wants to access Bob's resource R. Bob declares a $P_2$ requiring attribute information in a $C_1$ or that in a $C_3$ to protect R. Alice has a sensitive $C_1$ protected by a $P_1$ requiring attribute information in a $C_2$.

The possessed information presented in symbols is shown in figure 7.15 below.

WSA (Alice)
Credentials: $C_1$
Policies: $P_1$:$C_1 \leftarrow C_2$

WSB (Bob):
Resource: R
Credentials: nil
Policies: $P_2$:$R \leftarrow C_1 \vee C_3$

Figure 7.15. Possessed information in circumstance 4

*Units of Analysis:*

- Authorisation system types: WSA uses ABAC-based authorisation systems, and WSB uses TN-based authorisation systems (PERMIS and TrustBuilder2 for instance)
- Strategy types: both WSA and WSB support the use of the parsimonious strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in PL1 against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$
- Sensitivity of policies: nil

Before applying the proposed protocol to this circumstance, a comparison between the protocol of TrustBuilder2 (Lee, Winslett and Perano, 2009) and the improved TN protocol (see figure 4.2) is needed. As the comparison between the improved TN protocol and the protocol of PERMIS used in Web Services is shown in table 7.2 above, it is omitted here. The detailed comparison is shown in table 7.3.

| **Interoperability Layer** | **Improved TN Protocol** | **TrustBuilder2 protocol** |
|---|---|---|
| Strategy | (1) Preparation stage and (2) Negotiation Strategy Repository Component | Negotiation Strategy Repository Component |
| Capability | (1) Preparation stage and (2) Multiple languages for expressing credentials and Policies | Multiple languages for expressing credentials and Policies |
| Functionality | Policy Compliance Checker | Policy Compliance Checker |
| Syntax and Semantics | XML-based languages, SAML languages and XACML languages | Non-XML-based messages |

Table 7.3. Comparison between TrustBuilder2 and the improved TN protocol

Observing table 7.3, it can be identified that protocol messages and the preparation stage are not supported in TrustBuilder2. Assuming that they are to be implemented within TrustBuilder2, then the following steps will occur.

By applying the proposed protocol to circumstance 4, the communication process is shown as follows.
Steps 1 to 4 are the same as those shown in above circumstances, so they are omitted here.
**Step 5:** After WSA analyses this <PolicySet> message (see figures 4.7 and 4.8), it finds out that Alice does have a $C_1$ that can fulfil the $P_2$. Unfortunately, as Alice has treated the $C_1$ as sensitive and has declared a $P_1$ protecting its disclosure, it cannot send the $C_1$ directly to WSB. As a result, it sets the $P_1$ in a <PolicySet> message to be sent to WSB. The message is shown in figure 7.16 below.

```
<PolicySet ID="aps1" LocalPolicyFileName="http://WSA/Alice/Policy/P1"
RemoteResourceOwner="http://WSB/Bob"
LocalPolicyOwnerName="http://WSA/Alice"
ProtectedLocalResource="http://WSA/Alice/Credential/C1"
PolicyTotalNumber="1">
     <Policy ID="ap1">
       A P1 written in the PL1 language requiring the attribute
       information in a C2 written in the CL1 language
     </Policy>
</PolicySet>
```

Figure 7.16. Message 5 in the negotiation stage in circumstance 4

**Step 6:** When WSB receives this <PolicySet> message (see figures 4.7 and 4.8), it finds out that Bob does not have a $C_2$. Therefore, it sends out an <AuthzDecisionStatement> message as the last message to inform that TN has failed. The message is shown in figure 7.17 below, where the fault reason is highlighted in bold.

```
<AuthzDecisionStatement ID="n" InResponseTo="1"
Resource="http://WSB/Bob/Resource/R"
ResourceOwner="http://WSB/Bob" Decision="Denied">
   <Fault>No Local Credentials</Fault>
</AuthzDecisionStatement>
```

Figure 7.17. Message 6 in the negotiation stage in circumstance 4

When WSA receives this <AuthzDecisionStatement>, it finds out a <Fault> message is contained within this message, which means that authorisation has failed.

<u>Circumstance 5:</u>

Alice wants to access Bob's resource R. Bob declares a $P_2$ requiring attribute information in a $C_1$ to protect R. Alice has a sensitive $C_1$ protected by a $P_1$ requiring attribute information in a $C_2$. She also has a $C_3$. Bob possess a $C_4$.

The possessed information presented in symbols is shown in figure 7.18 below.

WSA (Alice)
Credentials: $C_1$, $C_3$
Policies: $P_1$:$C_1 \leftarrow C_2$

WSB (Bob):
Resource: R
Credentials: $C_4$
Policies: $P_2$:$R \leftarrow C_1$

Figure 7.18. Possessed information in circumstance 5

*<u>Units of Analysis:</u>*

- Authorisation system types: WSA uses ABAC-based authorisation systems, and WSB uses TN-based systems (PERMIS and Trust-X for instance)
- Strategy types: Both WSA and WSB supports the use of the eager strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in PL1 against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$
- Sensitivity of policies: nil

Before applying the proposed protocol to this circumstance, a comparison between the protocol of Trust-X (Squicciarini et al., 2007; Squicciarini et al., 2012) and the improved TN protocol (see figure 4.2) is needed. As the comparison between the improved TN protocol and the protocol of PERMIS used in Web Services is shown in table 7.2 above, it is omitted here. The detailed comparison is shown in table 7.4.

| **Interoperability Layer** | **Improved TN Protocol** | **Trust-X protocol** |
|---|---|---|
| Strategy | (1) Preparation stage and (2) Negotiation Strategy Repository Component | Strategy manager |
| Capability | (1) Preparation stage and (2) Multiple languages for expressing credentials | X-TNL language only |

|  |  |  |
|---|---|---|
|  | and Policies |  |
| Functionality | Policy Compliance Checker | (1) Policy Compliance Checker, (2) suspension and resume of TN and (3) trust tickets |
| Syntax and Semantics | XML-based languages, SAML languages and XACML languages | X-TNL language only |

Table 7.4. Comparison between Trust-X and the improved TN protocol

After comparing the protocols used in Trust-X against the proposed improved TN protocol (see figure 4.2), it can be identified that the preparation stage and protocol messages are not supported in Trust-X. Assuming they are to be implemented in Trust-X, then the following steps will occur.

By applying the proposed protocol to circumstance 5, the communication process is shown as follows.

The steps 1 to 3 are nearly the same as those shown in circumstance 3. The only difference is that the “eager” value is set in the <Strategy> message contained in a <TNPrepareRequest> message and in the <ChosenStrategy> message contained in a <TNPrepareResponse> message. Therefore, they are omitted here.

**Step 4:** After receiving the <AuthzDecisionQuery message>, following the logic shown in figure 4.6, WSB discovers that R has been protected by a $P_2$ requiring a $C_1$. As the eager strategy has been chosen, it then sets Bob’s $C_4$ in a <CredentialSet> message and sends this message to WSA. The message is shown in figure 7.19 below.

```
<CredentialSet ID="acs1" CredentialTotalNumber="1"
LocalCredentialOwner="http://WSB/Bob">
  <Credential ID="bc1" CredentialType="C4">
  Detailed attribute information within the C4 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.19. Message 4 in the negotiation stage in circumstance 5

**Step 5:** When WSA receives this <CredentialSet> message from WSB, following the logic shown in figures 4.9 and 4.10, it tries to use the contained $C_4$ to unlock Alice’s sensitive credentials. As the $C_4$ cannot unlock the $C_1$, and as the eager strategy has been chosen, it then sets Alice’s $C_3$ in a <CredentialSet> message and sends this message to WSB. The message is shown in figure 7.20 below.

```
<CredentialSet ID=”acs1” CredentialTotalNumber=”1”
LocalCredentialOwner=”http://WSA/Alice”>
  <Credential ID=”ac1” CredentialType=”C3”>
  Detailed attribute information within the C3 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.20. Message 5 in the negotiation stage in circumstance 5

**Step 6:** Likewise, when WSB receives this <CredentialSet> message, following the logic shown in figures 4.9 and 4.10, it looks to see whether the $C_3$ can be used to unlock Bob’s resource R. After a comparison, it finds out that the $C_3$ cannot fulfil the $P_2$; therefore it tries to use the $C_3$ to unlock Bob’s sensitive credentials. As Bob does not have any other sensitive credentials, WSB identifies that resending the $C_4$ within a <CredentialSet> message would be useless. Therefore, it sends out an <AuthzDecisionStatement> to WSA. The message is the same as the one shown in figure 7.17, so it is omitted here.

WSA finds out that a <Fault> message is contained within the received <AuthzDecisionStatement> message, so it knows that authorisation has failed.

Circumstance 6:

Alice wants to access Bob’s resource R. Bob declares a $P_2$ requiring attribute information in a $C_1$ to protect R. As Bob treats the $P_2$ as sensitive, it then declares a $P_4$ requiring attribute information in a $C_3$ or that in a $C_5$ to protect the disclosure of the $P_2$. Alice has a sensitive $C_1$ protected by a $P_1$ requiring attribute information in a $C_2$. She also has a $C_3$. Bob has a $C_2$.

The possessed information presented in symbols is shown in figure 7.21 below.

WSA (Alice)
Credentials: $C_1$, $C_3$
Policies: $P_1{:}C_1 \leftarrow C_2$

WSB (Bob):
Resource: R
Credentials: $C_2$
Policies:
$P_2{:}R \leftarrow C_1$
$P_4{:}P_2 \leftarrow C_3 \vee C_5$

Figure 7.21. Possessed information in circumstance 6

*Units of Analysis:*

- Authorisation system types: WSA uses ABAC-based authorisation systems, and WSB uses TN-based systems (PERMIS and Trust-X for instance)
- Strategy types: both WSA and WSB supports the use of the parsimonious strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in PL1 against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$
- Sensitivity of policies: $P_2$

As the comparison between the improved TN protocol and the protocol of PERMIS as well as the Trust-X protocol have been presented in table 7.2 and table 7.4 respectively, it is omitted here.

By applying the proposed protocol to circumstance 6, the communication process is shown as follows.
The steps 1 to 3 are the same as those shown in circumstance 3, so they are omitted here.
**Step 4:** After receiving the <AuthzDecisionQuery> message, following the logic shown in figure 4.6, WSB finds out that R is protected by a $P_2$, which is in turn protected by a $P_4$. Therefore, it sets the $P_4$ in a <PolicySet> message to be sent to WSA first. The message is shown in figure 7.22 below.

```
<PolicySet ID="bps1" LocalPolicyFileName="http://WSB/Bob/Policy/P4"
RemoteResourceOwner="http://WSA/Alice"
LocalPolicyOwnerName="http://WSB/Bob"
ProtectedLocalResource="http://WSB/Bob/Policy/P2"
PolicyTotalNumber="1">
     <Policy ID="bp1">
       A P4 written in the PL1 language requiring attribute
       information in a C3 or in a C5 written in the CL1 language
     </Policy>
</PolicySet>
```

Figure 7.22. Message 4 in the negotiation stage in circumstance 6

**Step 5:** WSA analyses this <PolicySet> message following the logic shown in figures 4.7 and 4.8, and understands that a $C_3$ is required. It then checks whether Alice has a $C_3$. After it checks that Alice has a $C_3$ that can fulfil the $P_2$, and that the $C_3$ is not treated as sensitive, it then sets the $C_3$ in a <CredentialSet> message and sends this message to WSB. As the message is the same as shown in figure 7.20 above, it is omitted here.

**Step 6:** WSB compares the $C_3$ contained within the received <CredentialSet> message against the $P_4$ following the logic shown in figures 4.9 and 4.11, it then makes sure that the $P_4$ has been unlocked. Therefore, it knows that the $P_2$ can be set in a new <PolicySet> message to be sent to WSA. The message is the same as shown in figure 7.12 above, so it is omitted here.

Steps 7 and 8 are the same as steps 5 and 6 as shown in circumstance 3, so the detailed descriptions are omitted here.

Circumstance 7:

Alice wants to access Bob's resource R. Bob declares a sensitive $P_2$ requiring attribute information in a $C_1$ to protect R, and declares a $P_4$ requiring attribute information in a $C_3$ to protect $P_2$. Alice has a $C_1$ and a $C_3$. Bob has a $C_2$.

The possessed information presented in symbols is shown in figure 7.23 below.

WSA (Alice)
Credentials: $C_1$, $C_3$
Policies: nil

WSB (Bob):
Resource: R
Credentials: $C_2$
Policies:
$P_2$:R$\leftarrow C_1$
$P_4$:$P_2 \leftarrow C_3$

Figure 7.23. Possessed information in circumstance 7

*Units of Analysis:*

- Authorisation system types: WSA uses ABAC-based authorisation systems, and WSB uses TN-based systems (PERMIS and Trust-X for instance)
- Strategy types: both WSA and WSB only support the use of the eager strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$.
- Sensitivity of credentials: nil
- Sensitivity of policies: $P_2$

As the comparisons between the improved TN protocol and the protocol of PERMIS as well as the Trust-X protocol have been presented in table 7.2 and table 7.4 respectively, they are omitted here.

By applying the proposed protocol to circumstance 7, the communication process is shown as follows.

Steps 1 to 3 are the same as those shown in circumstance 5. Step 4 in this circumstance is almost similar to step 4 in circumstance 5, where $C_2$ is used in this circumstance instead of $C_4$. So they are omitted here.

**Step 5:** After receiving the <CredentialSet> message, following the logic shown in figures 4.9 and 4.10, as the eager strategy has been used, WSA tries to use the $C_2$ contained within the received <CredentialSet> message to unlock local sensitive credentials. As Alice does not have any sensitive credentials, no more of Alice's sensitive credentials can be unlocked by the $C_2$. WSA finds out that the $C_1$ and the $C_3$ are not treated as sensitive, so it then sets the $C_1$ and the $C_3$ in a <CredentialSet> message and sends this message to WSB. The message is shown in figure 7.24 below.

```
<CredentialSet ID="acs1" CredentialTotalNumber="2"
LocalCredentialOwner="http://WSA/Alice">
  <Credential ID="ac1" CredentialType="C1">
  Detailed attribute information within the C1 written in the CL1
  language
  </Credential>
  <Credential ID="ac2" CredentialType="C3">
  Detailed attribute information within the C3 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.24. Message 5 in the negotiation stage in circumstance 7

**Step 6:** Following the logic shown in figures 4.9 and 4.10, WSB compares the received credentials $C_1$ and $C_3$ contained within the <CredentialSet> message against the $P_2$ protecting Bob's resource R respectively. It finds out that the $C_1$ can successfully meet the $P_2$, so it sends an <AuthzDecisoinStatement> message to inform WSA that TN has succeeded. The message is the same as the one shown in figure 7.14, so it is omitted here.

Circumstance 8:

Alice wants to access Bob’s resource R. Bob declares a sensitive $P_2$ requiring attribute information in a $C_1$ to protect R, and declares a $P_4$ requiring attribute information in a $C_3$ to protect the $P_2$. Alice has a $C_1$ and a $C_3$, which contain similar attribute information. As Alice treats the $C_1$ as sensitive, she declares a $P_1$ requiring attribute information in a $C_4$ to unlock the disclosure of the $C_1$. Bob has a $C_2$.

The possessed information presented in symbols is shown in figure 7.25 below.

| WSA (Alice) | WSB (Bob): |
|---|---|
| Credentials: $C_1$, $C_3$<br>Policies: $P_1$:$C_1 \leftarrow C_4$ | Resource: R<br>Credentials: $C_2$<br>Policies:<br>$P_2$:$R \leftarrow C_1$<br>$P_4$:$P_2 \leftarrow C_3$ |

Figure 7.25. Possessed information in circumstance 8

*Units of Analysis:*

- Authorisation system types: both WSA and WSB use TN-based systems (TrustBuilder2 and Trust-Serv for instance)
- Strategy types: both WSA and WSB only support the use of the eager strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$
- Sensitivity of policies: $P_2$

Before applying the proposed protocol to this circumstance, a comparison between Trust-Serv (Skogsrud et al., 2009) and the improved TN protocol (see figure 4.2) is needed. As the comparison between the improved TN protocol and the protocol of TrustBuilder2 used in Web Services is shown in table 7.3 above, it is omitted here. The detailed comparison is shown in table 7.5.

| **Interoperability Layer** | **Improved TN Protocol** | **Trust-Serv protocol** |
|---|---|---|
| Strategy | (1) Preparation stage and (2) Negotiation Strategy Repository Component | Negotiation Controller |
| Capability | (1) Preparation stage and (2) Multiple languages for expressing credentials | SAML messages for expressing credentials and WS- |

| | | |
|---|---|---|
| | and Policies | SecurityPolicy messages for expressing policies |
| Functionality | Policy Compliance Checker | (1) Policy Compliance Checker and (2) Policy migration |
| Syntax and Semantics | XML-based languages, SAML languages and XACML languages | SAML and WS-SecurityPolicy messages |

Table 7.5. Comparison between Trust-Serv and the improved TN protocol

Observing table 7.5, it can be identified that the inability of using multiple languages for expressing credentials and policies, the preparation stage, and the use of XACML messages as containers for containing policies expressed in different policy languages are not supported in Trust-Serv. Assuming that they are to be implemented within Trust-Serv, the following steps will occur.

By applying the proposed protocol to circumstance 8, the communication process is shown as follows.

Steps 1 to 3 are the same as those shown in circumstance 5. Step 4 in this circumstance is almost similar to step 4 in circumstance 5, where $C_2$ is used in this circumstance instead of $C_4$. So they are omitted here.

**Step 5:** After receiving the <CredentialSet> message, following the logic shown in figures 4.9 and 4.10, as the eager strategy has been used, WSA tries to use the $C_2$ contained within the received <CredentialSet> message to unlock local sensitive credentials. Alice does possess a credential $C_1$, but the relevant P1 declared by Alice requires a $C_4$ to unlock the disclosure of the $C_1$. As $C_2$ cannot meet the $P_1$, so the disclosure of the $C_1$ cannot be unlocked. WSA finds out that only the $C_3$ is not treated as sensitive, so it then sets the $C_3$ in a <CredentialSet> message and sends this message to WSB. The message is shown in figure 7.26 below.

```
<CredentialSet ID="acs1" CredentialTotalNumber="1"
LocalCredentialOwner="http://WSA/Alice">
  <Credential ID="ac1" CredentialType="C3">
  Detailed attribute information within the C3 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.26. Message 5 in the negotiation stage in circumstance 8

**Step 6:** Following the logic shown in figures 4.9 and 4.10, WSB compares the received credential $C_3$ contained within the <CredentialSet> message against the $P_2$ protecting Bob's resource R. It finds out that the attribute information in the $C_3$ can successfully meet the $P_1$, so it sends an <AuthzDecisoinStatement> message to inform WSA that TN has succeeded. The message is the same as the one shown in figure 7.14, so it is omitted here.

<u>Circumstance 9:</u>

Alice wants to access Bob's resource R. Bob declares a sensitive $P_2$ requiring attribute information in a $C_1$ to protect R, and declares a $P_4$ requiring attribute information in a $C_3$ to protect the $P_2$. Alice has a $C_1$ and a $C_3$, which contain similar attribute information. As Alice treats both the $C_1$ and the $C_3$ as sensitive, she declares a $P_1$ requiring attribute information in a $C_4$ to unlock the disclosure of the $C_1$ and the $C_3$. Bob has a $C_2$.

The possessed information presented in symbols is shown in figure 7.27 below.

WSA (Alice)
Credentials: $C_1$, $C_3$
Policies:
$P_1$: $(C_1, C_3) \leftarrow C_4$

WSB (Bob):
Resource: R
Credentials: $C_2$
Policies:
$P_2$:$R \leftarrow C_1$
$P_4$:$P_2 \leftarrow C_3$

Figure 7.27. Possessed information in circumstance 9

*<u>Units of Analysis:</u>*

- Authorisation system types: both WSA and WSB use TN-based systems (TrustBuilder2 and Trust-Serv for instance)
- Strategy types: both WSA and WSB only support the use of the parsimonious strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$, $C_3$
- Sensitivity of policies: $P_2$

As the comparison between the improved TN protocol and the TrustBuilder2 protocol and Trust-Serv protocol have been presented in table 7.3 and table 7.5 respectively, it is omitted here.

By applying the proposed protocol to circumstance 9, the communication process is shown as follows.

Steps 1 to 3 are the same as those shown in circumstance 3. Step 4 in this circumstance is almost similar to step 4 in circumstance 3, where $P_4$ is used instead of $P_2$. So they are omitted here.

**Step 5:** After receiving the <PolicySet> message, following the logic shown in figures 4.7 and 4.8, as the parsimonious strategy has been used, WSA tries to compare the $C_1$ and the $C_3$ against the $P_4$ and discovers that attribute information in either the $C_1$ or the $C_3$ can fulfil the $P_4$. Unfortunately, as disclosure of the $C_1$ and the $C_3$ are protected by the $P_1$, so WSA sets the $P_1$ in a <PolicySet> message to be sent to WSB. The message is shown in figure 7.28 below.

```
<PolicySet ID="aps1" LocalPolicyFileName="http://WSA/Alice/Policy/P1"
RemoteResourceOwner="http://WSB/Bob"
LocalPolicyOwnerName="http://WSA/Alice"
ProtectedLocalResource="http://WSA/Alice/Credential/C1&C2"
PolicyTotalNumber="1">
     <Policy ID="ap1">
       A P1 written in PL1 language requiring attribute information in
       a C4 written in the CL1 language
     </Policy>
</PolicySet>
```

Figure 7.28. Message 5 in the negotiation stage in circumstance 9

**Step 6:** Following the logic shown in figures 4.7 and 4.8, WSB tries to compare the $P_1$ contained within the received <PolicySet> message against Bob's $C_2$. As the attribute information in the $C_2$ cannot fulfil the $P_1$, and WSB finds out that Bob does not possess any other credentials except the $C_2$, so it sends an <AuthzDecisionStatement> message containing a <Fault> message to WSA. The message is the same as the one shown in figure 7.17, so it is omitted here.

When WSA receives this <AuthzDecisionStatement> message and detects a <Fault> message, it knows that authorisation has failed.

Circumstance 10:

Alice wants to access Bob's resource R. Bob declares a sensitive $P_2$ requiring attribute information in a $C_1$ to protect R, and declares a $P_4$ requiring attribute information in a $C_3$ to protect the $P_2$. Alice has a $C_1$ and a $C_3$, which contain similar attribute information. As Alice treats both the $C_1$ and the $C_3$ as sensitive, she declares a $P_1$ requiring attribute information in a $C_4$ to unlock the disclosure of the $C_1$ and the $C_3$. In addition, Alice also has a $C_7$ and a $C_9$. Bob has a $C_2$ and a $C_4$. As Bob treats the $C_4$ as sensitive, he declares a $P_6$ requiring attribute information in a $C_5$ or a combination of a $C_7$ and a $C_9$ to unlock the disclosure of the $C_4$.

The possessed information presented in symbols is shown in figure 7.29 below.

WSA (Alice)
Credentials: $C_1$, $C_3$, $C_7$, $C_9$
Policies:
$P_1$: $(C_1, C_3) \leftarrow C_4$

WSB (Bob):
Resource: R
Credentials: $C_2$, $C_4$
Policies:
$P_2$:$R \leftarrow C_1$
$P_4$:$P_2 \leftarrow C_3$
$P_6$:$C_4 \leftarrow C_5 \vee (C_7 \wedge C_9)$

Figure 7.29. Possessed information in circumstance 10

*Units of Analysis:*

- Authorisation system types: both WSA and WSB use TN-based systems (TrustBuilder2 and Trust-Serv for instance)
- Strategy types: both WSA and WSB only support the use of the parsimonious strategy
- Policy/Credential language: Both WSA and WSB can compare policies written in $PL_1$ against credentials written in $CL_1$.
- Sensitivity of credentials: $C_1$, $C_3$, $C_4$
- Sensitivity of policies: $P_2$

As the comparison between the improved TN protocol and the TrustBuilder2 protocol and Trust-Serv protocol have been presented in table 7.3 and table 7.5 respectively, it is omitted here.

By applying the proposed protocol to circumstance 10, the communication process is shown as follows.

Steps 1 to 5 are the same as those shown in circumstance 9, so they are omitted here.

**Step 6:** After receiving the <PolicySet> message, following the logic shown in figures 4.7 and 4.8, as the parsimonious strategy has been used, WSB tries to compare the $P_1$ contained within the received <PolicySet> message against Bob's $C_4$ and discovers that the attribute information in the $C_4$ can fulfil the $P_1$. However, as disclosure of the $C_4$ is protected by the $P_6$, so WSB sets the $P_6$ in a <PolicySet> message to be sent to WSA. The message is shown in figure 7.30 below.

```
<PolicySet ID="bps2" LocalPolicyFileName="http://WSB/Bob/Policy/P6"
RemoteResourceOwner="http://WSA/Alice"
LocalPolicyOwnerName="http://WSB/Bob"
ProtectedLocalResource="http://WSB/Bob/Credential/C4"
PolicyTotalNumber="1">
     <Policy ID="bp2">
       A P6 written in the PL1 language requiring attribute
       information in a C5 or in a combination of C7 and C9
       written in the CL1 language
     </Policy>
</PolicySet>
```

Figure 7.30. Message 6 in the negotiation stage in circumstance 10

**Step 7:** Following the logic shown in figures 4.7 and 4.8, as WSA cannot find the $C_5$ possessed by Alice, it then compares the combination of Alice's $C_7$ and $C_9$ against the $P_6$, and finds out that the attribute information in insensitive $C_7$ and $C_9$ can meet the $P_6$, so it sets the $C_7$ and the $C_9$ in a <CredentialSet> message to be sent to WSB. The message is shown in figure 7.31 below.

```
<CredentialSet ID="acs2" CredentialTotalNumber="2"
MeetRemotePolicy="http://WSB/Bob/Policy/P6"
MeetRemotePolicyOwner="http://WSB/Bob"
LocalCredentialOwner="http://WSA/Alice">
   <Credential ID="ac1" CredentialType="C7">
   Detailed attribute information within the C7 written in the CL1
   language
   </Credential>
   <Credential ID="ac2" CredentialType="C9">
   Detailed attribute information within the C9 written in the CL1
   language
```

```
  </Credential>
</CredentialSet>
```

Figure 7.31. Message 7 in the negotiation stage in circumstance 10

**Step 8:** Following the logic shown in figures 4.9 and 4.11, WSB compares the attribute information in the $C_7$ and $C_9$ against the $P_6$, and agrees that the $C_7$ and the $C_9$ can fulfil the $P_6$. As $P_6$ does not protect any local sensitive policies, so it sets the $C_4$ in a <CredentialSet> message to be sent to WSA. The message is shown in figure 7.32 below.

```
<CredentialSet ID="acs1" CredentialTotalNumber="1"
MeetRemotePolicy="http://WSA/Alice/Policy/P1"
MeetRemotePolicyOwner="http://WSA/Alice"
LocalCredentialOwner="http://WSB/Bob">
  <Credential ID="bc1" CredentialType="C4">
  Detailed attribute information within the C4 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.32. Message 8 in the negotiation stage in circumstance 10

Step 9: Following the logic shown in figures 4.9 and 4.11, WSA compares the attribute information in the $C_4$ against the $P_1$, and discovers that the $C_4$ can meet the $P_1$. So it then sets the $C_3$ in a <CredentialSet> message to be sent to WSB. The message is shown in figure 7.33 below.

```
<CredentialSet ID="acs3" CredentialTotalNumber="1"
MeetRemotePolicy="http://WSB/Bob/Policy/P4"
MeetRemotePolicyOwner="http://WSB/Bob"
LocalCredentialOwner="http://WSA/Alice">
  <Credential ID="ac3" CredentialType="C3">
  Detailed attribute information within the C3 written in the CL1
  language
  </Credential>
</CredentialSet>
```

Figure 7.33. Message 9 in the negotiation stage in circumstance 10

Step 10: Following the logic shown in figures 4.9 and 4.11, WSB compares the attribute information in the $C_3$ against the $P_4$, and agrees that the attribute information in the $C_3$ can fulfil the $P_4$; so it sets the $P_2$ in a <PolicySet> message to be sent to WSA. The message is the same as the one shown in figure 7.12, so it is omitted here.

Step 11 and step 12 are the same as step 5 and step 6 shown in circumstance 3 respectively, so they are omitted here.

The case study consisting of ten circumstances is presented above, the next section discusses data analysis based on the data collected from the ten circumstances.

## 7.4 Discussion on Qualitative Data Analysis

### 7.4.1 Data analysis strategy and technique

Following step 4 of the case study protocol shown in section 7.3.1, this section presents the relevant qualitative data analysis results. Yin (2013) recommends four strategies and five techniques to be used for case study data analysis.

The four strategies are (1) Relying on theoretical propositions (used when there exists a proposition within a piece of research), (2) Working your data from the "ground up" (used when there is no clear proposition within a piece of research), (3) Developing a case description (used when the first and the second strategies are not available) and (4) Examining plausible rival explanations (used when there exists rival explanations within the propositions within a piece of research). A conceptual multi-layered interoperability-solution design presented in Table 4.1 illustrating the mapping between each interoperability layer and each protocol element is a proposition without containing any rival explanation. This table proposes guidance for the design and development of the improved TN protocol. Therefore, the first strategy should be suitable to be used as the most appropriate data analysis strategy in the data analysis phase.

The five techniques are (1) Pattern Matching (used when a comparison between the findings from the data collection is the same as the predicted results before data collection is possible), (2) Explanation Building (used when the goal of the case study is to establish an explanation), (3) Time-Series Analysis (used when the case study may last for a long time, where data collected in different period may be different), (4) Logic Models (used when studying relationships between causes and effects) and (5) Cross-Case SynThesis (used when conducting multiple case studies). According to the second research aim, the research is not going to identify explanations for certain phenomena or to study relationships between causes and effects, but to explore potential solutions. As mentioned earlier, the selected type of the case study is an embedded single-case study design, and the data collection will not be affected by the time. As indicated by the proposed interoperability-solution design in table 4.1, a protocol designed following this design may address the second research problem (referred to as predicted results). The goal of this protocol evaluation is to verify whether the proposed protocol following the guidance of the design in table 4.1 can address the second research problem (referred to as evaluation results). Therefore, a comparison between the evaluation results against the predicted results is the purpose of conducting this evaluation. Following this purpose, the selection of the "pattern matching" technique as the best data analysis technique should be appropriate.

In conclusion, the strategy of reliance on "theoretical propositions" strategy and the "pattern matching" technique are selected for evaluation result data analysis.

### 7.4.2 Data analysis results for interoperability between authorisation systems and correctness of the protocol

The data analysis of this evaluation mainly aims at assessing whether the proposed improved TN protocol as a concrete example of utilising the interoperability-solution design can address the second research problem raised in section 1.2.1.

Within circumstance 1 of the case scenario as shown in section 3.4, it can be identified that the initial reason for causing the interoperability issues between two systems is attributed to the asymmetric strategic interoperability. In other words, the eager strategy is used within the TN-based authorisation system as a service provider, whereas it is not supported in the ABAC-based authorisation system as a service requester. Due to the unsupported strategy used in the ABAC-based authorisation system, the initial reason for causing interoperability issues is implicitly converted to the second reason, namely, asymmetric functionality interoperability. More precisely, using the functionality of comparing received credentials against local policies at this particular time point is not expected by the ABAC-based authorisation system. In other words, receiving credentials at this step does not fit the original logic designed within the internal structures of the ABAC-based authorisation system. In addition, the syntax and semantics of the message containing credentials and the syntax and semantics of the credentials used in the TN-based authorisation system may not be recognised by the ABAC-based authorisation system. The unknown syntax and semantics of the message and that of the credentials cause the syntactic and semantic interoperability issues and capability interoperability issue respectively. Therefore, the simultaneous occurrence of these different layered interoperability issues causes the interoperability issue as a final outcome between two systems.

Upon understanding the reasons underpinning this phenomenon, how these issues can be addressed with the use of the improved TN protocol through the data analysis is discussed as follows.
Observing circumstances 5 and 7 in section 7.3.4, it can be identified that the communication is also between an ABAC-based authorisation system and a TN-based authorisation system in both cases, where the eager strategy is also used within the TN-based authorisation system. Unlike circumstance 1 described in Chapter 3, there is no unexpected communication problem between two systems. Although the authorisation results in the two circumstances are different (authorisation failure in circumstance 5 and authorisation success in circumstance 7), at least communication finishes with an expected result (e.g. authorisation success or authorisation failure).

After the data analysis, it can be identified that the necessary components (e.g. the preparation stage and the eager strategy along with the relevant internal structures and the protocol messages) have been added to the ABAC/TN-based authorisation system, after making an explicit comparison between the improved TN protocol and the original protocol used in the ABAC/TN-based authorisation systems (see table 7.2). This comparison forms a necessary foundation, which can aid developers in identifying the difference between each part of a protocol used within the ABAC/TN-based authorisation systems and those within the improved TN protocol. Based on the understanding of the difference, developers can add the relevant elements to the ABAC-based and TN-based authorisation systems to avoid the occurrence of interoperability issues from layers 2 to 6.

As successful TN-based authorisation occurs in circumstance 7 shown in section 7.3.4, whereas it fails in circumstance 1 as described in section 3.4, it means that potential successful authorisation that would have failed in certain circumstances has been enabled with the use of the improved TN protocol.

Observing the three circumstances 2 to 4 described in section 3.4, the reasons for the occurrence of multiple interoperability issues in circumstance 4 are similar to those occurring in circumstance 1 as shown in section 3.4. The difference is that both the TN-based systems can support different strategies, but different strategies are not interoperable with one another. Again, the strategic interoperability issue causes the occurrence of the functionality interoperability issue, and meanwhile syntactic, semantic and capability interoperability issues may occur together. The simultaneous occurrence of all of these issues from layers 2 to 6 cause the communication between two entities to cease without reaching an expected authorisation result.

In terms of circumstances 2 and 3 described in section 3.4, the main reason for causing the occurrence of the interoperability issue in relation to capability is due to the fact that two unknown entities do not know what capabilities are owned by each other. More precisely, potential successful authorisation may fail, when two unknown entities do not realise that they actually have a common capability for comparing received remote credentials/policies against local policies/credentials.

Upon understanding this reason, a discussion of how the improved TN protocol can be used to address this interoperability issue is presented as follows.
Observing circumstances 1, 2, 3, 8, 9 10 shown in section 7.3.4, in circumstances 1, 2 and 3, communication occurs between two ABAC-based authorisation systems, whereas in circumstances 8, 9 and 10, communication occurs between two TN-based authorisation systems. Although communication in circumstances 1, 2 and 3 is not between two TN-based authorisation systems, the solutions for providing interoperability in relation to strategy and capability are the same as those used in circumstance 8, 9 and 10. Therefore, circumstances 1, 2 and 3 can be used together with circumstances 8, 9 and 10 to conduct the data analysis.

In circumstances 1 and 2, interoperability issues in relation to capability (i.e. language combination is used) and strategy can be identified in the preparation stage designed within the improved TN protocol. In circumstances 3, 8, 9 and 10, communication finishes with an expected authorisation result (i.e. authorisation success or authorisation failure). The conditions in the six circumstances shown in section 7.3.4 are quite similar to those in circumstances 2, 3 and 4 shown in section 3.4, but none of the communication processes within the six circumstances stops due to occurrence of interoperability issues.

The reason that the occurrence of the interoperability issues can be avoided is due to the fact that a comparison between each part designed within the improved TN protocol against those designed within the protocols used within the existing ABAC/TN-based authorisation systems has been made. Based on the identified differences, the preparation stage and the negotiation stage along with their relevant internal structures and the protocol messages (e.g. <CredentialSet> and <PolicySet> messages) have been added to the existing ABAC/TN-based authorisation systems. With the help of the preparation stage, both ABAC/TN-based authorisation systems have identified a common strategy and a common language combination. The use of the common strategy and the common language combination for expressing credentials and policies used in the negotiation stage can ensure strategic interoperability and capability interoperability respectively. In addition, in the negotiation stage, the syntax and semantics of the <CredentialSet> and <PolicySet> messages designed as a message container along with their relevant internal structures have been proposed by taking into consideration the use of multiple languages for expressing credentials and policies.

As successful TN-based authorisation occurs in circumstances 3, 8, and 10 shown in section 7.3.4, whereas it fails in circumstances 2, 3 and 4 as described in section 3.4, it means that potential successful authorisation that would have failed in certain circumstances has been enabled with the use of the improved TN protocol.

In terms of the correctness of the TN protocol, it can be verified through circumstances 3 to 10 shown in section 7.3.4, as successful or failed authorisation can occur following the specific conditions according to the rule of TN (see the general concept of TN presented in section 3.2). In addition, conditions in circumstances 6 and 9 shown in section 7.3.4 are similar to the case scenario shown in section 2.8. However, unlike the case shown in section 2.8, where authorisation fails by using the

RBAC/ABAC approach, authorisation can succeed by using the proposed improved TN protocol. Based on the analysis above, the correctness of the TN protocol can hold.

## 7.5 Limitations of the Protocol or of the Interoperability-Solution Design

### 7.5.1 Conceptual interoperability issue

Observing the ten circumstances described in section 7.3.4, there are ABAC-based authorisation systems involved within the authorisation communication. Once they use the improved TN protocol, ABAC-based authorisation systems are also equipped with TN features. These features enable ABAC-based authorisation systems to use either ABAC or TN approaches when authorisation is needed.

With the addition of these TN features, ABAC-based authorisation systems such as PERMIS or Akenti, actually are implemented with at least three protocols. The first protocol is the original protocol designed for each system (Chadwick and Otenko, 2002; Thompson, Esiari and Mudumbai, 2003). The second protocol is the ABAC-based protocol designed for these authorisation systems when they are integrated together within the context of Web Services (Schlager et al., 2006). The third protocol is the improved TN protocol proposed in this research. As the existing authorisation systems may have different protocols, conceptual interoperability at layer 7 of the proposed interoperability model may need to be taken into consideration. Otherwise, protocol confusion may occur, especially when the protocol messages used in different protocols are the same. For instance, SAML messages are used in both the ABAC protocol and the proposed TN protocol.

Taking into account conceptual interoperability, questions such as how to enable an authorisation system to know which authorisation protocol is the most appropriate to be used in a specific situation may arise. For instance, demonstration of higher authorisation success probability with the use of TN than that with the use of ABAC has been proved in the existing research (see Chapter 2 and Chapter 3). It is encouraged that TN should be preferred, when sensitive credentials and policies are held by two entities. This requires a system especially a service requester to know that the TN protocol should be used in such a situation.

The service specification of a protocol as suggested in the interoperability-solution design can provide such information (in what situation, the TN protocol should be used as appropriate) for developers to understand when to use a specific protocol. However, enabling systems to understand this information when making a decision about the use of a proper protocol may become an issue in the future.

### 7.5.2 Extra required functionalities

The improved TN protocol can only provide interoperability in relation to currently defined protocol messages along with the relevant internal structures from layers 2 to 6. That is, if extra functionalities are defined within the internal structures of other TN-based authorisation systems, which are not defined within this protocol, this protocol will fail to deliver the relevant interoperability. For instance, observing the comparison between the improved TN protocols and the protocols used within the existing TN-based authorisation systems in tables 7.3 and 7.4, it can be identified that several unique functionalities are not designed within the improved TN protocol, such as: enabling the resumption of TN (e.g. when interrupted); trust tickets as designed within the Trust-X system; policy migration as designed within the Trust-Serv system. Without the design of these functionalities, the relevant protocol messages are also not supported within the improved TN protocol. This will cause potential interoperability issues again. For instance, when communication breakage occurs between a Trust-X system and an authorisation system using the improved TN protocol, and if the Trust-X system tries to use the TN resumption functionality by sending the relevant protocol messages, the authorisation system using the improved TN protocol will not be able to process such a message.

### 7.5.3 No common strategy or capability between two entities

Observed from the ten sub cases, this protocol along with the interoperability-solution design can only ensure that two entities will not miss the possibility that they have the common strategy or capability to compare credentials against policies. In other words, this protocol fails to provide interoperability if two entities have no common strategy or capability of comparing credentials against policies. This requires a further solution that can make up for this limitation of the protocol and the interoperability-solution design.

### 7.5.4 Performance tests challenge

Within ten circumstances, although some conditions are common in several circumstances, the real process of each circumstance differs from each other, once one or two conditions are changed. This means that a TN process can be very complicated in comparison with an ABAC process. More precisely, after the data analysis, the complexity of each TN process is due to several conditions: (1) strategy used in TN, (2) number of credentials and policies held by two entities and (3) sensitivity of credentials and policies. Any change of a condition may cause the TN process to change a lot. This finding indicates that performance tests of a TN protocol based on one circumstance in some research (Lee, Winslett and Perano, 2009) cannot reflect the real nature of TN. Therefore, performance tests may become a further challenge, which requires future research to identify a proper way for reflecting real performance of a TN protocol.

## 7.6 Related Work

### 7.6.1 Differences between the protocol and state-of-the-art TN-based authorisation systems

In terms of the types of distributed systems environment, the state-of-the-art TN-based authorisation systems can be classified into two categories: Web Services-based and non-Web Services-based. A representative TN-based authorisation system for Web Services is Trust-Serv, and representatives of TN-based authorisation systems for non-Web Services are TrustBuilder2, Trust-X etc.

Within Trust-Serv, SAML messages are utilised as the only one language for expressing credentials and the language defined in the WS-SecurityPolicy specification is leveraged as the only one language for expressing policies. The phenomenon of using one language for expressing credentials and policies respectively also exists in TN-based authorisation systems for non-Web Services such as Trust-X. However, the assumption of using only one language for expressing credentials and policies between two communicating systems is not true in reality

(Saikou, 2010). Lee, Winslett and Perano (2009) proposing TrustBuilder2 strongly recommend that the use of multiple languages for expressing credentials and policies should be treated as one of the important features or characteristics supported by TN, even though TrustBuilder2 is a TN-based authorisation system for non-Web Services. This point of view is accepted within this research, as it is identified that specific features used in one language may not be supported in other languages through the review (see section 2.7.1). More precisely, this finding is achieved through the analysis of the existing policy languages leveraged in the context of Web Services.

Due to the fact that each policy language owns its unique features, it is possible that developers try to enable their Web Services to support different policy languages (Lang et al., 2006). As some policies may only be expressed in a specific policy language due to some unique requirements, the support of multiple policy languages within a Web Service can provide policy expression flexibility for different circumstances. This benefit should also be applied to credential languages in Web Services.

However, given the phenomenon of using multiple languages for expressing credentials and policies within Web Services, the state-of-the-art TN-based authorisation systems cannot ensure interoperability from layers 2 to 6. This TN protocol is initially designed by combining the necessary features/functionalities of the state-of-the-art TN-based authorisation systems designed for both categories as mentioned above. It then converts the features/functionalities of these TN-based authorisation systems to elements of a protocol as identified in Chapter 2 in order to fulfil interoperability at layers 2 to 4. As the state-of-the-art TN-based authorisation systems cannot provide interoperability at layers 5 and 6, a novel preparation stage is designed to enable the TN protocol to be equipped with two higher-layered interoperability (i.e. capability and strategic interoperability). Therefore, this core idea of the preparation stage can be treated as an extension of the state-of-the-art TN-based authorisation systems.

### 7.6.2 Differences between the protocol and state-of-the-art TN-based authorisation systems in Web Services

Through the review of state-of-the-art TN-based authorisation systems within Web Services, the most related work to the improved TN protocol proposed in this Thesis are TrustServ (Skogsrud et al., 2009) and the SRNS (Liu et al., 2013). However, similarities and differences can still be identified amongst them. In terms of the similarity, all of the work aims to bring TN into Web Services to provide the establishment of a bilateral trust relationship between two unknown Web Services for achieving authorisation.

In terms of the difference, TrustServ mainly focuses on addressing the problems of policy life cycle management and policy migration of TN. As these problems are closely related to innate natures of TN rather than a specific environment of TN, TrustServ is not designed to address problems for TN used in Web Services. In other words, the reason for its relevance to this research is that its solution is implemented as Web Services.

SRNS is a strategy, which allows two TN entities to exchange extra ontology information for clarifying the semantics of the vocabularies used within policies. However, the assumption that two entities can understand the semantics of policies by using the exchanged ontology information is weak. As stated in section 3.3.1.2, TN is primarily used in a context that two entities are unknown to each other. If they do not know each other, there is no guarantee that they have a common capability of understanding the same ontology information. Fortunately, the preparation stage designed within the improved TN protocol can be added to the SRNS to complement this weakness, so that two unknown entities can at least consult with each other to make sure whether they have a common capability of understanding the same ontology information. Similar to TrustServ, the SRNS is not specifically designed for TN in Web Services either, but it is just implemented in Web Services.

By contrast, the research problems in this research are raised by taking into consideration adopting TN within the specific Web Services context, that is, the occurrence of the problems are uniquely attributed to the nature of Web Services.

More precisely, the problems that need to be addressed are in relation to interoperability issues brought by using TN within Web Services. Therefore, the proposed TN protocol mainly focuses on resolving multi-layered interoperability issues, when TN is used in Web Services. Observing the differences, it can be identified that the focus of this research is completely different from that of TrustServ and of SRNS.

### 7.6.3 Differences between the protocol and an existing solution in Web Services

The existing WS-Agreement specification (Andrieux et al., 2007) strongly recommends a pre-stage between two Web Services to consult on capability interoperability. By contrast, the improved TN protocol recommends a pre-stage between two Web Services to consult both capability and strategic interoperability. Therefore, this core idea of the preparation stage can be treated as an extension of the state-of-the-art WS-Agreement specification.

### 7.6.4 Differences between the protocol and state-of-the-art ABAC protocols in Web Services

The state-of-the-art ABAC protocols are mainly designed in the AAI and Globus. Those protocols aim to help two Web Services establish a unilateral trust relationship for authorisation. More precisely, the protocols only enable a Web Service Provider to trust a Web Service Requester. The improved TN protocol instead can enable two Web Services to establish a bilateral trust relationship for authorisation. Some failed authorisation by using the ABAC protocols due to the existence of sensitive information (e.g. sensitive policies or credentials) can be transformed into successful authorisation. As the ABAC protocols can be directly used in the Globus toolkit, the improved TN protocol should also be used in the Globus toolkit with ease.

## 7.7 Impact of the Research

In terms of the research, its impact not only comes from the contributions, but also comes from the evaluation process. As the impact of the process of the protocol completeness test in Chapters 5 and a contribution in Chapter 6 have been stated, they are not stated again.

First of all, a conceptual multi-layered interoperability model has been improved on the basis of the existing LCIM suggested by Turnitsa (2005). As a first contribution of this research, this improved interoperability model redefines the concept of some interoperability layers presented in the LCIM for illustrating interoperability issues between authorisation systems in Web Services. In particular, one novel interoperability layer named strategic interoperability is explored as an extension of the state-of-the-art LCIM. By bearing this model in mind, it can help researchers in academia and practitioners in industry identify the interoperability issues between communicating systems in other fields.

Based on this model and a review of the notions of protocols, a conceptual multi-layered interoperability-solution design presenting the mapping between interoperability at each layer and elements of a protocol is then constructed as a second contribution (see table 4.1). In this interoperability-solution design, how to use a protocol-based approach to provide interoperability at the relevant layer is presented. Following the guidance of this interoperability-solution design, an improved TN protocol is proposed as a concrete example of utilisation of the interoperability-solution design. Through the evaluation test, the protocol has been proved to be effective in some certain circumstances. As this protocol was designed following the guidance of the interoperability-solution design, the effectiveness of the solution design can also be proved. The impact of this interoperability-solution design is its potential application to other distributed systems environments for providing interoperability. In other words, to use a protocol to provide interoperability between communicating systems, a whole protocol does not necessarily have to be implemented in the communicating systems. Instead, only the novel parts (i.e. protocol messages and internal structures) designed in the protocol need to be added to the communicating systems to provide interoperability at the relevant layer as presented in the interoperability-solution design.

In terms of the impact of the improved TN protocol as a third contribution, it may not be directly used in other distributed systems environments, but the novel idea of the preparation stage may be applied to other protocols, which can also help two unknown entities communicate. Hence, the impact of this protocol is mainly embodied in the preparation stage.

The evaluation process presented in this chapter has demonstrated the usefulness of applying the model-based testing within an embedded case study design. Following guidance of a case study protocol, the researcher was able to obtain the relevant data for conducting data analysis. Based on data analysis, the effectiveness of the protocol can be successfully assessed. In terms of the impact of this evaluation, new issues coming from the interoperability-solution design and the protocol have been explored, which can drive the researcher to carry out further research. Future work is discussed in the next Chapter.

## 7.8 Chapter Summary

This chapter provides an evaluation of the proposed interoperability-solution design to evaluate its effectiveness along with an evaluation of the correctness of the improved TN protocol. More precisely, the TN protocol can provide interoperability for two authorisation systems within Web Services following the interoperability-solution design, thereby raising the success probability of TN-based authorisation in some certain circumstances. The correctness of the TN protocol is verified through a multitude of circumstances, which can demonstrate that authorisation can succeed or fail according to different conditions pre-set to the two entities. In addition, failed authorisation by using the RBAC/ABAC approaches can succeed by using this TN protocol in some circumstances. Apart from those, the impact of the research is also discussed. The next chapter discusses and concludes the Thesis and presents some future work.

# Chapter 8. Conclusions and Future Work

## 8.1 Introduction

Research investigating a solution for providing interoperability between authorisation systems within Web Services has been presented in previous chapters. This last chapter discusses the whole research process. Research contributions and their impacts are then highlighted followed by discussion of future work.

## 8.2 Discussion of the Research

An appropriate starting point enabling a discussion of the research is to verify whether the defined research problems have been addressed. As a novice researcher commencing this research, research problems or issues were in the first instance difficult to elicit and to ensure that the research issues remained current. Through a detailed literature review, the research problems began to become apparent and to track through to recent identified research solutions to identify that the issues still remained. However, the process of converting identified research issues in the field to appropriate research questions was not straightforward. It did take the researcher a long time to realise the significance of the appropriate expression of the research questions. This is because a proper expression of a research question implicitly provides a general direction for a researcher to decide which research methodology and research methods are of most appropriateness for the whole research (Yin, 2013).

The identified research problems coalesced around existing Web Service concerns regarding interoperability, particularly interoperability as it related to access control. As expressed in Chapter 1, two research problems in relation to interoperability issues were identified within this area of interest. To verify whether the two research problems have been resolved, an appropriate method is to discuss whether the relevant research questions can be answered by the relevant proposed solutions.

The first research problem was ascribed to the lack of a comprehensive understanding of factors (some known and some unknown) causing interoperability issues between authorisation systems within Web Services. This research problem was converted to a "what" research question – What are the factors that cause interoperability issues between authorisation systems within Web Services? In fact, this question may also be asked in a way of "how many?". Raised in either way, the documentary analysis method as a survey or archival-based method was favoured to enable the researcher to explore the relevant factors along with their characteristics causing the issues (Yin, 2013).

The second research problem was attributed to the lack of a unified approach that can address interoperability issues caused by the relevant factors. The second research problem was converted to a "how" research question – How can a unified approach address interoperability issues caused by the identified factors to ensure that potential successful authorisation between authorisation systems within Web Services will not fail? The reasons to ask the second research question in the way of "how" rather than "what" or "why" were: (1) a "what" question would provide vague guidance of selecting a specific research method for evaluating the effectiveness of a proposed solution, and (2) a "why" question is normally used when a researcher intends to explain the reasons underpinning some specific phenomena (Yin, 2013). By contrast, a "how" question could provide the researcher with clear guidance in selecting a specific research method for evaluation in order to verify whether a solution could provide its proposed effectiveness.

For the first research question, seven identified factors presented within an improved multi-layered interoperability model (see table 3.2 in Chapter 3) were proposed as one possible answer. Identification of these factors could help the researcher gain an overall understanding about characteristics of the factors causing interoperability issues. Construction of this interoperability model was through qualitative data analysis. The data was collected from documentation analysis (Taylor et al., 2006) along with case studies (Yin, 2013). The review included ABAC-based authorisation systems in Web Services (see Chapter 2), TN-based authorisation systems within Web Services and conceptual interoperability models (see Chapter 3). The case studies focused on verification of possible interoperability issues between authorisation systems within Web Services (see Chapter 3).

The documentation analysis research method could enable the researcher to discover the significant interoperability issues between these authorisation systems within Web

Services, but it was not strong enough to help the researcher verify whether these issues did exist in practice, or indeed whether these issues were the only issues presented in Web Service communication scenarios.

To make up for the weakness of using documentation analysis, an approach using an embedded single case study design was used (see section 3.4). This design helped to verify factors identified in the documentation analysis and to provide confidence that no other factors influencing interoperability could be identified. In addition, through the analysis of the factors presented in the LCIM (see Chapter 3), several factors used in the LCIM (i.e. technical, pragmatic, dynamic) were renamed (i.e. connected, functional, capability), as it was discovered that they could not clearly indicate the key characteristics. For instance, characteristics of the "capability" factor could account for the interoperability issue caused by the "dynamic" factor, but not vice versa. Eventually, construction of the interoperability model was completed through modification and extension of the LCIM to take into account an additional factor (i.e. strategic) identified during the analysis process.

Whilst the verification process at this stage identified no other factors (either conceptual or connected) influencing interoperability, the researcher does not completely exclude other undetermined factors. Similar to the original authors of the LCIM, the researcher understands that models can promote further research in the future, which extend them to other contexts or highlight changes in the identified context, which can have an impact on the model. It is highlighted therefore that any remaining undetermined factors may weaken the effectiveness of the proposed interoperability model, and are unlikely to be covered by the current interoperability-solution design (see table 4.1 presented in Chapter 4).

The LCIM is a published model that went through a peer review process through publication (Tolk and Muguira, 2003; Turnista, 2005). Therefore to strengthen the findings made within the context of this Thesis of the modification and extension to the LCIM, a plan is in place to formulate the proposed improved interoperability model in a conference or journal paper for review and future publication.

Regarding the second research question, the use of a protocol-based approach presented in a conceptual multi-layered interoperability-solution design (see table 4.1 in Chapter 4) was proposed as one possible answer. In particular, the interoperability-solution design proposed that each individual element of a protocol (i.e. protocol messages and internal structures, discussed below) could provide interoperability at lower layers 2 to 4 (i.e. syntactic, semantic and functional layers). Interoperability at higher layers 5 and 6 could be provided by enabling two systems to reach a common awareness of capability and strategy through the realisation of the use of the protocol specification (combined use of internal structure and protocol messages). In terms of the highest layer – layer 7 (conceptual layer) – interoperability could only be partially supplied by using the service specification.

The concept of using a protocol-based approach was derived from reviewing existing understanding of how solutions could be constructed for similar research problems. To resolve interoperability challenges, approaches exist such as: selection of a singular set of mechanisms to be used in all instances (e.g. selection of only one authorisation mechanism); support of multiple languages used in different authorisation systems (e.g. an object-oriented framework used in Globus or plug-in modules used in TrustBuilder2 for supporting different policy and credentials languages); and identification of a protocol which can better facilitate interoperability (e.g. the ABAC-based authorisation protocol designed within the AAI).

From a review of the definition of the meaning of a protocol (see section 2.3), it was identified that researchers were agreed that a protocol was more than a combination of communication messages. More specifically, a protocol should comprise both a service specification (abstract information of a protocol) and a protocol specification (detailed information of a protocol). The protocol specification in turn should consist of protocol messages (communication messages) and internal structures (functionalities). This agreed comprehensive notion of a protocol became a contradiction to the concept of a protocol recognised in the LCIM, whereas a protocol could only provide syntactic interoperability. This is because the concept of the protocol understood in the design of the LCIM was only a combination of communication messages, which should be expressed as protocol messages as one component of the comprehensive notion of a protocol.

An improved TN protocol (see Chapter 4) was also presented in addition to the interoperability-solution design. The proposition of this TN protocol was that following step one of the interoperability-solution design, as a concrete protocol, it might provide interoperability at different layers between authorisation systems within Web Services. The main benefits of this protocol included that potential interoperability issues at layers 5 and 6 (capability and strategy) could be avoided through the novel design of the preparation stage, which could ensure that potential successful authorisation would not fail in some circumstances. In addition, interoperability at layers 2 to 4 (i.e. syntax, semantics and functionality) provided by this protocol was also taken into consideration.

The reason for the creation of this TN protocol was that without any specific utilisation of the interoperability-solution design, evaluation of its effectiveness might not be easily achievable. Relatively speaking, through solution development, an evaluation of the effectiveness of a specific protocol would be achievable with more ease. The TN protocol was therefore created as a specific example of the utilisation of the interoperability-solution design. In other words, the effectiveness of this specific protocol could demonstrate the effectiveness of the interoperability-solution design.

Evaluation of the effectiveness of the interoperability-solution design along with the protocol included (1) a protocol completeness test as one task of protocol verification, which identified that the protocol was not complete due to two intrinsic vulnerability issues (see Chapter 5), (2) a solution design aiming to address the vulnerability issues to complement the protocol completeness (see Chapter 6) and (3) an interoperability-solution evaluation including a protocol correctness test as another task of protocol verification (see Chapter 7).

According to step three of the protocol design and development methodology (see section 4.1), protocol verification needed to be conducted. There was a distinction between the purpose of protocol verification and that of protocol evaluation (see section 5.1). Through a review of the related work about protocol verification, a completeness test and a correctness test mainly needed to be conducted for achieving protocol verification. As there were various definitions of protocol correctness, a further review of related work was conducted. Eventually, a finding of the review indicated that the purpose of the protocol correctness test was to demonstrate whether it could achieve the pre-set goal (see section 5.1).

FSM (see section 5.2) was used for conducting the protocol completeness test. In fact, other methods as variations of FSM could have also been used as alternatives for this step, as their effects were quite similar to that of FSM. Due to this reason, the results achieved through the completeness testing might be the same, if any one of the other optional formal methods was used. In other words, there might be nuances between the results by using different formal methods for testing the completeness of the protocol.

Two potential vulnerability issues were explored in the protocol completeness test by using FSM. The first vulnerability issue was identified before the protocol design, but the second vulnerability issue was not identified until the conduction of the completeness test. In order to address the two issues, there was a need to understand the key reasons causing the occurrence of the vulnerability issues. The case study method was selected to help the researcher identify the core reasons underpinning the occurrence of the vulnerability issues. The reason for leveraging this method is explained in section 6.4.

Before designing a solution for the vulnerability issues, a detailed understanding of the policy-exchanged strategies involved in TN was developed (presented in section 3.3.1). Based on knowledge of the features of policy-exchanged strategies and the reasons for these resulting in an occurrence of the vulnerability issues, a solution design was proposed through the use of a relational database. Once the solution design was created, its effectiveness was evaluated through its application to the extra designed case studies, wherein the vulnerability issues might occur. The reason for using the case study evaluation was straightforward in that the effectiveness of the proposed solution could be directly demonstrated, if the vulnerability issues would not occur in the case studies with the use of the solution. The evaluation result showed that this solution design was effective, when the parsimonious strategy is used, which has been assessed as the most representative strategy of the policy-exchanged strategies category (see section 3.3.1).

The main weakness of the solution design was its inability to address the vulnerability issues of the protocol, when typical policy-exchanged strategies (i.e. PRUNES and DFANS) were used. As the unsuitability of the use of the PRUNES for TN and the deficiencies of the DFANS were critically assessed through the analysis of the relevant review in section 3.3.1, these issues are not resolved.

Reflecting on the process of evaluation of the solution design for the vulnerability issues, other research methods such as interviews and/or surveys based on questionnaires would not be as useful as the case study method. The reason was: as the occurrence of the RCRA had never been noticed, it would require the researcher to find a way to clarify what this issue was and in what circumstances this issue would occur. To explain its occurrence, the case study method would be the most effective way for the clarification of the issues and for the demonstration of the effectiveness of the solution design. Eventually, the case study method could be demonstrated as the most appropriate method used for this step than any other research methods.

Since the proposed solution design could address the vulnerability issues of the protocol, the protocol completeness proof was complemented. As stated in Chapter 5, the purpose of the protocol correctness test in this research was the same as the purpose of the protocol evaluation, which aimed to evaluate the effectiveness of the protocol (see section 5.1). In other words, the evaluation test aimed at proving whether the proposed protocol would be effective, which could in turn demonstrate whether the interoperability-solution design would be effective. With respect to the most appropriate research method for performing evaluation, the case study method was eventually selected (reasons are given in section 7.2).

With the use of a case study evaluation, a "case study protocol" (see section 7.3.1) was important to provide general guidance. In addition, a specific category of case studies should be selected. Eventually, an embedded single case design was selected (reasons are given in 7.3.2).

Within the embedded single case design, model-based testing was selected as a sub research method for conducting each embedded case study. The main reason for using this research method was ease of data collection (other reasons are explained in section 7.3.3). After conducting all of the sub case study, relevant data were collected for qualitative data analysis.

To ensure the validity of using the qualitative data analysis method, the researcher needed to follow a formal way. Fortunately, Yin (2013) provides four strategies and five techniques for conducting data analysis in the case study method. Each strategy and each technique is suitable to either a qualitative data analysis method or a quantitative data analysis method. After comparing the features of the research against them, the "relying on theoretical propositions" strategy and the "pattern matching" technique were eventually selected, as the situations for using them best matched the features of the research. Following the selected strategy and technique, the data analysis results proved that the protocol could ensure that potential successful authorisation would not fail in some circumstances. This in turn did demonstrate the effectiveness of the interoperability-solution design.

Reflecting on the process of the protocol evaluation process, the inappropriateness of using other research methods for performing the evaluation had been taken into consideration. For instance, questionnaires-based research methods such as interview or survey mainly required human participants, but the research object – authorisation systems, the research context – Web Services, and the proposed solution – protocol, did not require any human participants. Therefore, questionnaire-based research methods could not directly prove the effectiveness of the protocol, as human participants might not know what purpose the protocol could achieve. Without this knowledge, neither could the human participants assess the effectiveness of the protocol.

In terms of the sub research method used in the case study method, a real experiment along with implementation might be the best alternative method for the model-based testing, but the reason for not using it has been clarified in section 7.3.3.

In the preparation stage of the improved TN protocol, both keyword-based and semantic-based approaches are designed enabling two entities to agree on a common capability and a common strategy. Although, the use of a semantics-based approach can increase the matching accuracy, it is not always superior to the keyword-based approach, as its performance is slower than the keyword-based approach (Al-Safadi, Al-Dawood and Al-Abdullatif, 2010). The semantics-based approach is superior, when the information that needs to be matched may be referred to as different semantics, whereas the keyword-based approach is more suitable, if the information will have only one meaning through standardisation. For instance, X.509 certificates have been standardised, so Web Services that use this certificates shall have no ambiguous knowledge on its syntax and semantics. In such an instance, the keyword-based approach should be preferred, if the target name that needs to be matched is 'X.509'. In addition, as stated in Chapter 2, all of the protocols used in the context of Web Services should be standardised. Therefore, this standardisation should also apply to the proposed improved TN protocol as well as the terms of strategy names and names of credential languages and policy languages used in TN. From this perspective, the keyword-based approach used in the preparation stage should be powerful enough.

Analysed from the interoperability-solution evaluation result and the protocol correctness test, there were several limitations within either this protocol or the interoperability-solution design (see section 7.5). Firstly, within the protocol evaluation, all of the sub cases were assumed that the two participating Web Services would use the proposed TN protocol to communicate with each other. This assumption excluded two systems' ability to recognise the highest interoperability layer within the interoperability-solution design – layer 7 (conceptual interoperability). Unfortunately, this assumption may not be valid in practice, as it may not be always the case that two unknown systems will use the TN protocol to achieve authorisation, if they also support other authorisation protocols such as the existing ABAC-based authorisation protocol used in the AAI (Schlager et al., 2006).

To enable two systems to use this TN protocol requires two systems to intelligently understand the relevant factors suitable to the use of this protocol so as to make a decision of whether this protocol should be used. This intelligence closely relates to a conceptual understanding of a protocol. Within the current interoperability-solution design, interoperability at layer 7 can only be provided by the service specification of protocols, readers of which are mainly developers. In addition, it is assumed in this interoperability-solution design that ontology might be a potential solution for systems to have such intelligence. However, these propositions are not enough, as the interoperability-solution design does not identify the relevant factors for systems to make a decision of using a specific protocol. In addition, whether the use of ontology can enable systems to be conceptually intelligent is also a question to be answered. Therefore, assurance of the effectiveness of the propositions requires further research in exploring the relevant answers.

Secondly, extra functionalities have to be added as internal structures to the TN protocol (e.g. trust ticket, resumption of TN) to ensure that functionality interoperability can be provided by this protocol. As syntactic and semantic interoperability are closely related to functionality interoperability, relevant protocol messages will also be designed according to these internal structures. This limitation indicates that this TN protocol has to be updated periodically so as to keep its pace with state-of-the-art TN-based authorisation systems.

Thirdly, the preparation stage designed within the improved TN protocol could only ensure successful authorisation in some circumstances where two entities have a common strategy and capability of comparing credentials against policies. However, when two entities have no common strategy or capability, this protocol fails to deliver such interoperability for them. Eventually, potential successful authorisation has to fail due to this reason. This requires a potential solution (e.g. a third-party providing a language translation mechanism), which could be able to provide this strategy or capability interoperability for two entities. It would be ideal that such a potential solution could be integrated into this protocol and the interoperability-solution design.

Nevertheless, there is a benefit of the preparation stage in comparison to the potential third-party solution. Taking capability interoperability for example, if the potential third-party solution could only provide a language translation mechanism for providing capability interoperability, the communication cost of a three-entities communication might be much more than that of a two-entities communication by using the improved TN protocol. For instance, the exchange of credentials and policies between each entity and the third party for achieving translation would increase the potential communication cost of a three-entities involved TN authorisation communication. In the case that two entities have the common capability for achieving successful authorisation, the lower communication cost of a two-entity communications should always be preferred. However, if the potential third-party solution could provide extra functionalities (e.g. TN might only occur within the process of the third-party) besides the language translation functionality, where two entities only needed to submit all of the relevant credentials and policies once to the third party involved in TN, the communication cost might be decreased to an acceptable value. However, such a potential solution would require the third-party to possess far more capabilities than the two participating entities to ensure that the third-party would have the right capability that could translate the languages of the credentials and policies used by one entity to those used by the counterpart; otherwise, without this prerequisite, this third-party solution could not help to address the uncommon capability interoperability issue remained in this improved TN protocol. In addition, this potential solution might not be suitable to scenarios where privacy of sensitive credentials or policies are required, if owners are reluctant to upload them to this untrusted third party.

Fourthly, a proper performance test of this protocol becomes a further challenge due to complexity of TN. More precisely, as characteristics of each strategy are different, a fixed TN process cannot be anticipated in advance. In addition, any changes in a condition (see section 7.5.5) will even complicate the behaviours of each entity. Hence, exploring a performance test result that can represent TN behaviours may be a challenge.

Overall, the solutions proposed by this research do have their own limitations, but their main advantages were evaluated to be effective for addressing the research problems to some extent, although not completely. In addition, application of the solutions to other fields may embody their potential impacts, which are discussed in the next section.

## 8.3 Research Contributions and Impacts

In terms of the research contributions in this Thesis, as mentioned in the previous section, there are four of them.

The first contribution is an improved multi-layered interoperability model based on the LCIM. This model aims to clarify the existence of multiple layers of interoperability issues between authorisation systems (e.g. ABAC or TN) within Web Services. In particular, one novel interoperability layer in relation to strategy is identified in this model. Taking into account the similarities between interoperability issues within a distributed systems environment, it might help both researchers in academia and practitioners in the industry in the discovery of the interoperability issues between two communicating systems.

The second contribution is a conceptual multi-layered interoperability-solution design, which proposes how to use a protocol-based approach to provide interoperability for the majority of the interoperability layers. Existing approaches (e.g. ontologies, UML) for addressing specific interoperability issues are treated as elements of a protocol (e.g. internal structures). This model can provide guidance aiding protocol developers in addressing interoperability issues, if the relevant interoperability issues between communicating systems have been identified by using the above interoperability model.

The third contribution is an improved TN protocol, which is used as a detailed example of utilising the interoperability-solution design guiding the design of this protocol. This protocol can aid in communication between authorisation systems without specific interoperability issues (e.g. functionality, strategy) within Web Services. In addition, it can enable potentially successful TN-based authorisation to avoid failure, if initially designed multiple languages for expressing credentials and policies of different authorisation systems are used in the context of Web Services. In other words, it can raise the success probability for using TN in some circumstances. These benefits mainly ascribe to the design of the novel preparation stage of the protocol. This preparation stage could be used in other potential new protocols or other TN protocols used in non-Web Services contexts, wherein two unknown entities need to establish a bilateral trust relationship.

With respect to the circumstances of the application of this protocol, its utilisation is most suitable to the circumstances where two Web Services are unknown to each other. In addition, each of the two Web Services owns sensitive information that cannot be disclosed, unless a certain level of trust has been established. In other words, the identity of each other is not stored in their local databases. In this instance, PKI may be used to make up for this weakness, but the existence of multiple TTPs cannot ensure token interoperability. In the particular instance where token interoperability cannot be ensured, an RBAC approach may not be available. In addition, RBAC is not powerful enough, when an access control decision needs to be made based on additional attribute information. Due to the existence of local sensitive information, direct use of an ABAC approach may miss the opportunity of reaching successful authorisation. In such an instance, the TN protocol proposed in this Thesis should be the most appropriate method used by two unknown Web Services.

One notable point of harnessing this protocol is that both Web Services should be designed to allow their human users to design their own policies. However, observing how existing Web Services are used in practice such as Hotmail, granting such permission to human users is still not available. Instead, different options of policies are predefined in Web Services, and human users have no right to design their own policies, so that human users can only choose to use one of the policies predefined in the Web Services. To utilise the TN protocol, Web Services need to support the functionality that human users can design their own policies. A good starting point to try the human-users-defined policies approach is perhaps to design and develop Web Services for an E-Learning environment across different universities as illustrated in the case scenario presented in Chapter 2.

Observing the purpose of the TN protocol presented in Chapter 4, this protocol mainly serves the bilateral trust establishment between two unknown Web Services in their first-time communication, but there is no discussion about its application to scenarios where two Web Services need to achieve authorisation in their n-time communication (n is greater than 1 and n is a natural number). Through the literature review in Chapter 2, it can be identified that the existing authorisation approaches (i.e. RBAC or ABAC) can be used instead, if the two Web Services are known to each other by using the TN protocol during their first-time communication. Following the internal structure of the preparation stage, the common language combination has been stored in the local database of both entities. Therefore, capability interoperability (i.e. language combination) will not be an issue after their first-time communication. In conclusion, the combination of the use of this TN protocol and the existing authorisation approaches together should be suitable to n-time communication instances.

The fourth contribution is a solution design aiming to address intrinsic vulnerability issues within the proposed TN protocol. It can also be utilised to protocols designed within state-of-the-art TN-based authorisation systems, when some policy-exchanged strategies are used. In addition, this solution might also be applied to resolve variations of DoS attacks.

Upon discussing the contributions along with their impacts of the research, the next section discusses potential future work derived from the limitations of the proposed solutions.

## 8.4 Future Work

One interesting thing of carrying out a research is that a researcher may explore new challenges from the evaluation process of the current research contributions. These challenges cannot be predicted by the researcher at the starting point of the research, but can be discovered from the evaluation result of the research. The following sub sections discuss the most relevant future work.

### 8.4.1 Exploring factors relevant to conceptual interoperability

To enable systems to communicate without any conceptual interoperability issues, there is a need to explore possible key factors aiding systems in making a decision of using a specific protocol. If they can be explored, identification of relevant solutions will also be needed to enable systems to intelligently utilise these factors. In addition, integrating these key factors into the current interoperability-solution design may be another issue, as achieving this purpose may require the expansion of the current interoperability-solution design with proper modifications.

### 8.4.2 Uncommon strategy and capability issue

Two communicating systems without a common strategy or capability for comparing credentials against policies will miss an opportunity of reaching successful authorisation by using the proposed TN protocol. Within this protocol, taking capability interoperability as an example, language interpretation functionality and a policy compliance checker designed based on an object-oriented design for comparing local policies (credentials) against received remote credentials (policies) inbuilt in the two systems are treated as the capability owned by the systems (see section 4.5). That is, scenarios where no common capability is between two systems imply that there is no way to address the capability interoperability issue through direct communication between two systems. Therefore, a third-party providing a language interpretation mechanism might the best solution for these scenarios. To verify whether this proposition is useful requires further research.

### 8.4.3 Performance test challenge

A performance test that can represent general TN performance is much more challengeable through the data analysis of the protocol evaluation result. It has been identified that multiple conditions as factors are relevant to a TN process, and any change of these conditions may change the length of a TN process. This flexibility indicates that exploration of a representative TN performance test result might require a number of experiments. Exploration of these experiments designs requires further research.

### 8.4.4 Security consideration

In terms of the security consideration of the protocol, one of the most notable vulnerabilities was discovered after reflection on the protocol completeness tests process. The two innate vulnerability issues were discovered within this process. In particular, the second vulnerability - RCRA was discovered, when the researcher realised that the context of using the protocol could be broadened, where malicious Web Services could exist. RCRA existed, if a malicious Web Service utilised a certain design flaw within the protocol.

As RCRA was identified as a variation of DoS attacks, by extending the feature of RCRA, a new vulnerability issue could also be identified in the protocol. In fact, this vulnerability issue exists in general protocols. The attacks in relation to this vulnerability issue is called Deviation from protocol message sequence, as one category of DoS attacks, stated in Gruschka, Jensen and Luttenberger, (2007), 'attacks use message with correct message structure but sent in a sequence deviating from the protocol definition'. This attack will lead the internal structures of the protocol used by a Web Service Requester into unknown states. The reason that this phenomenon may occur is straightforward. The order of message exchanges defined in this protocol is fixed, and the information created in former steps will be used in later steps. For instance, when a <TNPrepareRequest> message has been processed, the common strategy name and the common language combination names will be stored in the database. The information will be used again, when an <AuthzDecisionQuery> message is being processed (see the preparation stage in section 4.5).

To enable a Web Service to defend against the attacks, Gruschka, Jensen and Luttenberger (2007) suggested that protocols defining the fixed order of message exchanges should be able to detect this kind of attacks, and deny processing the messages. They proposed a model called Successor Set Automaton (SSA) with the use of the Business Process Execution Language (referred to as BPEL, a Web Service standard for business processing modelling) in a firewall to enforce the message sequence exchange of a protocol. In their solution, a local Web Service will have a list of local states. Each local state is related to a specific protocol message of a protocol, and is used as a switch declaring whether or not a specific incoming message can be processed. The change of each state is decided based on the last received incoming message. However, this solution is not strong enough, as there is no concern about the relationship between the local states and outgoing messages. The lack of this relationship presents limitations when this solution is applied on the protocol proposed in this Thesis. For example, in the negotiation stage, after a Web Service Provider sends out an <AuthzDecisionStatement> message to the counterpart, it will not expect to receive any further messages from the counterpart. As the <AuthzDecisionStatement> message implicitly indicates that this is the last message used in the protocol, so no further incoming messages should be received from the perspective of the Web Service Provider. This example can demonstrate that the relationship between local states and local outgoing messages should also be taken in to consideration in a potential solution, which can effectively defend against deviation from protocol message sequence attacks. Exploration of an approach implementing this hypoThesis is treated as future work.

## 8.5 Chapter Summary

This Chapter has discussed the process of the whole research. Four research contributions along with their impacts are also presented. Through the identification of limitations of solutions, potential future work is also listed, which can guide the researcher in conducting further research.

# References

## A:

Aarts, R. Beatty, J. Cahill, C. Serret, X. & Whitehead, G., 2003. *Liberty ID-FF Protocols and Schema Specification*, Available at: http://www.projectliberty.org/liberty/content/download/2197/14625/file/draft-liberty-idff-protocols-schema-1.2-errata-v3.0.pdf. [Accessed 3rd April 2012].

Adams, C. & Lloyd, S., 1999. *Understanding the Public-key Infrastructures: Concepts, Standards, Deployment Considerations*. Available at: http://technet.microsoft.com/en-us/library/cc700808.aspx. [Accessed 21st March 2010].

Alfieri, R. Cecchini, R. Ciaschini, V. dell'Agnello, L. Frohner, Á. Gianoli, A. Lõrentey, K. & Spataro, F., 2004. VOMS, an Authorization System for Virtual Organizations. *Grid Computing Lecture Notes in Computer Science*, 2970, pp.33-40.

Al-Safadi, L. A. Al-Dawood, A. A. & Al-Abdullatif, N., 2010. Lexeme: An Ontology-Based Semantic Advertising Networks. *Journal of Computing*, 2(10), pp.71-75.

Alonso, J. M. Ambur, O. Amutio, M. A. Azanon, O. Bennett, D. Flagg, R. McAllister, D. Novak, K. Rush, S. & Sheridan, J., 2009. *Improving Access to Government through Better Use of the Web*. Available at: http://www.w3.org/TR/egov-improving. [Accessed 10th February 2010].

Andrieux, A. Czajkowski, K. Dan, A. Keahey, K. Ludwig, H. Nakata, T. Pruyne, J. Rofrano, J. Tuecke, S. & Xu, M., 2007. *Web Services Agreement Specification (WS-Agreement)*, Available at: http://www.ogf.org/documents/GFD.107.pdf. [Accessed 15 March 2010].

Andro, G., 2010. Automated Trust Negotiation Models. In *2010 Proceedings of the 33rd International Convention*. Opatija, Croatia. pp.1197-1202.

Ankolekar, A. Burstein, M. Hobbs, O. Martin, D. McDermott, D. Mcllraith, S. A. Narayanan, S. Paolucci, M. Payne, T. & Sycara, K., 2002. DAML-S: Web Service Description for the Semantic Web. In *Proceedings of the First International Semantic Web Conference on the Semantic Web (ISWC '02)*. Sardinia, Italia. pp.348-363.

ATHENA, 2005. *Enterprise Interoperability Maturity Model (EIMM)*. pp.1-11. Available at: http://athena.modelbased.net/methodology/eimm.pdf. [Accessed 18 March 2013].

## B:

Ballinger, K. Ehnebuske, D. Gudgin, M. Nottingham, M. & Yendluri, P., 2004. *WS-I Basic Profile Version 1.0*. Available at: http://www.ws-i.org/profiles/BasicProfile-1.0-2004-04-16.html. [Accessed 27 August 2010].

Barkley, J. F. Cincotta, A. V. Ferraiolo, D. F. Gavrilla, S. & Kuhn, D.R., 1997. Role Based Access Control for the World Wide Web. In *20th National Information System Security Conference.* Baltimore, Maryland. pp.1-11.

Barlow, T. Hess, A. & Seamons, K.E., 2001. Trust Negotiation in Electronic Markets. In *Proceedings of the Eighth Research Symposium on Emerging Electronic Markets*. Masstricht, The Netherlands. pp.1-13.

Bartel, M. Boyer, J. Fox, B. LaMacchia, B. & Simon, E., 2008. *XML Signature Syntax and Processing. (Second Edition)*. Available at: http://www.w3.org/TR/xmldsig-core. [Accessed 19th March 2010].

Baselice, S. Bonatti, P. A. & Faella, M., 2007. On Interoperable Trust Negotiation Strategies. In *Eighth IEEE International Workshop on Policies for Distributed Systems and Networks*. Bologna, Italy. pp.39-50.

Basney, J. Nejdl, W. Olmedilla, D. Welch, V. & Winslett, M., 2005. *Negotiating Trust on the Grid*. Available at: http://drops.dagstuhl.de/opus/volltexte/2005/387. [Accessed 6th May 2010].

Baum, L. E. & Petrie, T., 1966. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6), pp.1554-1563.

Baxter, P. & Jack, S., 2008. Qualitative Case Study Methodology : Study Design and Implementation for Novice Researchers. *The Qualitative Report*, 13(4), pp.544-559.

Bellwood, T. Capell, S. Clement, L. Colgrave, J. Dovey, M. J. Feygin, D. Hately, A. Kochman, R. Macias, P. Novotny, M. Paolucci, M. Riegen C. Rogers, T. Sycara, K. Wenzel, P. & Wu, Z., 2004. *UDDI Version 3.0.2*. Available at: http://uddi.org/pubs/uddi_v3.htm. [Accessed 5th May 2010].

Benatallah, B. Casati, F. Grigori, D. Nezhad, H. & Toumani, F., 2005. Developing Adapters for Web Services Integration. In *Proceedings of the 17th International Conference on Advanced Information Systems Engineering (CAiSE'05)*. Porto, Portugal. pp.415-429.

Benbasat, I. Goldstein, D. & Mead, M., 1987. The Case Research Strategy in Studies of Information Systems. *Journal of Mis Quarterly*, 11(3), pp.369-386.

Bertino, E. Ferrari, E. & Squicciarini, A.C., 2003a. On Specifying Security Policies for Web Documents with An XML-based Language. In *Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies*. Chantilly, VA, USA. pp.57-65.

Bertino, E. Ferrari, E. & Squicciarini, A. C., 2003b. X-TNL: An XML-based Language for Trust Negotiations. In *Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'03)*. Lake Como, Italy. pp.81-84.

Bertino, E. Ferrari, E. and Squicciarini, A. C., 2004a. Trust-X: A Peer-to-Peer Framework for Trust Establishment. *IEEE Transactions on Knowledge and Data Engineering*. 16(7) pp.827-842.

Bertino, E. Ferrari, E. & Squicciarini, A.C., 2004b. Trust Negotiations: Concepts, Systems, and Languages. *Computing in Science and Engineering*, 6(4), pp.27-34.

Bertino, E. Ferrari, E. & Squicciarini, A.C., 2005. Privacy-Preserving Trust Negotiations. *Lecture Notes In Computer Science*, 3424(2005), pp.283-301.

Beznosov, K. Flinn, D. J. Kawamoto, S. & Hartman, B., 2005. Introduction to Web Services and their Security. *Information Security Tech. Report*, 10(1), pp.2-14.

Bhargavan, K. Obradovic, D. & Gunter, C., 2002. Formal Verification of Standards for Distance Vector Routing Protocols. *Journal of the ACM*, 49(4), pp.538-576.

Bhatti, R. Joshi, J. B. D. Bertino, E. Ghafoor, A., 2003. Access Control in Dynamic XML-based Web-Services with X-RBAC. In *International Conference on Web Services (ICWS '03)*. Las Vegas, Nevada, USA. pp.243-249.

Bhatti, R. Bertino, E. & Ghafoor, A., 2004. A Trust-based Context-Aware Access Control Model for Web-Services. In *Proceedings of the IEEE International Conference on Web Services (ICWS '04)*. San Diego, California, USA. pp.184-191.

Bochmann, G. V. & Gecsei, J., 1977. A Unified Method for the Specification and Verification of Protocols. In *Proc. IFIP Congress*. North Holland. pp.229–234.

Bochmann, G. V., 1978. Finite State Descriptions of Communication Protocols. *Computer Networks*, 2, pp.361-372.

Bochmann, G. & Sunshine, C., 1980. Formal Methods in Communication Protocol Design. *IEEE Transactions on Communications*, 28(4), pp.624-631.

Bonatti, P. A. Coi, J. L. Olmedilla, D. & Sauro, L., 2010. A Rule-based Trust Negotiation System. *IEEE Transactions on Knowledge and Data Engineering*, 22(11), pp.1507-1520.

Boncella, R.J., 2004. Web Services and Web Services Security. *Communications of Association for Information Systems*, 14, pp.344-363.

Booth, D. Haas, H. McCabe, F. Newcomer, E. Champion, M. Ferris, C. Orchard, D., 2004. *Web Services Architecture*. Available at: http://www.w3.org/TR/ws-arch. [Accessed 11th May 2010].

Bray, T. Paoli, J. & Sperberg-McQueen, C.M., 1998. *Extensible Markup Language (XML) 1.0*. Available at: http://www.w3.org/TR/1998/REC-xml-19980210. [Accessed 9 February 2010].

Burnett, M. A. and Kleiman, D., 2005. *Perfect Password: Selection, Protection, Authentication*, MA: Syngress Publishing, Inc.

## C:

CGI, 2004. *Public Key Encryption and Digital Signature: How Do They Work?* Available at: http://www.cgi.com/files/white-papers/cgi_whpr_35_pki_e.pdf. [Accessed 21st June 2010].

Chadwick, D. W. & Otenko, A., 2002. The PERMIS X.509 Role Based Privilege Management Infrastructure. In *Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies (SACMAT '02)*. Monterey, CA, USA. pp.135-140.

Channabasavaiah, K. Tuggle, E. & Holley, K., 2003. *Migrating to A Service-Oriented Architecture, Part 1*. Available at: ftp://ftp.software.ibm.com/software/info/.../G224-7298-00_Final.pdf. [Accessed 26th June 2011].

Chen, W. & Jiang, W., 2011. Analysis and Design of An Adaptive Automated Trust Negotiation System. In *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*. Jilin, China. pp.2320-2325.

Chevalier, Y. & Vigneron, L., 2002. Automated Unbounded Verification of Security Protocols. In *Proceedings of 14th International Conference on Computer Aided Verification (CAV '02)*. Copenhagen, Denmark. pp.324-337.

Chkliaev, D. Hooman, J. & Stok, P., 2000. Serializability Preserving Extensions of Concurrency Control Protocols. In *Proceedings of Third International Andrei Ershov Memorial Conference*. Akademgorodok, Novosibirsk, Russia. pp.180-193.

Christensen, E. Curbera, F. Meredith, G. & Weerawarana, S., 2001. *Web Services Description Language (WSDL) 1.1*. Available at: http://www.w3.org/TR/wsdl. [Accessed 3rd June 2011].

Chumbley, R. Durand, J. Pilz, G. & Rutt, T. (eds.), 2010. *WSI Basic Profile Version 2.0*. Available at: http://ws-i.org/profiles/basicprofile-2.0-2010-11-09.html. [Accessed 10 July 2010].

Clark, T. & Jones, R., 1999. *Organisational Interoperability Maturity Model for C2*. Available at: www.sei.cmu.edu/library/assets/orginteroper.pdf. [Accessed 5th May 2012].

Curbera, F. Duftler, M. & Khalaf, R. & Nagy, W., 2002. Unravelling the Communication: SOAP. *IEEE Internet Computing*, 6(2), pp.86-93.

Curbera, F. Nagy, W. A. & Weerawarana, S., 2001. *Web Services: Why and How.* Available at: http://lsdis.cs.uga.edu/courses/8351Spring2006/papers2Read/wsWhyandHow.pdf. [Accessed 10 February 2010].

Curphey, M., 2005. Web Services: Developers Dream or Hackers Heaven? *Information Security Technical Report*, 10(4), pp.228-235.

## D:

Daigneau, R. & Robinson, I., 2011. Service Design Patterns: *Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services*, New Jersey: Addison-Wesley Professional.

Danthine, A., 1980. Protocol Representation with Finite-State Models. *IEEE Transactions on Communications*, 28(4), pp.632-643.

Danthine, A. & Bremer, J., 1978. Modelling and Verification of End-to-End Transport Protocols. *Computer Networks*, 2(4-5), pp.381-395.

Davis, C. H. & Vladica, F., 2007. *E-Commerce and V-Business: Digital Enterprise in the Twenty-First Century (2nd edi.).* Oxford: Butterworth-Heinemann.

Diallo, S.Y., 2010. *Towards A Formal Theory of Interoperability*. Old Dominion University. Norfolk, Virginia, USA. pp.1-81.

Debbabi, M., 2004. Towards the Correctness of Security Protocols. *Electronic Notes in Theoretical Computer Science*, 83, pp.1-46.

DoD, 1998. *Levels of Information Systems Interoperability (LISI)*. Available at: www.eng.auburn.edu/~hamilton/security/DODAF/LISI.pdf. [Accessed 2nd March 2012].

Doghmi, S. Guttman, J. & Thayer, F., 2007. Searching for Shapes in Cryptographic Protocols. In *Proceedings of 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS '07)*. Braga, Portugal. pp.523-537.

## E

EI-Bakry, H. M. & Mastorakis, N., 2009. Studying the Efficiency of XML Web Services for Real-time Applications. In *Proceedings of the 2nd WSEAS International Conference on Sensors, and Signals and Visualization, Imaging and Simulation and*

*Materials Science (SENSIG'09/VIS'09/MATERIALS'09)*. Stevens Point, Wisconsin, USA. pp.209-219.

Emig, C. Brandt, F. Abeck, S. Biermann, J. & Klarl, H., 2007. An Access Control Metamodel for Web Service-Oriented Architecture. In *Proceedings of the International Conference on Software Engineering Advances (ICSEA '07)*. Cap Esterel, French Riviera, France. pp.57.

Erber, R. Schläger, C. & Pernul, G., 2007. Patterns for Authentication and Authorisation Infrastructures. In *18th International Workshop on Database and Expert Systems Applications (DEXA '07)*. Valencia, Spain. pp.755-759.

## F:

Fabrega, F. J. T. Herzog, J.C.& & Guttman, J.D., 1998. Strand Spaces: Why Is A Security Protocol Correct? In *Proceedings of 1998 IEEE Symposium on Security and Privacy*. Oakland, Califoria, USA. pp.60-171.

Fielding, R. Gettys, J. Mogul, J. Frystyk, H. Masinter, L. Leach, P. & Berners-Lee, T., 1999. *Hypertext Transfer Protocol – HTTP/1.1*. Available at: http://www.ietf.org/rfc/rfc2616.txt. [Accessed 15th July 2010].

Feigenbaum, J., 1998. *Towards An Infrastructure for Authorization*. Available at: https://www.usenix.org/event/ec98/pki/feigenbaum.pdf. [Accessed 13th June 2013].

Fensel, D. & Bussler, C., 2002. The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, 1(2), pp.113-137.

Ferraiolo, D. & Kuhn, R., 1992. Role-Based Access Controls. In *15th National Computer Security Conference*. Baltimore, Maryland. pp.554-563.

Ferraiolo R. Sandhu, R. Gavrila, S. Kuhn D. R. & Chandramouli, R., 2001. Proposed NIST Standard for Role-based Access Control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3), pp.224-274.

Foster, I. Kesselman, C. & Tuecke, S., 2001. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15(3), pp.200-222.

Foster, I., 2006. Globus toolkit version 4: Software for Service-oriented Systems. *Journal of Computer Science and Technology*, 21(4), pp.513-520.

Frikken, K. B. Li, J. & Atallah, M.J., 2006. Trust Negotiation with Hidden Credentials, Hidden Policies, and Policy Cycles. In *Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, California, USA. pp.157-172.

## G:

Garzoglio, G. Alderman, I. Altunay, M. Ananthakrishnan, R. Bester, J. Chadwick, K. Ciaschini, V. Demchenko, Y. Ferraro, A. Forti, A. Groep, D. Hesselroth, T. Hover, J. Koeroo, O. Joie, C. Levshina, T. Miller, Z. Packard, J. Sagehaug, H. Sergeev, V. Sfiligoi, I. Sharma, N. Siebenlist, F. Venturi, V. & Weigand, J., 2009. Definition and Implementation of a SAML-XACML Profile for Authorization Interoperability Across Grid Middleware in OSG and EGEE. *Journal of grid computing*, 7(3), pp.297-307.

Gavriloaie, R. Nejdl, W. Olmedilla, D. Seamons, K. E. & Winslett, M., 2004. No Registration Needed: How to Use Declarative Policies and Negotiation to Access Sensitive Resources on the Semantic Web. In *Proceedings of* the *1st European Semantic Web Symposium*. Heraklion, Crete, Greece. pp.342-356.

Glass, G., 2001. *Web Services: Building Blocks for Distributed Systems*, Delhi: Prentice Hall Ptr.

Globalsign, 2013. *About GlobalSign*. Available at: https://www.globalsign.com/company/. [Accessed 15th February 2014].

Garofalakis, J. Panagis, Y. Sakkopoulos, E. & Tsakalidis, A., 2006. Web Service Discovery Mechanisms: Looking for a Needle in a Haystack? Available at: http://mmlab.ceid.upatras.gr/people/sakkopoulou/conf/ht04.pdf. [Accessed 1st July 2010].

Geer, D., 2003. Taking Steps to Secure Web Services. *Computer*, 36(10), pp.14-16.

Gouda, M. G. & Manning, E.G., 1976. On the Modelling, Analysis and Design of Protocols - A Special Class of Software Structures. In *Proceedings of the 2nd International Conference on Software Engineering (ICSE'76)*. Los Alamitos, CA. pp.256-262.

Gruschka, N. Jensen, M. & Luttenberger, N., 2007. A Stateful Web Service Firewall for BPEL. In *IEEE International Conference on Web Services (ICWS 2007)*. Salt Lake City, UT. pp.142-149.

Gudgin, M. Hadley, M. Rogers, T. (eds.), 2006. *Web Services Addressing 1.0 - Core*. Available at: http://www.w3.org/TR/ws-addr-core/. [Accessed 15th June 2010].

Guo, S. & Jiang, W., 2010. An Adaptive Automated Trust Negotiation Model and Algorithm. In *2010 International Conference on Communications and Intelligence Information Security (ICCIIS)*. Nanning, China. pp.130-134.

Guruge, A., 2004. *Web Services: Theory and Practice*, Oxford: Digital Press.

## H:

Halevey, A., 2005. Why Your Data Won't Mix. *Queue - Semi-structured Data*, 3(8), pp.50-58.

Hamilton, J. A. Rosen, C. & Summers, M., 2002. *An Interoperability Road Map for C4ISR Legacy Systems*, Available at: www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA487874. [Accessed 5th August 2010].

Hess, A. Holt, J. Jacobson, J. & Seamons, K.E., 2004. Content-Triggered Trust Negotiation. *ACM Transactions on Information and System Security (TISSEC)*, 7(3), pp.428-456.

Hitzler, P. Krotzsch, M. Parsia, B. Patel-Schneider, P. F. & Rudolph, S. (eds.), 2012. *OWL 2 Web Ontology Language Primer (Second Edition)*. Available at: http://www.w3.org/TR/2012/REC-owl2-primer-20121211/. [Accessed 8 March 2011].

Ho, R. & Yen, Y., 2005. Design and Evaluation of An XML-based Platform-independent Computerized Adaptive Testing System. *IEEE Transactions on Education*, 48(2), pp.230-237.

Holt, J. E. Bradshaw, R. W. Seamons, K. E. & Orman, H., 2003. Hidden Credentials. In *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*. Scottsdale, Arizona, USA. pp.9-20.

Housley, R., Polk, W. Ford, W. & Solo, D., 2008. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. Available at: http://tools.ietf.org/html/rfc5280. [Accessed 25th May 2010].

Howard, R., 2001. *Encrypting SOAP Messages*. Available at: http://msdn.microsoft.com/en-us/library/ms972410.aspx. [Accessed 5th April 2010].

Huang, C. & Hsu, J., 1994. An Incremental Protocol Verification Method. *The Computer Journal*, 37(8), pp.698-710.

## I:

IBM & Microsoft 2002. *Security in A Web Services World: A Proposed Architecture and Roadmap*. Available at: http://msdn.microsoft.com/en-us/library/ms977312.aspx. [Accessed 4th April 2010].

IEEE, 2011. *IEEE Guide — Adoption of the Project Management Institute (PMI ®) Standard A Guide to the Project Management Body of Knowledge (PMBOK ® Guide) — Fourth Edition IEEE Computer Society*. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6086685. [Accessed 9th July 2011].

IEEE Press, 2000. *IEEE 100 The Authoritative Dictionary of IEEE Standards Terms (7th edition)*, Institute of Electrical and Electronics Engineers (IEEE). Available at: http://ieeexplore.ieee.org/servlet/opac?punumber=4116785. [Accessed 10th June 2011].

Imamura, T. Dillaway, B. & Simon, E., 2002. *XML Encryption Syntax and Processing*. Available at: http://www.w3.org/TR/xmlenc-core/. [Accessed 19th August 2010].

## J:

Jamroga, W. Melissen, M. & Schnoor, H., 2014. On Defendability of Security Properties. In *Proceedings 2nd International Workshop on Strategic Reasoning*. Grenoble, France. pp.17-25.

Jonker, J. & Pennink, B., 2010. *The Essence of Research Methodology*, London: Springer-Verlag Berlin Heidelberg.

Johnston, W. Mudumbai, S. & Thompson, M., 1998. Authorization and Attribute Certificates for Widely Distributed Access Control. In *Proceedings of the 7th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '98)*. Stanford, California, USA. pp.340-345.

Josefsson, S., 2006. *The Base16, Base32, and Base64 Data Encodings*. Available at: http://tools.ietf.org/html/rfc4648. [Accessed 21st May 2011].

Josuttis, N. M., 2007. *SOA in Practice: The Art of Distributed System Design (Theory in Practice)*, CA: O'Reilly Media Inc.

## K

Karp, A.H., 2006. Authorization-Based Access Control for the Services Oriented Architecture. In *the Fourth International Conference on Creating, Connecting and Collaborating through Computing, (C5 '06)*. Berkeley, California, USA. pp.160-167.

Kitchenham, B. Pfleeger, S. Pickard, L. Jones, P. Hoaglin, D. El, E. & Rosenberg, J., 2002. Preliminary Guidelines for Empirical Research in Software Engineering. *IEEE Transactions on Software Engineering*, 28(2), pp.721-734.

Klein, M., 2001. XML, RDF and Relatives. *IEEE Intelligent Systems*, 16(2), pp.26-28.

Klein, H. K. & Myers, M. D., 1999. A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. *MIS Quarterly - Special Issue on Intensive Research in Information Systems*, 23(1), pp.67-93.

Klyne, G. & Carroll, J.J., 2004. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. Available at: http://www.w3.org/TR/rdf-concepts/. [Accessed 15th May 2012].

Koshutanski, H. & Massacci, F., 2005. Interactive Credential Negotiation for Stateful Business Processes. *Lecture Notes In Computer Science*, 3477(2005), pp.256-272.

Kull, A., 2009. *Model-Based Testing of Reactive Systems*. Tallinn University of Technology. Tallinn, Estonia.

Kumar, C. R., 2008. *Research Methodology*, New Delhi: APH Publishing Corporation.

Kothari, C. R., 2009. *Research Methodology Methods and Techniques*, New Delhi: New Age International (P) Ltd.

## L:

Lampson, B.W., 1969. Dynamic Protection Structures. In *Proceedings of the Fall Joint Computer Conference (AFIPS '69 Fall)*. Las Vegas, Nevada, USA. pp.27-38.

Lang, B. Foster, I. Siebenlist, F. Ananthakrishnan, R. & Freeman, T., 2006. A Multipolicy Authorization Framework for Grid Security. In the *Fifth IEEE International Symposium on Network Computing and Applications*. Cambridge, Massachusetts, USA. pp.267-272.

Lawrence, K. & Kaler, C., 2004. *Web Services Security: SOAP Message Security 1.1*. Available at: http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf. [Accessed 17th February 2010].

Lawrence, K. & Kaler, C., 2009a. *WS-Trust 1.4*. Available at: http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/cd/ws-trust-1.4-spec-cs-01.html. [Accessed 20th March 2010].

Lawrence, K. & Kaler, C., 2009b. *WS-SecureConversation 1.4*. Available at: http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.4/cd/ws-secureconversation-1.4-spec-cs-01.html. [Accessed 21st March 2010].

Lawrence, K. & Kaler, C., 2009c. *WS-SecurityPolicy 1.3*. Available at: http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.3/os/ws-securitypolicy-1.3-spec-os.html. [Accessed 18th March 2010].

Lee, A.S., 1989. A Scientific Methodology for MIS Case Studies. *MIS Quarterly*, 13(1), p.33.

Lee, A. J. Winslett, M. Basney, J. & Welch, V., 2006. Traust: A Trust Negotiation-Based Authorization Service for Open Systems. In *Proceedings of the Eleventh ACM Symposium on Access*. Lake Tahoe, California, USA. pp.39-48.

Lee, A. J. & Winslett, M., 2008c. Towards Standards-Compliant Trust Negotiation for Web Services. *IFIP International Federation for Information Processing*, 263, pp.311-326.

Lee, A. J. Winslett, M. & Perano, K.J., 2009. TrustBuilder2: A Reconfigurable Framework for Trust Negotiation. *IFIP Advances in Information and Communication Technology*, 300, pp.176-195.

Lewis, G. & Wrage, L., 2006. *Model Problems in Technologies for Interoperability : Web Services*, Available at: http://resources.sei.cmu.edu/asset_files/TechnicalNote/2006_004_001_14705.pdf. [Accessed 3rd October 2010].

Li, N. Du, W. & Boneh, D., 2003. Oblivious Signature-based Envelope. *Distributed Computing*, 17(4), pp.293-302.

Li, J. Li, b. & Meng, L., 2010. HiTrust: A Hybrid Tree Based Trust Negotiation Service. In *IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA 2010)*. Perth, Australia. pp.854-859.

Li, J. Li, N. & Winsborough, W.H., 2005. Automated Trust Negotiation Using Cryptographic Credentials. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*. Alexandria, VA, USA. pp.46-57.

Linn, R. J. & JR., 1989. Conformance Evaluation Methodology and Protocol Testing. *IEEE Journal on Selected Areas in Communications*, 7(7), pp.1143-1158.

Liu, P. & Chen, Z., 2004. An Access Control Model for Web Services in Business Process. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI '04)*. Beijing, China. pp.292-298.

Liu, H. Pallickara, S. & Fox, G., *Performance of Web Services Security*, Indiana, USA. Indiana University.

Liu, X. Tang, S. Huang, Q. & Yu, Z., 2013. An Ontology-based Approach to Automated Trust Negotiation. *Computer Standards & Interfaces*, 36(1), pp.219-230.

Lu, h. & Liu, B., 2009. DFANS: A Highly Efficient Strategy for Automated Trust Negotiation. *Computers & Security*, 28(7), pp.557-565.

Locke, J., 2004. *Open Source Solutions for Small Business Problems*, Hingham, Massachusetts: Charles River Media.

Lopez, J. Oppliger, R. & Pernul, G., 2004. Authentication and Authorization Infrastructures (AAIs): A Comparative Survey. *Computers & Security*, 23(7), pp.578-590.

## M:

Mahmoud, Q. H., 2005. *Securing Web Services and the Java WSDP 1.5 XWS-Security Framework*. Available at: http://www.oracle.com/technetwork/articles/java/security-140160.html. [Accessed 10th May 2010].

Mallalieu, T. & Carriere, J., 2004. *Enterprise Interoperability: .NET and J2EE*. Available at: http://msdn.microsoft.com/en-us/library/ms954598.aspx. [Accessed 24 August 2010].

Manes, A.T., 2003. *Web Services: A Manager's Guide*, Boston, MA: Addison Wesley Professional.

Matsuo, S. Miyazaki, K. Otsuka, A. & Basin, D., 2010. How to Evaluate the Security of Real-life Cryptographic Protocols ? *Financial Cryptography and Data Security*, 6054, pp.182-194.

Mbanaso, U. M. Cooper, G. S. Chadwich, D. W. & Proctor, S., 2006. Privacy Preserving Trust Authorization Framework Using XACML. In *Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks*. Buffalo-Niagara Falls, New York. pp.673-678.

Merlin, P.M., 1976. A Methodology for the Design and Implementation of Communication Protocols. *IEEE Transactions on Communications*, 24(6), pp.614-621.

Merlin, P., 1979. Specification and Validation of Protocols. *IEEE Transactions on Communications*, 27(11), pp.1671-1680.

Mewar, V. S. Aich, S. & Sural, S., 2007. Access Control Model for Web Services with Attribute Disclosure Restriction. In *Proceedings of the Second International Conference on Availability, Reliability and Security (ARES '07)*. Vienna, Austria. pp.524-531.

Microsoft, 2004. *Application Interoperability: Microsoft® .NET and J2EE*, O'Reilly Media, Inc.

Millen, J. & Shmatikov, V., 2001. Constraint Solving for Bounded-Process Cryptographic Protocol Analysis. In *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS '01)*. Philadelphia, Pennsylvania, USA. pp.166-175.

Mishra, P. Philpott, R. & Maler, E. (eds.), 2005. *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0*. Available at: http://docs.oasis-open.org/security/saml/v2.0/. [Accessed 1st July 2010].

Mitra, N. & Lafon, Y., 2007. *SOAP Version 1.2*. Available at: http://www.w3.org/TR/soap/. [Accessed 1st February 2010].

Mykkänen, J. & Tuomainen, M., 2008. An Evaluation and Selection Framework for Interoperability Standards. *Information and Software Technology*, 50(3), pp.176-197.

Mohammad, A. Kanaan, G. Kanaan, R. Khdour, T. Bani-Ahmad, S. Alarabeyyat, A., 2011. Toward Access Control Model for Web Services Applications. *International Journal of Research and Reviews in Computer Science (IJRRCS)*, 2(2), p.253.

## N:

Naedele, M., 2003. Standards for XML and Web Services Security. *Computer*, 36(4), pp.96-98.

Neale, P. Thapa, S. & Boyce, C., 2006. *PREPARING A CASE STUDY : A Guide for Designing and Conducting a Case Study for Evaluation Input*, Available at: www2.pathfinder.org/site/DocServer/m_e_tool_series_case_study.pdf. [Accessed 3rd February 2010].

Nezhad, H. R. M. Benatallah, B. Casati, F. & Toumani, F., 2006. Web Services Interoperability Specifications. *Computer Practices*, 39(5), pp.24-32.

NIST, 1996. *Electronic Data Interchange*. Available at: http://www.itl.nist.gov/fipspubs/fip161-2.htm. [Accessed 7th February 2010].

Nordbotten, N. A., 2009. XML and Web Services Security Standards. *Communications Surveys & Tutorials*, 11(3), pp.4-21.

Nurse, J. R. C., 2010. *A Business-Oriented Framework for Enhancing Web Services Security for E-Business*. The University of Warwick. Coventry, UK.

## O:

OASIS, 2012. *About Us*. Available at: https://www.oasis-open.org/org. [Accessed 15 September 2013].

Olmedilla, D. Lara, R. Polleres, A. & Lausen, H., 2004. Trust Negotiation for Semantic Web Services. In *1st International Workshop on Semantic Web Services and Web Process Composition in Conjunction with the 2004 IEEE International Conference on Web Services*. San Diego, CA, USA. pp.81-95.

Olson, L. Winslett, M. Toni, G. Seeley, N. Uszok, A. & Bradshaw, J., 2006. Trust Negotiation as An Authorization Service for Web Services. In *22nd International Conference on Data Engineering Workshops*. Atlanta, GA, USA. pp.21.

Oppliger, R., 2003. Microsoft .NET Passport: A Security Analysis. *Computer*, 36(7), pp.29-35.

Orkphol, K. & Li, J., 2012. Multi-negotiation Targets in Automated Trust Negotiation over TrustBuilder2 Framework. In *8th International Conference on Computing Technology and Information Management (ICCM)*. Seoul, Korea. pp.101-105.

## P:

Paci, F. Mecella, M. Ouzzani, M. Bertino, E., 2011. ACConv – An Access Control Model for Conversational Web Services. *ACM Transactions on the Web (TWEB)*, 5(3). no.13.

Pallis, G. Stoupa, K. & Vakali, A., 2008. Storage and Access Control Issues for XML Documents. In *Information Security and Ethics: Concepts, Methodologies, Tools, and Applications*. London, UK. Idea Group Inc., pp.2616-2621.

Palmer, J. W. & Sabnani, K., 1986. A Survey of Protocol Verification Techniques. In *Military Communications Conference – Communications-Computers: Teamed for the 90's, 1986 (MILCOM 1986)*. Monterey, CA, USA. pp.1.5.1-1.5.5.

Paolucci, M. & Sycara, K., 2003. Autonomous Semantic Web Services. *IEEE Internet Computing*, 7(5), pp.34-41.

Papazoglou, M., 2012. *Web Services and SOA: Principles and Technology (2nd edition)*, Canada: Pearson.

Parducci, B. & Lockhart, H., 2010. *eXtensible Access Control Markup Language (XACML) Version 3.0*. Available at: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-en.html. [Accessed 18th March 2010].

Pautasso, C. & Wilde, E., 2009. Why Is the Web Loosely Coupled ? A Multi-Faceted Metric for Service Design. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. Madrid, Spain. pp.911-920.

Pearlman, L. Welch, V. Foster, I. Kesselman, C. & Tuecke, S., 2002. A Community Authorization Service for Group Collaboration. In *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02)*. Monterey, CA, USA. pp.50-59.

Peterson, J.L., 1977. Petri Nets. *ACM Computing Surveys (CSUR)*, 9(3), pp.223-252.

Pironti, A. Pozza, D. & Sisto, R., 2011. Automated Formal Methods for Security Protocol Engineering. In *Cyber Security Standards, Practices and Industrial Applications: Systems and Methodologies*. pp.138-166.

Prud, E. & Seaborne, A., 2008. *SPARQL Query Language for RDF*. Available at: http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/. [Accessed 13th April 2011].

## R:

Rein, R. & Fokkinga, M., 1999. Protocol Assuring Universal Language. In *Third International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*. Florence, Italy. pp.241-258.

Rezaei, R. Chiew, T. K. Lee, S.P. & Aliee, Z. S., 2014. Interoperability Evaluation Models: A Systematic Review. *Computers in Industry*, 65(1), pp.1-23.

Rezaei, R., Chiew, T. K. & Lee, S.P., 2014. 2014.An Interoperability Model for Ultra Large Scale Systems. *Advances in Engineering Software*, 67, pp.22-46.

Roman, D. Keller, U. Lausen, H. Bruijn, J. & Lara, R. Stollberg, M. Polleres, A. Feier, C. Bussler, C. & Fensel, D., 2005. Web Service Modelling Ontology. *Applied Ontology*, 1, pp.77-106.

Rosen, M. Lublinsky, B. Smith, K. T. & Balcer, M.J., 2008. *Applied SOA: Service-Oriented Architecture and Design Strategies*, Indianapolis, Indiana: Wiley Publishing, Inc.

Rowan, L., 2005. Security in a Web Services World. *Network Security*, 2005(6), pp.7-10.

Runeson, P. & Höst, M., 2009. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Software Engineering*, 14(2), pp.131-164.

Ryutov, T. Zhou, L. Neuman, C. Leithead, T. & Seamons, K.E., 2005. Adaptive Trust Negotiation and Access Control. In *Proceedings of the Tenth ACM Symposium on Access Control Models and Technologies*. Stockholm, Sweden. pp.139-146.

## S:

Sabbari, M & Alipour, H.S., 2011. Improving Attribute Based Access Control Model for Web Services. In *2011 World Congress on Information and Communication Technologies (WICT)*. Mumbai, India. pp.1223-1228.

Sandhu, R. S. Coyne, E. J. Feinstein, H. L. & Youman, C.E., 1996. Role-based Access Control Models. *Computer*, 29(2), pp.38-47.

Schlager, C. Sojer, M. Muschall, B & Pernul, G., 2006. Attribute-based Authentication and Authorisation Infrastructures for E-Commerce Providers. In *Proceedings of 7th International Conference on Electronic Commerce and Web Technologies (EC-Web 2006)*. Krakow, Poland. pp.132-141.

Schneier, B., 1995. *Applied Cryptography: Protocols, Algorithms and Source Code in C (2nd edition)*, New York: John Wiley & Sons.

Seamons, K. E. Chan, T. Child, E. Halcrow, M. Hess, A. Holt, J., Jacobson, J. Jarvis, R. Patty, A. Smith, B. Sundelin, T. & Yu, L., 2003. TrustBuilder: Negotiating Trust in Dynamic Coalitions. *DARPA Information Survivability Conference and Exposition*. pp.49-51.

Seamons, K. E. Winslett, M. & Yu, T., 2001. Limiting the Disclosure of Access Control Policies during Automated Trust Negotiation. In *New York and Distributed System Security Symposium*. San Diego, CA, USA. pp.1-11.

Seamons, K. E. Winslett, M. Yu, T. Smith, B. Child, E. Jacobson, J. Mills, H. & Yu, L., 2002a. Requirements for Policy Languages for Trust Negotiation. In *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks*. Monterey, CA, USA. pp.68-79.

Seamons, K. E. Winslett, M. Yu, T. Yu, L. & Jarvis, R., 2002b. Protecting Privacy during On-line Trust Negotiation. In *Proceedings of the 2nd International Conference on Privacy Enhancing Technologies*. San Francisco, CA, USA, pp.129-143.

Shah, R. & Apte, N., 2004. *Web Services: A Business Module Packaing Strategy.* Available at: http://www.informit.com/articles/article.aspx?p=170421. [Accessed 4th July 2010].

Shen, H. and Hong, F., 2006. An Attribute-Based Access Control Model for Web Services. In *Proceedings of the Seventh International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT '06)*. Taipei, Taiwan. pp.74-79.

Skogsrud, H. Benatallah, B. and Casati, F. 2003. Model-Driven Trust Negotiation for Web Services. *IEEE Internet Computing*. 7(6) pp.45-52.

Skogsrud, H. Benatallah, B. and Casati, F. 2004a. Trust-Serv:Model-Driven Lifecycle Management of Trust Negotiation Policies for Web Services. In *Proceedings of the 13th International Conference on World Wide Web*. New York, USA. pp.53-62.

Skogsrud, H. Benatallah, B. Casati, F. & Dinh, M.Q. 2004b. Trust-Serv: A Lightweight Trust Negotiation Service. In *Proceedings of the 30th VLDB Conference*. Toronto, Canada. pp.1329-1332.

Skogsrud, H., Benatallah, B. and Casati, F. 2004c. A Trust Negotiation System for Digital Library Web Services. *International Journal on Digital Libraries*. 4(3) pp.185-207.

Skogsrud, H. Motahari-Nezhad, H. R. Benatallah, B. & Casati, F., 2009. Modeling Trust Negotiation for Web Services. *Computer*, 42(2), pp.54-61.

Smith, B. Seamons, K. E. & Jones, M.D., 2004. Responding to Policies at Runtime in TrustBuilder. In *Fifth IEEE International Workshop on Policies for Distributed Systems and Networks*. New York, USA. pp.149-158.

Smith, R. E., 2001. *Authentication: From Passwords to Public Keys*, Boston: Addison Wesley.

Squicciarini, A. C. Bertino, E. Trombetta, A. & Braghin, S., 2012. A Flexible Approach to Multi-Session Trust Negotiations. *IEEE Transactions on Dependable and Secure Computing*, 9(1), pp.2-15.

Stake, R.E., 1995. *The Art of Case Study Research*, CA: Thousand Oaks.

Steel, C. Nagappan, R. and Lai, R., 2006. *Core Security Patterns Best Practices and Strategies for J2EE, TM Web Services, and Identity Management*, Pearson Education, Inc.

Steiner, J. G. Neumant, C. & Schiller, J.I., 1988. Kerberos: An Authentication Service for Open Network Systems. In *Proceedings of the Winter 1988 Usenix Conference.* Dallas, Texas, USA. pp.191-202.

Sunshine, C., 1979a. Formal Techniques for Protocol Specification and Verification. *Computer*, 12(9), pp.20-27.

Sunshine, C., 1979b. *Formal Methods for Communication Protocol Specification and Verification*, Indiana: RAND Corporation.

Sunshine, C. A. Thompson, D .H. Erickson, R. W. Gerhart, S.L. & Schwabe, D., 1982. Specification and Verification of Communication Protocols in AFFIRM Using State Transition Models. *IEEE Transactions on Software Engineering*, SE-8(5), pp.460-489.

## T:

Taylor, B. Sinha, G. & Ghoshal, T., 2006. *Research Methodology: A Guide For Researchers In Management And Social Sciences*, New Delhi: Prentic-Hall of India Pvt.Ltd.

Tere, G. M. & Jadhav, B.T., 2010. How to Improve XML Web Services Performance? In *Proceedings of the International Conference and Workshop on Emerging Trends in Technology (ICWET '10)*. Mumbai, Maharashtra, India. pp.257-260.

Thompson, M. R. Esiari, A. & Mudumbai, S., 2003. Certificate-based Authorization Policy in A PKI Environment. *ACM Transactions on Information and System Security (TISSEC)*, 6(4), pp.566-588.

Tolk, A. Diallo, S. & Turnitsa, C., 2007. Applying the Levels of Conceptual Interoperability Model in Support of Integratability, Interoperability, and Composability for System-of-Systems Engineering. *International Journal Systemics, Cybernetics and Informatics*, 5(5), pp.65-74.

Tolk, A. and Muguira, J. A., 2003. The Levels of Conceptual Interoperability Model. In *Proceedings of the IEEE Fall Simulation Interoperability Workshop*. Orlando, Florida, USA. pp.14-19.

Turnitsa, C.D., 2005. *Extending the Levels of Conceptual Interoperability Model*. Available at: http://en.wikipedia.org/wiki/Conceptual_interoperability. [Accessed 16th May 2011].

## U:

Utting, M. & Legeard, B., 2006. *Practical Model-Based Testing: A Tools Approach*, San Francisco, CA, USA: Morgan Kaufmann Publishers.

## V:

Vedamuthu, A. Orchard, D. Hirsch, F. Hondo, M. Yendluri, P. Boubez, T. & Yalcinalp, U., 2007. *Web Services Policy 1.5-Framework*. Available at: http://www.w3.org/TR/ws-policy. [Accessed 18th February 2010].

Verisign, 2013. *About Verisign*. Available at: http://www.verisigninc.com/en_US/company-information/about-verisign/index.xhtml. [Accessed 4th July 2013].

Vogels, W., 2003. Web Services are not Distributed Objects. *IEEE Internet Computing*, 7(6), pp.59-66.

## W:

Wang, H. Huang, J. Z. Qu, Y. & Xie, J., 2004. Web Services: Problems and Future Directions. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(3), pp.309-320.

Wang, W. Tolk, A. & Wang, W., 2009. The Levels of Conceptual Interoperability Model : Applying Systems Engineering Principles to M & S. In *Proceedings of the 2009 Spring Simulation Multiconference (SpringSim '09)*. San Diego, CA, USA. pp.128.

Welch, V. Barton, T. Keahey, K. & Siebenlist, F., 2005. *Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Collaboration.*

Available at: http://grid.ncsa.illinois.edu/papers/gridshib-pki05-final.pdf. [Accessed 2nd March 2011].

West, C. H., 1978. General Technique for Communications Protocol Validation. *IBM Journal of Research and Development*, 22(4), pp.393-404.

Winsborough, W. H. & Li, N., 2002a. Towards Practical Automated Trust Negotiation. In *IEEE 3rd International Workshop on Policies for Distributed Systems and Networks*. Monterey, CA. USA. pp.92-103.

Winsborough, W. H. & Li, N., 2002b. Protecting Sensitive Attributes in Automated Trust Negotiation. In *Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society*. Washington, DC. USA. pp.41-51.

Winsborough, W. N. & Li, N., 2006. Safety in Automated Trust Negotiation. *ACM Transactions on Information and System Security (TISSEC)*, 9(3), pp.352-390.

Winsborough, W. H. Seamons, K. E. & Jones, V.E., 1999. Negotiating Disclosure of Sensitive Credentials. In *Second Conference on Security in Communication Networks*. Amalfi, Italy. pp.1-13.

Winsborough, W. H. Seamons, K. E. & Jones, V., 2000. Automated Trust Negotiation. In *DAPRA Information Survivability Conference and Exposition*. South Carolina, USA. pp.88-102.

Winslett, M. Ching, N. Jones, V. & Slepchin, I., 1997. Using Digital Credentials on the World Wide Web. *Journal of Computer Security - Special Issue on Security in the World Wide Web*, 5(3), pp.255-267.

Winslett, M. Yu, T. Seamons, K. E. Hess, A., Jacobson, J. Jarvis, R. Smith, B. & Yu, L., 2002. Negotiating Trust on the Web. *IEEE Internet Computing*, 6(6), pp.30-37.

Wonohoesodo, R. & Tari, Z., 2004. A Role Based Access Control for Web Services. In *Proceedings of the 2004 IEEE International Conference on Services Computing (SCC '04)*. Shanghai, China. pp.49-56.

Woods, D. & Mattern, T., 2006. *Enterprise SOA: Designing IT for Business Innovation*, CA: O'Reilly Media, Inc.

W3C, 2012. *About W3C*. Available at: http://www.w3.org/Consortium/. [Accessed 6th May 2012].

## X:

Xu, F. Lin, G. Huang, H. & Xie, L., 2004. Role-based Access Control System for Web Services. In *The Fourth International Conference on Computer and Information Technology (CIT '04)*. Wuhan, China. pp.357-362.

## Y:

Yin, R. K., 2013. *Case Study Research Design and Methods (5th edition)*, London: SAGE Publications Inc.

Yolum, P., 2004. Correctness Requirements for Multiagent Commitment Protocols. In *Proceedings of 19th International Symposium on Computer and Information Sciences (ISCIS '04)*. Kemer-Antalya, Turkey. pp.955-965.

Yu, D. Tan, C. Wang, H. & Yang, J., 2011. An Express Trust Negotiation Mechanism Based on Negotiation History. *Journal of Information & Computational Science*, 8(4), pp.609-617.

Yu, T. Winslett, M. & Seamons, K. E., 2001. Interoperable Strategies in Automated Trust Negotiation. In *Proceedings of the 8th ACM conference on Computer and Communications Security*. Philadelphia, PA, USA. pp.146-155.

Yu, T. & Winslett, M., 2003a. A Unified Scheme for Resource Protection in Automated Trust Negotiation. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*. Oakland, California, USA. pp.110-122.

Yu, T. Winslett, M. & Seamons, K. E., 2003. Supporting Structured Credentials and Sensitive Policies through Interoperable Strategies for Automated Trust Negotiation. *ACM Transactions on Information and System Security (TISSEC)*, 6(1), pp.1-42.

Yuan, E. & Tong, J., 2005. Attributed Based Access Control (ABAC) for Web Services. In *Proceedings of the IEEE International Conference on Web Services (ICWS '05)*. Orlando, Florida, USA. pp.561-569.

## Z:

Zartman, W. I. and Berman, M.R., 1982. *The Practical Negotiator*, New Haven, CT, USA. Yale University Press.

Zhang, W. & Engelen, R., 2008. An Adaptive XML Parser for Developing High-Performance Web Services. In *Proceedings of the 2008 Fourth IEEE International Conference on eScience (ESCIENCE '08)*. Indianapolis, Indiana, USA. pp.672-679.

Zhang, Y. Wu, M. Wu, L. & Li, Y., 2014. Attribute-Based Access Control Security Model in Service-Oriented Computing. *Lecture Notes in Electrical Engineering*, 163, pp.1473-1479.

Zhao, F., 2006. *Maximize Business Profits Through E-partnerships*, London: IRM Press.

Zhang, S. Guan, S. Mei, Y. & Pan, L., 2009. NAGUAL: A Novel Automated Trust Negotiation Model Based on Attribute Constraint. In *2009 International Conference on Networking and Digital Society*. Guiyang, Guizhou, China. pp.63-68.

# Appendix A. Protocol Messages

## A.1 a <TNPrepareRequest> message

Table A.1 describes the attributes and elements used in a <TNPrepareRequest> message within the protocol.

| | |
|---|---|
| /***TNPrepareRequest*** | A unique header message for a Web Service Requester to trigger the preparation stage of TN. |
| /TNPrepareRequest /***ID*** | This required attribute indicates the unique identifier of a <TNPrepareRequest> message. |
| /TNPrepareRequest /***@RemoteResourceOwner*** | This required attribute uniquely specifies a resource owner within the Web Service Provider. The value of this attribute is an identifier to indicate a resource owner. |
| /TNPrepareRequest /***@Resource*** | This required attribute specifies the target resource name owned by the resource owner. A combination of this attribute with the attribute "RemoteResourceOwner" uniquely indicates the target resource. |
| /TNPrepareRequest /***StrategyList*** | A sub element embedded in the <TNPrepareRequest> element, which contains all the information about the supported strategies. |
| /TNPrepareRequest /StrategyList /***@Number*** | A required attribute used to indicate the number of the <StrategyList> sub elements. The motivation of designing this attribute is that a Web Service Provider can efficiently deal with the sub elements <Strategy> by knowing the total number of these elements. |
| /TNPrepareRequest /StrategyList /***Strategy*** | A multitude of this sub <Strategy> element can be embedded in the <StrategyList> element. The value of this element is a name of a specific strategy. |
| /TNPrepareRequest /StrategyList /Strategy /***@ID*** | This required attribute indicates the unique identifier of the name of a strategy. |
| /TNPrepareRequest /LanguageCombinations /***@Number*** | A required attribute used to indicate the number of the <LanguageCombination> sub elements. |
| /TNPrepareRequest | A sub element embedded in the <LanguageCombinations> |

| | |
|---|---|
| /LanguageCombinations /***LanguageCombination*** | element to store a specific combination of a policy language, and a credential language. In each <LanguageCombination> element, only one policy language and one credential language can be set. A multitude of this element can be embedded in the <LanguageCombinations> element. |
| /TNPrepareRequest /LanguageCombinations /LanguageCombination /***PolicyLanguage*** | This is an element indicating a specific policy language that is supported in combination with a specific credential language, where the policy compliance checker can support this. The value of this element is a name of a specific policy language. |
| /TNPrepareRequest /LanguageCombinations /LanguageCombination /***CredentialLanguage*** | An element indicating a specific credential language that can be supported in combination with a specific policy language where the policy compliance checker can support this. The value of this element is a name of a specific credential language. |
| /TNPrepareRequest /**{any}** and /TNPrepareRequest /**@{any}** | These mechanisms provide extensibility to allow Web Services to define new elements or information to provide extensibility. Such mechanisms are also presented in all other steps but are not restated. |

Table A.1. Syntax and Semantics of a <TNPrepareRequest> message

## A.2 A <TNPrepareResponse> message

Table A.2 describes the attributes and elements used in a <TNPrepareResponse> message within the protocol.

| | |
|---|---|
| /***TNPrepareResponse*** | A unique header message used in response to the message in the <TNPrepareRequest> element at step three. It allows the Web Service Provider to establish all the available options for a common strategy language combination to be chosen. |
| /TNPrepareResponse /***ID*** | This required attribute indicates the unique identifier of an authorisation request message. |
| /TNPrepareResponse /***InResponseTo*** | This optional attribute indicates which unique <TNPrepareRequest> message determined by its ID will be sent in response. The value of this attribute should be a specific ID obtained from the <TNPrepareRequest> message sent from the current counterpart. |
| /TNPrepareResponse /***@TNCanBeUsed*** | An attribute to explicitly inform a Web Service Requester whether TN can be used with respect to the |

| | |
|---|---|
| | interoperability issue. The value of the attribute can only be “yes” or “no”. If the value is “yes”, it means that a common strategy and language combination have been found so the interoperability issue will not affect TN. If the value is “no”, it means that TN cannot be used due to the interoperability issue. |
| /TNPrepareResponse /***Strategy*** | A sub element embedded in the <TNPrepareResponse> element to indicate the specific strategy the Web Service Provider can support. The value of this element is the name of a specific chosen strategy. Note that only one strategy can be set under the <TNPrepareResponse> element. The notion of the ideal strategy should be defined at a system level. If one or more interoperable strategy can be found, the Web Service Provider should consider which strategy should be chosen according to a predefined preference order within the system. |
| /TNPrepareResponse /***ChosenLanguageCombination*** | This is a sub element embedded in the <TNPrepareResponse> element to indicate which combination in the previous <TNPrepareRequest> message has been chosen. |
| /PrepareResponse /ChosenLanguageCombination /***PolicyLanguage*** | A sub element indicates that a policy language has been chosen by the Web Service Provider for the specific combination. The value of this element is the name of a specific policy language. |
| /TNPrepareResponse /ChosenLanguageCombination /***CredentialLanguage*** | A sub element indicates that a specific credential language has been chosen by the Web Service Provider for the specific combination. The value of this element is the name of a specific credential language. |
| /TNPrepareResponse /***Fault***: | This is an optional element used when the value of the attribute “TNCanBeUsed” is “no” to explicitly inform the Web Service Requester that TN cannot be used. The reasons are “Strategic interoperability issue” or “Language interoperability issue”. |

Table A.2. Syntax and semantics of a <TNPrepareResponse> message

## A.3 A <AuthzDecisionQuery> message

Table A.3 describes the attributes and elements used in an <AuthzDecisionQuery> message within the protocol. Existing SAML syntax and semantics are not highlighted in bold. This is in order to distinguish the new or modified elements and attributes added into the <AuthzDecisionQuery> message from the existing elements and attributes of this message as defined in the SAML specification.

| | |
|---|---|
| /AuthzDecisonQuery /@*ID* | This required attribute indicates the unique identifier of an authorisation request message. |
| /AuthzDecisionQuery /@*Resource* | This required attribute specifies the name of the resource that a Web Service Requester is intended to access. |
| /AuthzDecisionQuery /***InResponseTo*** | This optional attribute indicates which unique <TNPrepareResponse> message determined by its ID, is going to be contained in this response. The value of this attribute should be a specific ID obtained from the <TNPrepareResponse> message sent from the current counterpart. |
| /AuthzDecisionQuery /**@*RemoteResourceOwner*** | This required attribute indicates a unique owner's name of the remote resource, because different resources belonging to different owners may have the same name. To enable the Web Service Provider to understand which specific resource is requested by the Web Service Requester, the owner's name should be provided to allow the Web Service Provider to locate the unique resource. A Web Service Provider involved in TN is actually a system on behalf of a human user or an organisation to negotiate with a Web Service Requester, which is on behalf of another human user or an organisation. Thus, the owner's name needs to be known by a Web Service Provider (or a Web Service Requester) to look for the relevant policies and credentials in its local system. This attribute and the attribute "Resource" predefined in the <AuthzDecisionQuery> uniquely specify an owner's resource. |
| /AuthzDecisionQuery /**@*LocalRequesterName*** | This required attribute denotes a unique local requester's name. |
| /AuthzDecisionQuery /Issuer | This optional attribute uniquely specifies the identity of the issuer. |

| | |
|---|---|
| /*@SPProvidedID* | |
| /AuthzDecisionQuery /Subject /SubjectConfirmation /***@Method*** | This attribute has been defined in the SAML specification, but to use it in the protocol defined in this research, a new value "TN" is added to this attribute to infer the use of TN. |
| /AuthzDecisionQuery /*Action* | This sub element allows a Web Service Requester to assert what actions it wishes to perform with respect to the target resource. |

Table A.3. Syntax and semantics of an <AuthzDecisionQuery> message

## A.4 A <PolicySet> message

Table A.4 describes the attributes and elements used in a <PolicySet> message. The existing syntax and semantics defined in the XACML specification are not highlighted in bold, in order to distinguish them from the new and modified syntax and semantics as defined in this protocol.

| | |
|---|---|
| /PolicySet | A unique header message used to contain the content of a temporary policy file, where a temporary policy file can include one or more policies. |
| /PolicySet /@PolicySetId | This required attribute indicates the unique identifier of a message expressing policies. |
| /PolicySet /***@RemoteResourceOwner*** | This required attribute specifies the owner's name of the remote resource. Therefore, an identifier indicating each counterpart's identity can help a Web Service distinguish one from the other. |
| /PolicySet /***@LocalPolicyOwner*** | This required attribute indicates the local policy owner's name. It is used as an identifier to enable a Web Service that provides the policy to remember each local user's identity it is on behalf of. This attribute is designed by taking into consideration the case that a Web Service may use TN with different counterparts at the same time, and in each session of TN, it is on behalf of a different local user to negotiate with a different counterpart. Hence, this attribute providing an identifier of each local user can aid a Web Service in distinguishing one from the other. |
| /PolicySet | This required attribute denotes which local resource the |

| | |
|---|---|
| /@***ProtectedLocalResource*** | entire policy file is protecting. This attribute is designed to guarantee that the policy disclosed by a Web Service is the correct one, in correspondence with the protection of the local resource requested by the counterpart. |
| /PolicySet<br>/@***LocalPolicyFileName*** | This required attribute records the name of the policy file. |
| /PolicySet<br>/@***IndividualPolicyTotalNumber*** | This required attribute records the total number of the <IndividualPolicy> sub elements. |
| /PolicySet<br>/***Policy*** | This is the existing element defined in the XACML specification, but its semantics are modified in this protocol. In order to help those policy languages that cannot express multiple policies, a change is made to enable support for such functionality. A policy that is not expressed in an XML language can be set as a value under this message. The value can be encoded in a BASE64 format, if necessary. |
| /PolicySet<br>/Policy<br>/@PolicyId | This is an existing attribute defined in the XACML specification. It is a required attribute used as a unique identifier to indicate an individual policy, when the <Policy> element is used. |

Table A.4. Syntax and semantics of a <PolicySet> message

## A.5 A <CredentialSet> message

Table A.5 describes the attributes and elements used in a <CredentialSet> message.

| | |
|---|---|
| /***CredentialSet*** | A unique header message used to contain multiple combinations of different credential files. |
| /CredentialSet<br>/@***ID*** | This required attribute indicates the unique identifier of a message expressing credentials. |
| /CredentialSet<br>/@***CredentialTotalNumber*** | This required attribute indicates the number of submitted credentials. |
| /CredentialSet<br>/@***LocalCredentialOwner*** | This required attribute shows the local credential owner's name within the <CredentialSet> element. It is used as an identifier to help a Web Service remember the identity of a user, it is on behalf of. |
| /CredentialSet<br>/@***MeetRemotePolicy*** | This optional attribute specifies the name of the remote policy from the other Web Service within the <CredentialSet> |

| | |
|---|---|
| | element, this individual credential can satisfy. This attribute is only used, when the credentials are submitted to fulfil the specific policies in a <PolicySet> message. |
| /CredentialSet /***@MeetRemotePolicyOwner*** | This optional attribute specifies the remote policy owner's name within the <CredentialSet> element. It is designed to be a complement to the attribute "meetRemotePolicy" to help a Web Service to quickly understand the identity of the owner of the policy, whose policy can be satisfied with the individual credentials. This attribute is only used, when the credentials are submitted to fulfil the specific policies in a <PolicySet> message. |
| /CredentialSet /***Credential*** | A multitude of this sub element can be embedded in the <CredentialSet> element. Each <Credential> element is used to contain an individual credential file including its attributes. This design is necessary, because more than one credential may be required to be disclosed in a round. |
| /CredentialSet /Credential /***@ID*** | This required attribute indicates the identifier of each credential within each <Credential> element, which enables a Web Service to distinguish a specific credential from the others. |
| /CredentialSet /Credential /***@CredentialType*** | This required attribute used to denote the type of the local credential within each <Credential> element. |
| /CredentialSet /Credential /***Fault*** | This optional element is used, when there is no local credential that can be submitted to satisfy the remote policy from the other Web Service. |

Table A.5. Syntax and semantics of a <CredentialSet> message

## A.6 A <Response> message

Table A.6 describes the attributes and elements in a <Response> message. The existing syntax and semantics of this message defined in the SAML specification are not highlighted in bold, in order to distinguish them from the new and modified syntax and semantics defined in this protocol.

| | |
|---|---|
| /Response /***@InResponseTo*** | This attribute has been defined within the SAML specification, and its semantics are changed in this protocol. This optional attribute defined in SAML is supposed to specify to which request message this response |

| | |
|---|---|
| | message should respond with; therefore, the value of this attribute is supposed to indicate the unique <AuthzDecisionQuery> message. However, in TN, both the Web Service Requester and Web Service Provider can generate this response. Thus, only when the Web Service Provider makes sure that the resource can be accessed (all the pertinent policies have been fulfilled), the value of this attribute should indicate the first authorisation message; otherwise, the value of this attribute should indicate the previous received message from the counterpart, when TN has failed. |
| /Response<br>/Status<br>/StatusCode<br>/***@Value*** | This attribute has been defined within the SAML specification, and its semantics are changed in this protocol. There are a variety of values defined in SAML, but in TN, only one value is used, which is "fail". |
| /Response<br>/***Fault*** | This is an optional element used to inform the Web Service Provider about the reason why TN is not successful. Five reasons are defined here, which are exactly as those defined in the <Fault> message within the <AuthzDecisionStatement> message (see section A.7). |

Table A.6. Syntax and semantics of a <Response> message

## A.7 A <AuthzDecisionStatement> message

Table A.7 describes the attributes and elements in an <AuthzDecisionStatement> message. The existing syntax and semantics of this message defined in the SAML specification are not highlighted in bold, in order to distinguish them from the new and modified syntax and semantics defined in this protocol.

| | |
|---|---|
| /AuthzDecisionStatement<br>/***@InResponseTo*** | This attribute has been defined within the SAML specification, and its semantics are changed in this protocol. This optional attribute defined in SAML is supposed to specify to which request message this response message should respond with; therefore, the value of this attribute is supposed to indicate the unique <AuthzDecisionQuery> message. However, in TN, both the Web Service Requester and Web Service Provider can generate this response. Thus, only when the Web Service Provider makes sure that the resource can be accessed (all the pertinent policies have been fulfilled), the value of this attribute should indicate the first authorisation message; otherwise, the value of this attribute should indicate the previous received |

| | |
|---|---|
| | message from the counterpart, when TN has failed. |
| /AuthzDecisionStatement /@*Resource*: | This attribute has been defined within the SAML specification. The value of this attribute is the target resource name that a Web Service Requester aims to access. |
| /AuthzDecisionStatement /**@*ResourceOwner*:** | This attribute indicates the unique name of the resource owner. |
| /AuthzDecisionStatement /@*Decision*: | This attribute has been defined within the SAML specification, and its semantics are changed in this protocol. There are a variety of values defined in SAML, but in TN, only two values are used: "Permit" and "Denied". |
| /AuthzDecisionStatement /*Action*: | This element has been defined within the SAML specification. The value of this element should be related to proper access permission such as "read only", "read and write" etc. |
| /AuthzDecisionStatement /***Fault***: | This is an optional element used to inform the Web Service Requester about the reason why TN is not successful. Five reasons are defined:<br>• **Reason one**: Wrong Received Credentials. This normally occurs when the remote Web Service is a malicious Web Service, or the internal logic of TN developed in the remote Web Service is wrong, so the received credentials cannot fulfil local policies; otherwise, a response message should have been received to inform that no remote credentials can fulfil local policies.<br>• **Reason two**: Unknown language. This also occurs when the remote Web Service Requester is a malicious Web Service, or the internal logic of TN developed in the remote Web Service Requester is wrong, because the received policies or credentials are written in an unknown language. Since this result should have been avoided, if two Web Services are developed according to the preparation stage.<br>• **Reason three**: No Local Credentials. With the use of the eager strategy, this occurs, if there are no more local credentials that can be unlocked by the received credentials. With the use of the parsimonious strategy, this occurs if there are no more local credentials containing the required attributes that can fulfil the received policies. |

| | |
|---|---|
| | • **Reason four**: Policy Cyclic Dependencies. This occurs, when policy cyclic dependencies have been discovered. • **Reason five**: Beyond Maximum Threshold. This occurs, if the number of times of the disclosure of the same credentials is beyond the maximum threshold. |

Table A.7. Syntax and semantics of an <AuthzDecisionStatement> message

# Appendix B. Case Study Evaluation in Chapter 6

This appendix presents a detailed case study evaluation of the proposed solution design in Chapter 6.

## B.1 Case Study Evaluation

### B.1.1 Construction of case studies

The use of case study for evaluation requires the construction of a proper case study design. Observing the two case scenarios presented in section 6.2, the context information for two case scenarios is different. Within case scenario 1, TN occurs between two honest Web Services, whereas in case scenario 2, TN occurs between an honest Web Service and a malicious Web Service. The different context information requires a multiple-case study design. More precisely, the number of the multiple-case study design is two.

In terms of the detailed case information, they are different for the two contexts. For the first vulnerability, the case information is that both two entities hold their own sensitive credentials protected by their own policies. To unlock each other's policies, each entity's sensitive credentials need to be disclosed. Unfortunately, as neither of the two entities owns the willingness to disclose its sensitive credentials first, PCD eventually occurs. For the second vulnerability, the case information is that a malicious entity keeps sending different policies for requesting the same credentials held by an honest entity, which does not realise the received different policies in essence are the same in terms of their requirements.

Once the decision of using the two-case study design has been made, the next decision to be made is whether there is a need for using embedded units of analysis. Within the two-case study design, as the two vulnerability issues can only occur, when a policy-exchanged-strategy is used, the first units of analysis should be different policy-exchanged-strategies. In the first case study, as the way of disclosing policies (credentials are always relevant to policies) used in different strategies is different, another two units of analysis are policies declared by entities and credentials held by entities. However, in the second case study, the conditions of policies and credentials held by entities for causing the occurrence of RCRA follows a fixed condition pattern as detailed in the case above, so there is no need to change the conditions of credentials and policies as sub units of analysis.

As a conclusion, the two case scenarios presented in section 6.2 can be directly used a two-case study design for evaluation. Combined with the embedded units of analysis, a decision was made that the embedded two-case study design would be used for evaluation (see figure B.1).

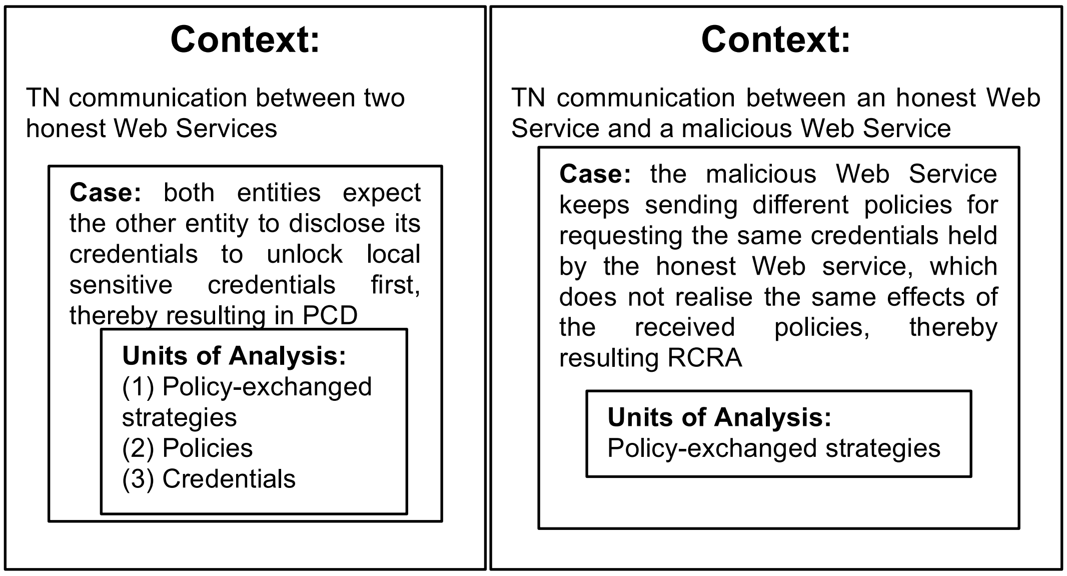

Context:
TN communication between two honest Web Services
Case: both entities expect the other entity to disclose its credentials to unlock local sensitive credentials first, thereby resulting in PCD
Units of Analysis:
(1) Policy-exchanged strategies
(2) Policies
(3) Credentials
Context:
TN communication between an honest Web Service and a malicious Web Service
Case: the malicious Web Service keeps sending different policies for requesting the same credentials held by the honest Web service, which does not realise the same effects of the received policies, thereby resulting RCRA
Units of Analysis:
Policy-exchanged strategies

Figure B.1. A general overview of embedded single-case designs

### B.1.2 Application of the proposed solution within case study for data collection

As reviewed in section 3.3.1, the currently existing policy-exchanged policies are the parsimonious strategy, PRUNES, DFANS, adaptive strategy and SRNS. Observing the characteristics of their information-exchanged phase, it can be concluded that the parsimonious strategy and adaptive strategy are designed to disclose policies as a whole to the counterpart; therefore they can be treated as one group in the two case studies. In terms of the information-exchanged phase in PRUNES and DFANS, there is a similarity and a difference.

The similarity held by two strategies is that with the use of either one strategy, an entity will only submit an atomic element (i.e. requesting one credential as an element of conjunction or disjunction in a rule as a part of a policy) of a policy for requesting one credential held by the counterpart. As this similarity causes the same behaviour when they disclose a policy in the first case study, the two strategies are categorised into one group when evaluated in the first case study.

In terms of their difference, with the use of the DFANS, an entity will disclose a relevant credential fulfilling the rule as a part of a policy disclosed by the counterpart immediately. By contrast, with the use of the PRUNES, an entity will not immediately disclose a relevant credential, even if it can fulfil the rule as a part of a policy disclosed by the counterpart. The action of submitting credentials by using the DFANS is the same when any one of the parsimonious strategy, adaptive strategy and SRNS is used. More precisely, with the use of any one of the four strategies, an entity submits local non-sensitive credentials to the counterpart immediately in the next round, after receiving policies from the counterpart. As this similarity held by four strategies is the same in terms of the actions performed in the second case study, they can be categorised as one group, whereas the PRUNES is separated as another group.

In terms of the SRNS, Liu et al. (2013) do not explicitly express whether an entity discloses its policies like the way of the parsimonious strategy or that of the PRUNES. Due to this reason, the SRNS is not taken into consideration in the evaluation of the first case study. Nevertheless, the evaluation result of using either the parsimonious strategy or PRUNES in the first case study should also apply to the SRNS, once this feature can be confirmed.

In conclusion, there are two case studies used for evaluation, and in each case study design, embedded cases are used. For simplicity, the term “circumstance” is used to refer to each embedded case. An overview of basic information of each circumstance is presented in table B.1.

| **Case** | **Case 1-circumstance 1** | **Case 1-circumstance 2** | **Case 1-circumstance 3** |
|---|---|---|---|
| **Occurrence** | PCD | PCD | Non-PCD |
| **Strategy** | Parsimonious/Adaptive | PRUNES/ DFANS | PRUNES/ DFANS |
| **Credential** | • WSA has a sensitive $C_1$.<br>• WSB has a sensitive $C_2$. | • WSA has a sensitive $C_1$.<br>• WSB has a sensitive $C_2$. | • WSA has a sensitive $C_1$ and a non-sensitive $C_3$.<br>• WSB has a sensitive |

| | | | |
|---|---|---|---|
| | | | $C_2$. |
| **Policy** | • WSA declares a $P_1$ requesting attribute information in a $C_2$ for protecting a $C_1$.<br>• WSB declares a $P_2$ requesting attribute information in a $C_1$ for protecting resource R and a $C_2$. | • WSA declares a $P_1$ requesting attribute information in a $C_2$ or a $C_4$ protecting a $C_1$.<br>• WSB declares a $P_2$ requesting attribute information in a $C_1$ or a $C_3$ for protecting resource R and a $C_2$. | • WSA declares a $P_1$ requesting attribute information in a $C_2$ or a $C_4$ protecting a $C_1$.<br>• WSB declares a $P_2$ requesting attribute information in a $C_1$ or a $C_3$ for protecting resource R. |
| **Case** | **Case 2-circumstance 1** | | **Case 2-circumstance 2** |
| **Strategy** | Parsimonious/DFANS/Adaptive/SRNS | | PRUNES |
| **Occurrence** | RCRA | | |
| **PMNOTOBR** | 1 | | |
| **Credential** | WSA has a $C_1$ | | |
| **Policy** | WSB declares multiple policies requesting attribute information in a $C_1$. | | |

Table B.1. An overview of circumstances in the two-case study design

In terms of the context of the first vulnerability issue – PCD, both entities are honest Web Services. Therefore, application of the proposed solution to either one entity is the same, as both of them want to detect the occurrence of PCD. According to the process of case scenario 1 presented in section 6.2, WSB should have detected the occurrence of PCD earlier than WSA. Therefore, the solution is mainly used in WSB to assess its effectiveness for the three circumstances in case 1.

Case 1-Circumstance 1

Units of Analysis:

- Strategy: both WSA and WSB use the parsimonious strategy
- Credentials: WSA has a sensitive $C_1$. WSB has a sensitive $C_2$.
- Policies: WSA declares a $P_1$ requesting a $C_2$ for protecting a $C_1$. WSB declares a $P_2$ requesting a $C_1$ for protecting resource R and a $C_2$.

The possessed information presented in symbols is shown in figure B.2 below.

WSA (Alice)
Credentials: $C_1$
Policies: $P_1$:$C_1 \leftarrow C_2$

WSB (Bob):
Resource: R
Credentials: $C_2$
Policies: $P_2$: (R, $C_2$)$\leftarrow C_1$

Figure B.2. Possessed information in case 1-circumstance 1

At step two: WSB discovers that Bob has declared the $P_1$ for protecting the disclosure of R, so it decides to send out the $P_1$. Following the algorithm presented in section 6.4.3, lines 3-6 activate. As the "Local Policy" table is an empty table, lines 11 and 12 activate, so WSB adds data relevant to the $P_1$ to the "Local Policy" table (shown in table B.2).

| **Table: Local Policy** |
|---|
| **LPID** |
| $P_1$ |

Table B.2. WSB adds P1 to the "Local Policy" table in case 1-circumstance 1

As PCD has not been detected, lines 15 and 16 activate, so WSB sends the $P_1$ in a $PS_1$ to WSA.

At step four: WSB decides to send out a $P_1$. Following the algorithm presented in section 6.4.3, lines 3-7 activate. As WSB finds out that the $P_1$ has been stored in the "Local Policy" table, line 8 activates to change the value of the variable PCD.detected to true, and line 10 activates, which in turn activates lines 18 and 19. At this point, WSB detects the occurrence of PCD.

From this circumstance, it can be identified that the proposed solution can successfully help WSB detect the occurrence of PCD at the correct time, if either the parsimonious/adaptive strategy is used.

<u>Case 1-Circumstance 2</u>

<u>Units of Analysis:</u>

- Strategy: both WSA and WSB use the PRUNES or DFANS
- Credentials: WSA has a sensitive $C_1$. WSB has a sensitive $C_2$.
- Policies: WSA declares a $P_1$ requesting attribute information in a $C_2$ or a $C_4$ protecting a $C_1$ and a $C_3$. WSB declares a $P_2$ requesting attribute information in a $C_1$ or a $C_3$ for protecting resource R and a $C_2$.

The possessed information presented in symbols is shown in figure B.3 below.

| | |
|---|---|
| WSA (Alice)<br>Credentials: $C_1$<br>Policies: $P_1$:$C_1 \leftarrow C_2 \vee C_4$ | WSB (Bob):<br>Resource: R<br>Credentials: $C_2$<br>Policies: $P_2$: (R, $C_2$)$\leftarrow C_1 \vee C_3$ |

Figure B.3. Possessed information in case 1-circumstance 2

Step two: WSB discovers that Bob has declared a $P_1$ for protecting the disclosure of R, so it decides to send out the first rule requesting a $C_1$ to WSA. Following the algorithm presented in section 6.4.3, lines 3-6 activate. As the "Local Policy" table is an empty table, lines 11 and 12 activate, so WSB adds data relevant to the $P_1$ to the "Local Policy" table (see table B.2 above).

Step three: WSA finds out that Alice has a sensitive $C_1$ protected by a $P_1$ requesting a $C_2$ or a $C_4$. With the use of the PRUNES, it sends out the first rule of the $P_1$ requesting a $C_2$ to WSB.

Step four: WSB discovers that Bob has a $C_2$ that is protected by the $P_1$, so it decides to send out the second rule of the $P_1$ requesting for a $C_3$ following the PRUNES. According to the algorithm presented in section 6.4.3, lines 3-6 activate. As WSB finds out that the $P_1$ has been stored in the "Local Policy" table, line 8 activates to change the value of the variable PCD.detected to true, and line 10 activates, which in turn activates lines 18 and 19. At this point, WSB decides that PCD will occur.

From this circumstance, it can be identified that the proposed solution cannot help WSB detect the occurrence of PCD at the correct time.

<u>Case 1-Circumstance 3</u>

<u>Units of Analysis:</u>

- Strategy: both WSA and WSB use the PRUNES or DFANS
- Credentials: WSA has a sensitive $C_1$ and a non-sensitive $C_3$. WSB has a sensitive $C_2$.
- Policies: WSA declares a $P_1$ requesting attribute information in a $C_2$ or a $C_4$ protecting a $C_1$ and a $C_3$. WSB declares a $P_2$ requesting attribute information in a $C_1$ or a $C_3$ for protecting resource R and a $C_2$.

The possessed information presented in symbols is shown in figure B.4 below.

WSA (Alice)
Credentials: $C_1$, $C_3$
Policies: $P_1$:$C_1 \leftarrow C_2 \vee C_4$

WSB (Bob):
Resource: R
Credentials: $C_2$
Policies: $P_2$: (R, $C_2$)$\leftarrow C_1 \vee C_3$

Figure B.4. Possessed information in case 1-circumstance 3

All the steps in this circumstance are the same as those shown in case 1-circumstance 2. The reason to use this case circumstance is to demonstrate the weakness of the proposed solution. By observing the conditions held in this circumstance, potential successful TN is possible, if WSB had sent the second rule of the $P_2$ requesting attribute information in a $C_3$ at step 4, WSA could have sent a $C_3$ to WSA at step 5. Unfortunately, due to the design of the proposed solution, circumstances without PCD will be wrongly determined to circumstances with PCD, when the PRUNES or DFANS is used, as long as a policy has to be separated by rules to be sent out.

The evaluation processes of the three circumstances of case 1 have been presented in detail. The following presents the evaluation process of two circumstances in case 2.

In terms of the context of the second vulnerability issue – RCRA, the honest Web Service is the victim suffering the attacks from the malicious Web Service by utilising this vulnerability of TN. So, application of the proposed solution is mainly used to help the honest Web Service detect the occurrence of RCRA so as to defend against it. According to the process of case scenario 2 presented in section 6.2, WSA is the honest Web Service. Therefore, the solution is mainly used in WSA to assess its effectiveness for the two circumstances in case 2.

Case 2-Circumstance 1

Units of Analysis:

Strategy: WSA and WSB use any one of the parsimonious strategy, adaptive strategy, PRUNES and SRNS

The possessed information presented in symbols is shown in figure B.5 below.

| | |
|---|---|
| WSA (Alice)<br>Credentials: $C_1$ | WSB (Bob):<br>Resource: R<br>Policies: $P_i$: R$\leftarrow C_1$ (i belongs to natural numbers) |

Figure B.5. Possessed information in case 2-circumstance 1

Steps 1 and 2 are the same as those shown in case scenario 2 as presented in section 6.2, so they are omitted here.

Step 3: After WSA analyses a $P_1$, it decides to submit Alice's $C_1$. Following the algorithm presented in section 6.4.3, lines 21 to 25 activate. As the "Local Credential" table is an empty table, lines 33 to 35 activate, so that WSA adds new data into this table (shown in table B.3).

| **Table: Local Credential** | | |
|---|---|---|
| **LCID** | **NOTOBR** | **PMNOTOBR** |
| $C_1$ | 1 | 1 |

Table B.3. WSA adds $C_1$ to the "Local Credential" table in case 2-circumstance 1

As RCRA has not been detected, lines 37 and 38 activate, so WSA sends a $C_1$ in a $CS_1$ to WSB.

Step 5: Upon analysing a $P_2$, WSA decides to submit Alice's $C_1$. Following the algorithm, lines 21 to 25 activate. At this time, WSA can find out data relevant to the $C_1$ in the "Local Credential" table. As the value of NOTOBR is "1", which is not less than "1" as the value of PMNOTOBR, so lines 30 and 31 activate setting the value of the variable RCRA.detected to true. This statement causes line 37 and lines 39 to 41 to activate, so that WSA detects the occurrence of RCRA. In order to defend against this attack, WSA decides to send out a last message to WSB and stops TN communication with it.

From this circumstance, it can be identified that the proposed solution can successfully help WSB detect the occurrence of RCRA and defend against this attack at the correct time, when any of the parsimonious strategy, adaptive strategy, PRUNES and SRNS is used.

Case 2-Circumstance 2

Units of Analysis:

Strategy: WSA and WSB use the PRUNES

The possessed information presented in symbols is the same shown in figure B.5 above.

Steps 1 and 2 are the same as those shown in case scenario 2 as presented in section 6.2, and step 3 is the same as the one shown in case 2-cirucmstance 1, so they are omitted here.

Step 5: Upon analysing the $P_2$, WSA knows Alice's $C_1$ can fulfil the $P_2$. According to the PRUNES, it sends a grant message to WSB informing that it has a $C_1$ that can fulfil the $P_2$. However, the real action of sending the $C_1$ has not been triggered.

Step 6: WSB sends another $P_3$ requesting other attribute information in the $C_1$.

Step i (i>=7, and i is an odd number) is the same as step 5.

With the use of the PRUNES, the credential-exchange phase (see section 3.3.1.2) will only occur, when the WSB as a service provider discloses the grant information that the resource R has been unlocked. However, as WSB is a malicious Web Service, it does not provide such information to WSA. As WSA only keeps sending grant information to WSB without deciding to perform real actions of sending out credentials, the proposed solution cannot be triggered to help WSA defend against RCRA. As a result, the proposed solution is not effective, when the PRUNES is used for TN.