Establishing Federated Trust Networks Among Web Services

A Thesis  
in TCC 402  

Presented to  
The Faculty of the  
School of Engineering and Applied Science  
University of Virginia  

In Partial Fulfillment  
of the Requirements for the Degree  
Bachelor of Science in Computer Engineering  

by  
James William Van Dyke  

March 23, 2004  

On my honor as a University student, on this assignment I have neither given nor received unauthorized aid as defined by the Honor Guidelines for Papers in TCC Courses.  

Signed: _______________________

Approved ______________________ Date __________________
Technical Advisor – Professor Alfred C. Weaver

Approved ______________________ Date __________________
TCC Advisor – Processor Patricia C. Click
PREFACE

I would like to thank the following individuals for their invaluable support:

Professor Alfred C. Weaver
Processor Patricia C. Click
Professor Samuel J. Dwyer III
Andrew M. Snyder
James Hu
Xiaohui Chen
Tim Mulholland
Andrew Marshall
Vincent Noel

I would also like to thank the following institutions for their support:

University of Virginia
UVA Computer Science Department
UVA Health System
Microsoft Corporation
# Table of Contents

LIST OF FIGURES.......................................................................................................................... III
GLOSSARY OF TERMS.................................................................................................................. IV
ABSTRACT ........................................................................................................................................ V

I. INTRODUCTION.......................................................................................................................... 1
   PURPOSE STATEMENT ............................................................................................................... 1
   BACKGROUND AND MOTIVATION ....................................................................................... 1
   PROBLEM STATEMENT .......................................................................................................... 2
   SCOPE AND METHOD ........................................................................................................... 5
   THESIS OVERVIEW ............................................................................................................... 6

II. LITERATURE, RESEARCH, AND STATE OF THE ART REVIEW ........................................... 7
   INTRODUCTION .................................................................................................................... 7
   BASIC TERMINOLOGY ......................................................................................................... 7
   SIMPLE TRUST RELATIONSHIPS .......................................................................................... 8
   FEDERATION .......................................................................................................................... 10
   OVERVIEW OF CORE WEB SERVICE TECHNOLOGIES ..................................................... 11
   OVERVIEW OF SUPPORTING WS-SPECIFICATIONS ......................................................... 12
   RESEARCH COMMUNITY CONTRIBUTIONS ....................................................................... 12

III. METHODS.................................................................................................................................. 14
   BACKGROUND ....................................................................................................................... 14
   RESEARCH .............................................................................................................................. 15
   DESIGN AND REVIEW ......................................................................................................... 15

IV. ARCHITECTURE SPECIFICATION ....................................................................................... 18
   INTRODUCTION AND PURPOSE .......................................................................................... 18
   INTRODUCTION TO TRUST LEVELS .................................................................................... 19
      Background and Limitations of the State of the Art ............................................................ 19
      Proposed Trust Level Concept ......................................................................................... 21
   INTRODUCTION TO TRUST GROUPS ............................................................................... 25
      Background and Motivations ............................................................................................ 25
      Proposed Trust Groups Concept ...................................................................................... 26
   INTRODUCTION TO TRUST AUTHORITIES .................................................................... 29
   GENERAL SYSTEM ARCHITECTURE ................................................................................. 30

V. DISCUSSION OF SYSTEM ARCHITECTURE ......................................................................... 32
   BENEFITS OF THE CURRENT SYSTEM ARCHITECTURE ................................................. 32
   LIMITATIONS AND OPEN-QUESTIONS RELATED TO THE CURRENT SYSTEM ARCHITECTURE .... 33

VI. CONCLUSION ............................................................................................................................ 36
   SUMMARY .............................................................................................................................. 36
   INTERPRETATIONS ............................................................................................................... 36
   RECOMMENDATIONS AND FUTURE WORK ...................................................................... 38

BIBLIOGRAPHY ............................................................................................................................. 40
APPENDICES ................................................................................................................................. 44
LIST OF FIGURES

FIGURE 1: BASIC TRUST RELATIONSHIP ........................................................................................................... 8
FIGURE 2: BASIC “DIRECT TRUST” FEDERATION ........................................................................................... 10
FIGURE 3: THE FEDERATED ATM NETWORK ..................................................................................................... 11
FIGURE 4: SYSTEM ARCHITECTURE DESIGN ................................................................................................. 31
Glossary of Terms

Claim
A claim is a declaration made by an entity (e.g. name, identity, key, group, privilege, capability, attribute, etc).

Direct Trust
Direct trust is when a relying party accepts as true all (or some subset of) the claims in the token sent by the requestor.

Direct Brokered Trust
Direct Brokered Trust is when one party trusts a second party who, in turn, trusts or vouches for, the claims of a third party.

Federation
A federation is a collection of realms that have established trust. The level of trust may vary, but typically includes authentication and may include authorization.

Indirect Brokered Trust
Indirect Brokered Trust is a variation on direct brokered trust where the second party can not immediately validate the claims of the third party to the first party and negotiates with the third party, or additional parties, to validate the claims and assess the trust of the third party.

Security Token
A security token represents a collection of claims.

Signed Security Token
A signed security token is a security token that is asserted and cryptographically signed by a specific authority (e.g. an X.509 certificate or a Kerberos ticket).

Proof-of-Possession
Proof-of-possession is authentication data that is provided with a message to prove that the message was sent and or created by a claimed identity.

Signature
A signature is a value computed with a cryptographic algorithm and bound to data in such a way that intended recipients of the data can use the signature to verify that the data has not been altered since it was signed by the signer.

Security Token Service (STS)
A security token service is a Web service that issues security tokens (see WS-Security). That is, it makes assertions based on evidence that it trusts, to whoever trusts it. To communicate trust, a service requires proof, such as a security token or set of security tokens, and issues a security token with its own trust statement (note that for some security token formats this can just be a re-issuance or co-signature). This forms the basis of trust brokering.

Trust
Trust is the characteristic that one entity is willing to rely upon a second entity to execute a set of actions and/or to make set of assertions about a set of subjects and/or scopes.

Trust Domain/Realm
A Trust Domain/Realm is an administered security space in which the source and target of a request can determine and agree whether particular sets of credentials from a source satisfy the relevant security policies of the target. The target may defer the trust decision to a third party (if this has been established as part of the agreement) thus including the trusted third party in the Trust Realm.

*Terms and Definitions from
ABSTRACT

The distributed, loosely-coupled nature of web services may revolutionize the way society communicates and exchanges information. The technology provides an easy way for entities to share data and services with other entities using a common framework and a standardized messaging protocol. Engineering solutions that address the security and privacy concerns of web services, however, is a challenging task. Many corporations and standards organizations currently undertaking this task have developed specifications and tools to address these concerns, but this effort is largely a work in progress.

This thesis approaches the security and privacy concerns of web services from the context of the healthcare industry. Healthcare information networks are complex systems involving a variety of participants, providers, roles, services, and external stakeholders. Thus, they expose many of the challenges inherent in large-scale distributed web service systems. Furthermore, motivations such as patient privacy protection and recent laws like the Health Insurance Portability and Accountability Act (HIPAA) make security a fundamental concern within the healthcare industry.

A research group led by Professor Alfred C. Weaver has developed an initial testbed of web services that models information exchange within a healthcare system. This thesis builds on the group’s efforts and extends current industry specifications to enhance web service support for security and federation across trust domains. Brokering trust relationships among web services in various domains is essential when developing large, multi-organizational distributed systems. Respecting privacy and maintaining security through trust relationships among entities is essential for this interaction to occur.
I. INTRODUCTION

Purpose Statement

This thesis proposes a high-level system architecture that enables broad federation of trust among web services. It builds upon several recently proposed industry specifications regarding web services, and it suggests a mechanism for brokering dynamic trust relationships. Motivated by the needs of the healthcare industry, the thesis introduces this architecture in the context of medical information systems. At the same time, it demonstrates how the architecture’s advantages are generally applicable to a variety of distributed web service systems.

Background and Motivation

Like many complex distributed systems, healthcare information systems involve a variety of services and participants. When giving and receiving medical care, for example, participants such as doctors, technicians, administrative staff, and patients frequently interact with information services such as medical records databases, radiology image stores, and billing systems. In addition, users and services also communicate with external entities such as insurance companies, pharmacies, and health clinics. While regular communication is essential between healthcare providers, these exchanges are largely inefficient. Currently, exchanges generally occur in paper form or electronically using mostly custom, incompatible legacy systems.

Because these disparate users and services lack a common communications framework, it is difficult for healthcare participants to obtain comprehensive medical information about patients when providing care. A patient may have multiple medical
records stored at various locations (e.g., at a hospital, doctor’s clinic, pharmacy), and data such as lab results, drug prescriptions, and disease histories are often not consolidated. Thus, it is likely that healthcare participants could provide higher quality medical care if they had access to such information and resources, especially during emergency situations. In creating mechanisms to collect and to retrieve medical information, however, one must recognize that protecting patient privacy is a fundamental system requirement.

With the recent advent of web services, a standardized, loosely-coupled framework now exists that can incorporate the complex, cross-boundary interactions of a healthcare system into a fully-connected, distributed computer system. Web services are applications that can share information and services with other applications over the Internet using a common interface and messaging system (Wolter 1). Such an integrated environment for exchanging information may revolutionize communication and information-sharing practices not only for healthcare systems, but also for a variety of other industries. Imagine creating an integrated healthcare portal that provides personalized content for patients, connects doctors with the latest medical news and information, and facilitates communication between third-party research institutions, equipment suppliers, and administrative services. This system could improve the quality of medical care, increase operating efficiency, reduce costs, and expand healthcare providers’ access to resources and services, especially in remote locations and rural areas.

**Problem Statement**

Before the healthcare industry adopts a distributed web service architecture, it must be able to guarantee the technology’s capacity to protect patient privacy. The recent federal Health Insurance Portability and Accountability Act (HIPAA) of 1996 mandates that all
electronic healthcare transactions adhere to strict security and privacy requirements ("Standards" 1). First generation web service technologies, however, inherently lack support for security. According to industry leaders developing web service technologies, the single most important factor limiting widespread adoption of web services is security (Della-Libera “Federation” 3). Before web services can facilitate extensive integration of information and services, they must gain widespread adoption. Yet, without mechanisms to guarantee the security and privacy of communications, many organizations and customers are simply unwilling to invest in them. Please see Appendix VI for more information about HIPAA and healthcare security.

Maintaining security in a distributed web service system is a complex task. In large systems such as healthcare information networks, entities frequently need to share information and services across business, organizational, and technical boundaries. Because entities have different procedures, expectations, and responsibilities regarding the local use, storage, and transmission of information, it becomes difficult for one entity to trust the internal data-handling procedures of another. Likewise, entities may have incompatible mechanisms for assigning and managing identities, roles, and permissions among their organizations. Thus, if information exchange is fundamentally essential among businesses, organizations, and individuals, how is it practical given these differences? A certain degree of standardization is necessary. Standardized communication mechanisms should facilitate distributed information exchange, but they should remain loosely-coupled and function effectively within preexisting business structures.

Microsoft, IBM, VeriSign, and several other companies have formed a consortium to develop the next generation of web services. These companies are developing technologies that build upon existing business relationships, and they have specifically addressed many of
the technical challenges inherent in this effort, including issues involving security, privacy, and trust. The consortium has released recommended specifications on several of these subjects; however, this task remains largely a work in progress.

The consortium has defined mechanisms for securing information and communications, specifying security and privacy policies, communicating trust relationships, and other tasks. But, it has not satisfactorily identified a means for establishing trust relationships among entities. According to IBM and Microsoft, “‘trust’ is the characteristic that one entity is willing to rely upon a second entity to execute a set of actions and/or to make assertions about a set of subjects and/or scopes” (Della-Libera “Security in a Web Services World” 10). Trust may be direct or brokered, as “when one party trusts a second party who, in turn, trusts or vouches for, the claims of a third party” (11).

Considering the large number of participants interacting in a fully distributed web service system, it is impractical for entities to establish and to maintain explicit trust relationships with every customer and partner. While the consortium describes how to specify trust policies, how to communicate trust, and how to broker trust directly and indirectly through third-parties, it leaves the method for originating a trust policy to the organization or the web service administrator. While it is constructive for the specifications to remain implementation independent, entities still need a basis for initially defining and subsequently maintaining their trust policies. If a service potentially acquires a large number of clients in a complex distributed system, how can a web service designer or policy administrator practically enumerate all possible trust relationships with these clients or their respective organizations? Without having a process to establish trust relationships dynamically, it is difficult for entities to benefit fully from the loosely-coupled, distributed nature of large-scale web service networks.
Scope and Method

My thesis technical advisor, Professor Alfred C. Weaver, several graduate and undergraduate students, and I formed a research group to study web service security and trust federation beginning in the summer of 2002. Given the mandates of HIPAA and the UVA Computer Science Department’s close relationship with the UVA Medical School, we recognized that modeling a healthcare information network offered a promising testbed for studying web services and security. The healthcare system involves a diverse group of internal and external participants, and many of them could benefit from the improved efficiency and communication features provided by web services. Medical information should be readily available to a patient’s healthcare providers when they are offering care; however, it is imperative that systems broker access to such private information responsibly and limit its availability outside of healthcare contexts. Thus, in conjunction with the UVA Department of Radiology and with grant support from Microsoft, we established an initial testbed of medical web services during the first half of 2003.

Building on past research experience, this thesis suggests an architecture for designing dynamic web service trust networks. Specifically, it discusses how three primary concepts, trust levels, trust groups, and trust authorities, can enhance existing web service security specifications. Trust levels can be thought of as mutually-recognized, standardized identifiers of the “level of trustworthiness” for a given user or service. For example, an entity that interacts with a user who holds an authentication ticket signed by one of its trusted partners should be able to trust this user to some degree, as outlined by the entity’s trust policy. With these additions, systems can define dynamic trust policies and generate dynamic trust relationships. While beneficial in the context of our medical testbed, this trust architecture is also generally applicable to a variety of distributed web service systems.
In completing this thesis project, I followed the method outlined below. First, I conducted extensive research on web services and associated security technologies, and I reviewed existing web service specifications and literature. Then, I identified the limitations of current technologies in terms of trust federation, and I proposed several concepts and a system architecture design to address these concerns. Throughout the design process, I shared these ideas with professors and members of my research team, and I revised the design based on their feedback. As part of my future work with Professor Weaver, our research group plans to enhance the design and to implement a working prototype of the system. Additionally, I also plan to share the design ideas with contacts at Microsoft to obtain their feedback and suggestions.

**Thesis Overview**

This thesis proposes a system architecture for establishing dynamic networks of trust. It enumerates past research and background information, methods, results, and conclusions regarding the thesis project. Chapter Two reviews relevant literature and current technical specifications; it also discusses the limitations of these specifications. The chapter outlines the cooperative progress of our research group and discusses how the group’s achievements contributed to the direction of this thesis project. Chapter Three explains the methods for the project, including an overview of the design and review stages. Chapter Four describes the project’s results and final design architecture specification. Chapter Five discusses these results and the proposed architecture. It reviews the architecture’s applications, benefits, limitations, and future extensibility. The Conclusion summarizes the thesis projects, interprets the results and their impact on the future of web services, and suggests recommendations for future work.
II. LITERATURE, RESEARCH, AND STATE OF THE ART REVIEW

Introduction

Information sharing is a fundamental component of all communications, transactions, and interactions. Individuals and services exchange information both internally within organizations and also externally with business partners, associates, and customers. While we share some information generally in the public domain, we intend much of it for particular recipients. In traditional communications, we verify the identity of senders and receivers directly in-person or by referencing proofs of identity such as signatures, passports, or photo ids. However, when we share information electronically, we must use other techniques to validate one’s identity.

Basic Terminology

In Internet security terminology, we define a claim as a declaration made by an entity, such as one’s name or group, and we define a security token as a collection of such claims (Kaler “WS-Federation” 3). A proof-of-possession is information like a password or fingerprint scan that a sender provides to verify, or authenticate, his or her identity. Thus, a proof-of-possession token is a security token containing claims along with supporting evidence to substantiate these claims (3). After evaluating tokens and proof-of-possession data, a service determines whether to trust a requestor’s identity claims and thereby authenticate the user based on its security policy. Subsequently, the service performs an authorization step to determine whether the identified user has permission to access the requested resources.
Simple Trust Relationships

So that users do not have to authenticate their identities each time they access resources within an organization, administrators can specify security policies that establish implicit trust relationships between services and a local security token service. As shown in Figure 1, all web services in a single trust domain usually trust security tokens issued and electronically signed by the local security token service. Thus, when a client attempts to access resources from a service, the client first contacts the system’s local security token service and authenticates. If the process is successful, the security token service issues the user a signed security token. The user may present this token when requesting information from a service. If the service’s security policy specifies that it can trust the token signed by the local authority, the service can proceed directly to the authorization process. If the user has permission to access the requested information, the web service responds appropriately. Note that the user can present the security token on subsequent requests to services in the local trust domain until the token expires.

![Figure 1: Basic Trust Relationship](image)

This figure shows the basic scope of trust between a client, an STS, and a service in a trust domain (Smith).
The process described above becomes more complicated when communications cross organizational boundaries. Consider two separate organizations (i.e., two separate trust domains) that have a preexisting trust relationship between them. Services in each organization, however, can only interact with clients who have security tokens issued by their respective local security token services. Thus, if the client from one organization contacts the service of the second organization, it would take the following steps as shown in Figure 2: the client would send claims and proof-of-possession information to its local security token service and request a security token. Upon successful authentication, the security token service would then issue the client a signed security token. At this point, the client could present this token to services in the STS’s local trust domain; however, in this case, the client intends to access a resource in another trust domain. As stated earlier, the two security token services in this scenario have a previously established trust relationship, which is reflected in each of their security policies. In order to access a resource in the other trust domain, the client first needs to acquire a security token issued by the resource’s security token service. When the client contacts the STS, it presents the security token from its local STS. The resource’s STS checks its policy and determines how to handle requests from the client’s domain based on the established trust relationship. If the policy allows, the STS issues a signed security token to the client, which it can then use to access resources in the STS’s local trust domain. The queried resource checks its own policy during the authorization step before fulfilling the client’s request.
This figure shows a basic federated trust relationship between two trust domains (Kaler “WS-Federation”).

Federation

The term federation describes the process where entities broker trust and exchange information across organizational boundaries. The WS-Federation specification defines a federation as “a collection of realms that have established trust” (Kaler “WS-Federation” 3). As shown in Figure 3, a useful example of federation as it exists today is the ATM banking network. Several examples of trust in this system include the following: independent banks agree to conduct financial transactions with each other’s clients; the client trusts that a foreign bank’s ATM will communicate his or her account number and PIN responsibly and securely to his or her home institution to conduct the transaction; the client’s bank trusts that the foreign bank generates a valid request on behalf of its customer; and the foreign bank trusts that upon issuing funds to the client, the client’s home institution will transfer the appropriate funds to it. Proponents of web services believe that new web service
technologies will expand federation as it exists for ATM networks into a generalized architecture that is applicable for many types of Internet communications.

![Image](image-url)

**Figure 3: The Federated ATM Network**

This figure demonstrates the federated-nature of the worldwide ATM Network.

**Overview of Core Web Service Technologies**

Three primary technologies form the foundational layer upon which to build web services. These are the Simple Object Access Protocol (SOAP), the Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI). Web services use SOAP to communicate. This protocol specifies the format and content for messages exchanged between web services (Gudgin 1). WSDL standardizes how organizations describe the capabilities and functionality of their web services. Potential clients can use this information to determine whether a certain web service satisfies their needs. Developers also reference WSDL documents when creating applications that interact
with web services (Christensen 1). UDDI defines a “whitepage directory” of web services that allows clients to search for services that suit their needs (Wolter 1).

**Overview of Supporting WS-Specifications**

While the initial core web service technologies do not support security, the consortium of companies developing the next generation of web services has suggested several new specifications to address many of the security scenarios discussed previously. The WS-Security specification defines mechanisms for protecting web service communications, including the use of encryption and digital signatures to protect message integrity and confidentiality (Kaler “WS-Security” 1). The WS-Trust specification suggests methods for exchanging security tokens between web services (Kaler “WS-Trust” 1). WS-Policy and WS-SecurityPolicy specify how web services can express policies regarding their security practices and requirements (Hondo “WS-Policy” 1). WS-Federation, using WS-Trust as a protocol for communicating trust, discusses how to broker trust relationships between web services across organizational and business boundaries (Kaler “WS-Federation” 1). WS-SecureConversation proposes an end-to-end solution for maintaining message confidentiality. A variety of other supporting specifications also exist, and some have yet to be written (Kaler “WS-SecureConversation” 1).

**Research Community Contributions**

Because web services are a new technology and the development of related security standards is an ongoing process, there are many current research projects on the topic. A group of researchers from Hewlett Packard Laboratories published a paper on creating third-party trust services for web service systems. The paper identified the advantages of
reusing commonly implemented trust and security mechanisms via web services, and it also argued that software engineering practices should be expanded to consider the operational aspects of these service components. While the Hewlett Packard paper does not suggest specific solutions to the problem, this thesis proposes a system architecture based on trust levels, trust groups, and trust authorities that addresses several related concerns (Baldwin 507).

Research at the University of Vermont identified two primary impacts of web services on companies in the technology industry: “1. outsourcing many traditional IT activities as credible and reliable providers of services emerge; and 2. leveraging internal capacities to design distinctive web services that can be sold to other organizations” (Ratnasingam 5). Referring to technology trust issues related to web services, University of Vermont Professor Pauline Ratnasingam suggests, “For any web services scheme to work, attention must be paid to rigorous authentication, integrity, non-repudiation, encryption, and security matters that enforce technology trust” (Ratnasingam 6).
III. METHODS

Background

Professor Weaver and I recognized the potential for web service technologies while I was an undergraduate research assistant for him during the summer of 2002. We realized that many challenging security and privacy issues existed involving these technologies, especially in implementing real-world applications of web services. Furthermore, considering current patient privacy issues and recent HIPAA laws, we also recognized motivations for studying web services in the context of a complex healthcare information system. Researching interactions between internal and external healthcare participants could provide us with valuable insight on web service security issues. Thus, in conjunction with Professor Samuel Dwyer of the UVA Hospital Radiology Department and graduate student Andrew Snyder, we jointly collaborated on a grant proposal to Microsoft proposing development of a testbed of medical web services to study web service security and trust federation issues. We successfully received equipment and $200,000 in funding from Microsoft in the fall of 2002 to begin working on this research project.

During the spring semester of 2003, we formed a larger research group of undergraduate and graduate students and developed the first stages of our medical testbed. This testbed included a web portal, a medical records database, biometric authentication support, prototype authentication and authorization rule engines, and a concept for persistent, single-sign-in user authentication tokens. After demoing our progress to a Microsoft representative in May 2003, we were also invited to present our research progress at Microsoft’s Faculty Research Summit DemoFest in Redmond, Washington, near the end of July 2003.
Research

During the summer of 2003, I selected trust federation as a topic related to our research project that I wanted to pursue for my fourth-year thesis. While participating in a summer internship at Microsoft during this time, I met with a contact from the Indigo Web Services group who was developing next-generation web service technologies. We discussed ideas and our research project, and he offered to review our group’s work and results in the future. Throughout the fall semester of 2003, I conducted extensive research on web services, GXA web service specifications, security, distributed computing, federation, and other related concepts. Additionally, I reviewed HIPAA technical specifications and privacy rules, HL-7 formats, and the Clinical Document Architecture specification. I also read proposals and academic literature about current research projects on web service technologies.

During this process, I identified several limitations with existing web service security and policy specifications. These limitations involved methods for establishing trust relationships, brokering and federating trust, publishing security policies, enforcing security requirements, and exchanging security tokens. Exploring these limitations and potential solutions, I proposed three primary concepts to address these concerns: trust levels, trust groups, and trust authorities. Subsequently, I began designing a system architecture for trust federation that utilized these concepts, and I also explored ways to incorporate them into extended versions of existing GXA web service specifications.

Design and Review

Following extensive research, I designed the proposed system architecture using an iterative process. First, I identified the components, participants, and core services of the
system. Then, I enumerated a list of the required roles, behaviors, and responsibilities of the components and services. Referencing existing web service documentation and specifications, I also noted the security requirements and privacy concerns related to these roles and behaviors. Subsequently, I considered various scenarios and use cases within the system. Using this information, I expanded and refined the lists of components and their roles. Furthermore, I also proposed the contents of our system’s security tokens, identified potential factors that influence trust levels and trustworthiness, and considered operators and elements necessary to generate policies using these concepts.

After identifying the concepts above, I reviewed the existing GXA specifications, especially WS-Policy, WS-SecurityPolicy, WS-Trust, and WS-Federation, and I compared the specifications’ features to my list of requirements. Some of the functionality and features that I recorded were supported in the specifications; however, other primary concepts were not supported, such as the ability to distinguish between various security tokens based on the relative trustworthiness of the authentication mechanisms used. Thus, after performing this analysis, I proposed extensions to the original specifications to support trust levels, trust groups, trust authorities, and other features. After documenting these extensions, I proposed a high-level system architecture incorporating these concepts.

Seeking frequent reviews and feedback from others was essential during the design process. I wanted to make sure that my research contributions were relevant and beneficial to the research community and current industry efforts involving web services. During weekly research group meetings, I shared my progress and ideas with Professor Weaver and other members of the team. They critiqued my work and suggested improvements. I also consulted Professor Humphrey of the Computer Science Department regarding the project. He suggested ideas and references from his own research, and he also confirmed that my
work on web services was progressing in a unique and useful direction. Furthermore, I consulted new WS-* specifications from Microsoft and IBM that emerged after completing my initial literature review, and I also referred to references on the Security Assertion Markup Language (SAML).

During the review process, I experimented with coding examples from Microsoft’s MSDN website using the Web Service Enhancements (WSE) 2.0 tool. This tool supports basic WS-Policy, WS-Security, WS-Trust, and several other specifications (Smith 1). After reviewing a sample project from the MSDN website that implemented a web service policy requiring a username/password token, I modified the code so that it accepted a custom security token containing a trust level value. Unfortunately, I found that WSE 2.0 does not support custom security tokens, and therefore, the web service’s policy was unable to interpret the custom token correctly (Horrell 1). While the next version of WSE will support custom tokens, I learned that using them is currently possible with additional custom implementation. Performing this experiment offered me valuable insight on how policies are consumed and enforced by web services. I also learned how to integrate C# code and GXA web service specification documents to create secure web services. This experience influenced the development of my proposed trust level and trust group definition documents, and it also helped me visualize and design the system architecture.

Combining several iterations of feedback from professors, group members, research results, and experimentation, I improved the specification and design of the proposed system architecture. As described in my future work section, our research group plans to implement a system incorporating the core federation concepts discussed in this thesis. Additionally, I also plan to ask several contacts at Microsoft to review these concepts and the proposed architecture.
IV. ARCHITECTURE SPECIFICATION

Introduction and Purpose

The WS-* specifications discussed in the last chapter introduce a variety of technologies that enhance the functionality of web services. With newly added support for communication security, transaction integrity, and message confidentiality, web services are becoming a practical alternative for developing large-scale, loosely-coupled distributed systems on the Internet. While these technologies enable an array of essential business interactions and communications, they lack the breadth of security features needed to integrate large, heterogeneous systems. Specifically, while WS-* technologies specify how two entities can share information through indirect or direct trust agreements, how are such trust relationships established initially? Considering the large number of web service interactions that may occur in future wide-area distributed systems, is it practical for network administrators to identify and to implement all possible trust agreements manually? Current specifications do not address this issue. In addition, how can one entity trust the security practices of another entity; is the trustworthiness of its authentication and authorization mechanisms acceptable? This architecture specification addresses these concerns by proposing three new security concepts: trust levels, trust groups, and trust authorities. Please see Appendix I for proposed extensions to WS-SecurityPolicy that integrate these concepts into the existing security token element.
Introduction to Trust Levels

BACKGROUND AND LIMITATIONS OF THE STATE OF THE ART

Throughout the Internet today, the most common method of authentication, the process by which an authority verifies the identity of a user, is certification of username (the claim) and password (the proof of possession for the claim). Recently, however, emerging biometric devices and other new technologies have begun to offer more robust solutions for verifying one’s identity. These mechanisms frequently consider multiple variables and degrees of freedom when analyzing patterns such as one’s fingerprint or iris. Such technologies eliminate the inconvenience of memorizing passwords, and they also attach biological proof of possession claims to particular users, making them difficult to falsify. Thus, the degree to which an authority is “sure” about the identity of a user varies depending on the type of authentication mechanism employed.

As discussed in Chapter Two, if a user or service demonstrates sufficient proof of identity to a security token service authority (i.e. sufficient proof to satisfy its security policy), the STS can choose to accept the credentials provided by the user, to verify them, and subsequently to issue a signed security token to the user for its local trust domain. Currently, regardless of the authority’s policy and chosen authentication mechanism, the contents of an issued security token is basically static. Specifically, the signed token generated by an authority is commonly one of the following types: x.509, Kerberos, or XML-based, such as a Security Assertion Markup Language (SAML) token (Hondo “WS-Policy” 1). Thus, when a foreign authenticated user visits a resource service in the STS’s local trust domain, the STS does not know whether the foreign STS performed authentication using a simple username and password pair or a more robust iris scan. Furthermore, even if a service in the domain conducts transactions that are more sensitive than those of other services, it cannot require a
minimum level of identity authentication trustworthiness for its requestors. Once a user has authenticated with the local STS, any service that trusts the STS will likely trust tokens signed by the STS, regardless of the original authentication method used to identify the user.

This problem is particularly evident in situations involving multiple trust domains. For example, assume that STS-A in domain A and STS-B in domain B have previously established a mutual trust relationship. Now, suppose that a user in domain A authenticates with STS-A and then attempts to access a resource web service in domain B. Following simple federation procedures as discussed in Chapter Two, assume that the resource requires each requestor to have a security token signed by its local STS authority before it will attempt to authorize the user for access. STS-B’s security policy specifies whether it will trust security tokens issued by STS-A; however, STS-B cannot make this decision conditionally based upon the trustworthiness of the particular authentication method used by STS-A to verify the identity of the user. What if STS-A’s authentication mechanism is acceptable for certain resource services within STS-B’s domain, but not for others? What if STS-A changes its mechanisms for authenticating users but does not notify STS-B? Once the user contacts STS-B with a security token signed by STS-A, STS-B must either accept or reject the token and subsequently issue the user a local security token for domain B. This decision cannot directly depend on the authentication mechanism employed by STS-A; STS-B must rely on the original trust agreement policy established for STS-A.

Note: In the following discussions, we assume that the policies of all services (resources) within a local trust domain implicitly require requestors to have security tokens issued by their local STS authority. Thus, all outside requestors must request and receive security tokens from the local STS before they can communicate with local services.
Additionally, please note the following information about policy documents referenced in the proposed trust level scheme below:

- Each individual resource has a policy document that determines whether requestors 1) have a signed security token issued by the resource’s locally trusted STS; 2) have a signed security token affirming that they have authenticated using a suitably trustworthy mechanism (e.g., a sufficient minimum trust level, as specified below); and 3) have appropriate roles, permissions, and contexts to obtain authorized access to the resource.

- STSs have token granting policies and mapping policies. Token granting policies specify whether an STS trusts tokens issued by a particular foreign STS (i.e., whether a trust relationship exists with the foreign STS’s domain). Mapping policies determine how a security token issued by a trusted foreign STS maps to a security token in the local trust domain (e.g., mapped trust level). While the token granting policy specifies whether a trust relationship exists, the mapping policy specifies the characteristics of the trust relationship. It is possible that these separate policies may be combined into one policy document.

PROPOSED TRUST LEVEL CONCEPT

This architecture proposes the use of trust levels to address the problems discussed above. A trust level can be described as a numeric value that represents the relative trustworthiness of a specific authentication mechanism. It expresses an STS’s level of confidence in the identity claims of a user. In other words, it conveys how “sure” an STS is that a requestor is who it claims to be.
System administrators should designate trust levels to distinguish between the relative trustworthiness of various authentication methods; trust level zero should likely identify the least secure method (perhaps anonymous access), while higher trust level values should indicate more secure methods (e.g., trust level one for username/password authentication, two for fingerprint scan authentication, and three for iris scan authentication). When defining trust levels, administrators may reference a variety of authentication mechanism characteristics, including a particular technology’s hardware device type, manufacturer, version, degrees of freedom, encryption support, etc. In addition, administrators have the option of designating distinguishing subcategories for a trust level using fractional values, as in the following example:

- Primary Trust Level = 1.0 for Username/Password Authentication
  - Secondary Trust Level = 1.1 for Username/Password Authentication with a required minimum password length of 6 characters
  - Secondary Trust Level = 1.2 for Username/Password Authentication with a required minimum password length of 6 characters and with a password containing at least one number and/or symbol

The following list identifies additional characteristics of the proposed trust level scheme:

- The actual definitions of trust levels may be arbitrary; however, the relative trustworthiness of various methods (as perceived or designated by administrators of a given trust domain) should be preserved in the scheme.
- Resource services can specify trust level requirements in their security policies. For example, a certain web service may require all requestors to be authenticated with a trust level of two or higher.
• Because security policies specify the actual interpretation of trust levels, a service may decide that trust levels can be additive. For example, authenticating using both a username and password (e.g. with a trust level of one) and a fingerprint scan (e.g. with a trust level of two) may be equivalent to authenticating with a single iris scan (e.g., with a trust level of three).

• Trust level definitions should at least be standardized within a local trust domain (i.e., the domain served by a single STS). A trust authority in the local trust domain should maintain trust level definitions and policies so that designations are consistent across the entire local trust domain. Trust levels may be standardized by hierarchies of trust authorities or perhaps by a central, widely-recognized trust authority. Policies should exist within each lower hierarchy and local domain regarding how to map a trust authority’s trust level definitions and designations to those of the local domain.

• Trust authorities should have policies for mapping outside trust level definitions to those consistent with the trust level scheme of the local trust domain. A particular trust level in one domain may map to a higher or lower trust level in a second domain, depending on existing trust relationships and associated policies defined for the various trust domains. For example, the trust authority for domain A may map domain B’s trust level three designation to a trust level of two; however, the authority for domain A may map domain C’s trust level three designation to a trust level of one.

*Example policies for STS A regarding its trust relationship with STS B:*

○ Trust STS B without reservation. Map STS B’s trust level definitions directly to corresponding trust levels in domain A.
Trust STS B with reservations.

- Variation 1: Map all of STS B’s trust level definitions to a trust level of one in domain A. Issue a security token for domain A with a trust level of one to the user with a security token from STS B. Note that the user may request to match a higher trust level by performing an additional authentication step with STS A as defined by its security policy.

- Variation 2: Follow the procedure specified in Variation 1, but only if the security token signed by STS B was issued in the last 15 minutes. Otherwise, require the user to re-authenticate.

Do not trust STS B. Require all users with security tokens signed by STS B to re-authenticate with STS A.

- The trust level is included as a field within the security token issued by an STS. The STS signs the token confirming that it has performed the authentication mechanism associated with the specified trust level. The explicit meaning of the trust level in the context of the STS’s trust domain is verifiable by checking the local trust authority’s definition document for the trust level. Based on their security policies, other STSs can choose to trust the original STS’s signed security token and associated trust level.

- Services have individual policies that specify their trust level requirements.

- One of the most important characteristics of a set of trust levels is its associated definition document in a given trust domain. Just as any inquiring web service or authority can request to view the security policy document of another service, authority, or STS, they can also request to view a trust authority’s trust level definition document (see the definition of a trust authority below). When establishing a trust relationship and defining mappings between trust levels for
various partners, administrators can reference these documents. As also discussed below, these documents can be useful for establishing dynamic trust relationships.

Please see Appendix II for the proposed WS-TrustLevel specification. This document suggests a standard format for trust level definitions.

Introduction to Trust Groups

BACKGROUND AND MOTIVATIONS

Services in various domains trust each other because of previously established trust relationships. These trust relationships, defined in each service’s policy document, identify how a service should handle requests from other principles. Because we assume that all services in a trust domain implicitly require requestors to have security tokens issued by the domain’s local STS, the STS is ultimately responsible for evaluating possible trust relationships, issuing security tokens, and making policy decisions (e.g., mapping trust levels to the local domain) based on these trust relationships. A fundamental question, however, is how are these trust relationships established between domains initially? From a business perspective, it is likely that forming a trust relationship with another organization first involves negotiating contracts and business agreements before it reaches the technical implementation stage. Subsequently, administrators or designers must manually document such relationships in policy documents using the WS-Policy and WS-SecurityPolicy specifications (Nadalin “WS-SecurityPolicy” 1). As stated previously in the Introduction, considering the large number of interactions in a future distributed web service system, it may be impractical for administrators to write such policy documents manually for all
partners, especially the minor ones. Thus, this thesis proposes the use of trust groups, in part, to facilitate dynamic trust relationships.

As discussed below, a second motivation for the use of trust groups is that they provide an easy, practical, and powerful mechanism for consolidating trust level definitions and other security requirements for groups of web services and domains.

**Proposed Trust Groups Concept**

This architecture proposes the use of trust groups to address the problems discussed above. A trust group can be described as a collection of security requirements agreed upon by administrators for a group of trust domains and claimed or validated by a trust authority. These security requirements could include minimum trust level criteria, policy requirements specified using WS-Policy and WS-SecurityPolicy (e.g., requiring the use of a certain encryption algorithm as described in WS-Security), and other custom membership policy claims. Custom membership policy claims may include custom-defined and custom-enforced metrics such as rankings, reliability levels, privacy categories, data update frequency values, domain namespaces, membership lists, supported platforms, etc. Such criteria are largely subjective; however, if trust group members define them in such a way that trust authorities can validate their compliance among members (automatically or by hand, perhaps through an application or auditing process), they can be useful in enabling dynamically established trust relationships.

This thesis defines dynamically established trust relationships as those in which an administrator or service designer does not manually need to create an explicit policy to govern a trust relationship. Rather, for example, an STS’s policy can specify that it requires a requestor to have a security token with a minimum trust level (as assigned by a recognized
and trusted authority) or to be a member of a certain trust group before it will issue it a local security token (or at least a security token with a higher trust level). In this case, the local STS may not be directly familiar (i.e. have a specific trust relationship) with the particular foreign STS that signed the requestor’s security token. However, if the local STS can obtain verifiable attribute information about the foreign STS from a trusted trust authority, then its policy may instruct the local STS to trust the token and its associated identify claims (at least for a low trust level value) and to issue the requestor a local security token. This description makes an important distinction between trusting an authority that verifies the trustworthiness of a domain (i.e., explicit trust, which may be direct or indirect via a third-party trust authority) and trusting an authority that verifies information about a domain. The trust authority does not assert direct claims about the trustworthiness of a domain; rather, the inquiring entity is free to make its own conclusions based on the provided information (i.e., dynamic trust). It is likely that dynamic trust would not be practical for high-security, privacy-sensitive scenarios. Likewise, it would also probably not be useful for establishing trust with large, important partners. Rather, dynamic trust could be used for frequent, low-priority and low-risk trust relationships, such as those described below.

Consider the procedure for establishing dynamic trust in the following scenario. Assume that the University of Virginia Health System has previously established business trust relationships with several pharmacies, including CVS, Revco, and Wal-mart. Thus, if web services hosted by these three entities present security tokens signed by their respective STSs to the local health system’s STS, they can obtain locally issued tokens for the UVA Health System domain in accordance with the domain’s policy requirements. Then, if authorized, these requestors can communicate with services in the UVA Health System trust domain.
In order to simplify the process of establishing trust relationships with health systems, also assume that a recognized consortium of pharmacies has created a trust group with certain membership requirements that it has defined and hosted with a recognized trust authority. This trust group defines minimum security and privacy requirements for participating domains, and it requires new candidates to apply for membership and to participate in a thorough auditing process for compliance before gaining membership in the group. If a new candidate successfully meets the membership requirements, the trust group administrator can contact the trust authority and add it to the list of members in the trust group.

The UVA Health System decides that it trusts the pharmacy consortium to evaluate its members. Thus, in order to benefit its customers and patients who use several of the pharmacies in the consortium, UVA decides to grant a minimum level of trust (e.g., a trust level of one) to all members of the consortium’s trust group. The UVA Health System can contact the trust authority, which it also trusts and through which the consortium created its trust group, to verify certain member lists and membership requirements. Thus, if a web service visits the UVA Health System domain with a security token signed by a foreign STS that is a member of the trust group, the local STS can grant the service a local security ticket (at an appropriate trust level as specified in its policy) even though the local domain does not have an explicit trust relationship (neither direct nor indirect) with the foreign domain. This is an example of a dynamic trust relationship. Note that the local STS may condition issuing the ticket on a variety of factors related to the original security token, including the associated trust level, trust group, issue time, etc.

In addition to facilitating dynamic trust, trust groups are also useful because they allow entities to consolidate and to standardize trust level definitions and other security
requirements for groups of partners through a commonly trusted trust authority. Rather
than including a list of members, a domain may simply use a trust group to specify security
requirements for potential partners and trust relationships.

Please see Appendix III for the proposed WS-TrustGroup specification. This document
suggests a standard format for trust group definitions.

**Introduction to Trust Authorities**

In the system described above, trust authorities maintain a repository of trust level,
trust group, and policy definitions for one or more trust domains. They respond to
definition inquiries from requestors and send digitally-signed replies. Trust authorities may
be widely trusted and recognized, similar to the way that certificate authorities such as
VeriSign are trusted, or they may be local for individual domains or groups of domains.
Thus, networks and hierarchies of trust authorities may exist. Trust authorities provide
interfaces for entities to establish, to update, and to reference policies and definitions. Trust
authorities standardize the most recent versions of such policies and definitions within
domains and groups of domains so that they are interpreted consistently. In some ways, the
responsibilities of trust authorities resemble those of a distribute file system server that
maintains access permissions and versioning.

Entities may choose to publish their trust level and trust group definitions publicly
via a trust authority. Trust domains may use these definitions to specify basic requirements
that partners must meet before being considered for potential trust relationships. Domains
seeking partnerships may implement solutions to meet these requirements and then seek
certification either from the associated domain or the trust authority. Alternatively, domain
administrators may review a particular domain’s trust definitions (listed on a trust authority)
and decide to adopt them into the policies of their own domain (e.g., standards organizations could attempt to standardize basic trust level or trust group designations for use with certain classes of business or government transactions, etc.). Finally, an individual domain may specify and control the administration and membership of its own trust group, thereby using the trust group to manage a list of trusted partners and trust relationships.

Groups of entities may also establish a trust authority to manage a common set of trust level definitions and security requirements. Entities may consult these documents whenever conducting business with members of this group to determine the appropriate communication and security procedures to follow. Trust authorities may hold standard definitions for the trust levels of common authentication devices or procedures. Entities that trust this authority may reference these standard definitions often, and associated web services may be able to interpret them automatically. In contrast, entities that use custom security devices or other security solutions may write custom trust level definitions that require interpretation from administrators via policy definitions before a service can process them automatically.

A trust authority may be a distinct service, or its functionality may be incorporated into an STS. If they are separate, an STS contacts a trust authority whenever it needs to confirm various trust definitions or policies, including its token granting and mapping policies. Administrators modify definitions and policies via the trust authority.

**General System Architecture**

This section suggests integrating the trust level, trust group, and trust authority concepts discussed above with components proposed by existing web service specifications,
including WS-Federation. The goal is to create a general system architecture that supports federation. Figure 4 is a diagram of the proposed system design.

Figure 4: System Architecture Design
This figure shows the high-level system design proposed in this thesis. Most importantly, this design broadly supports federation through the use of trust levels, trust groups, and trust authorities.

Please see Appendix V for a description of each component included in the design.

The following chapter discusses the strengths, limitations, and future development of this architecture.
V. DISCUSSION OF SYSTEM ARCHITECTURE

Benefits of the Current System Architecture

The proposed system architecture incorporates trust levels, trust authorities, and trust groups to create a system that broadly supports trust federation among domains. A local domain’s trust authority centrally hosts the domain’s trust level and trust group definitions, ticket-granting policies, and mapping policies. The security token service and other local services throughout the domain consistently reference these definitions and policies. Furthermore, outside entities request such information from a domain’s trust authority when establishing trust relationships with the domain and interacting with its services.

Incorporating trust levels and trust groups into the system and maintaining them through a central trust authority provides several basic elements needed for large-scale federation. In order for one domain to trust the security practices of a second domain, the first domain must be able to view and to evaluate the security practices of the second domain. The definitions and policies should be available in a common format so that domain administrators can compare and map them to their own practices. The trust authority manages trust level, trust group, and policy definitions and can provide them to inquiring requestors. In this way, the first domain can obtain the information it needs to establish a trust relationship with the second, and it can determine how to make requests and receive responses from the domain. Using these concepts, domains can establish trust relationships and conduct trusted transactions with a variety of other domains. In doing so, businesses, organizations, individuals, and governments can dynamically interconnect with other partners to communicate and to exchange information.
Limitations and Open-Questions Related to the Current System Architecture

The following list includes descriptions of several limitations and open questions related to the current architecture:

- The trust level definition specification is largely subjective, especially in the section of WS-TrustLevel containing optional custom fields. How much information should a trust level definition specify to have sufficient detail? How do trust levels take into account the non-technical aspects of trustworthiness? For example, a domain could have a highly secure physical network, but this fact says nothing about the trustworthiness of its business practices or corporate culture. Furthermore, while a domain may support highly secure iris scans, how can another domain evaluate the trustworthiness of its user enrollment process for iris scans? For example, a domain’s authentication process for matching an iris scan to an identity may be secure, but its policy for initially associating a user with a particular iris scan to an identity may not be secure. What if the enrollment process for an iris scan only requires the user to supply a password? While this policy may not be secure, is it practical for administrators to enroll all users for iris scans by hand?

- Even though the Results chapter claims that trust groups may be useful for low-risk, low-security, low-trust-level situations, is it practical that they can be used realistically to establish dynamic trust relationships? Many would argue that forming trust relationships requires human involvement, business agreements, and other non-objective factors. On the other hand, are not some trust relationships established using a compilation of objective criteria? For example, if an individual encounters a random person offering valuable information, is the individual more likely to trust
the person if he or she is a tenured University of Virginia professor (an example of objective criteria)? Why?

• How does an entity determine the list of objective criteria included in its trust level and trust group definitions? Are entities liable for the authentication tokens that they sign? Is the degree to which an entity assumes liability for wrongful authentications an important objective criterion?

• How does an STS map identifiers used to reference an entity in one domain to different identifiers used in a second domain? How does a domain practically store, reference, and specify all of its trust relationships with other customers, partners, suppliers, and stakeholders?

• How will trust authorities realistically function in this system? Will they be businesses? How will they practically verify that entities meet the requirements for certain trust levels and trust groups? What if an entity changes its security practices – how will the trust authority know about this change? Are trust authorities’ assertions reliable?

• How does the method of access that an authenticated individual uses affect the security of a service request? For example, assuming that encryption is used, is there a need to know whether the user’s connection is wireless or wireline?

• How does a domain create a loosely-coupled, web-service-based system without exposing sensitive internal resources to outside threats?

This thesis proposes and defines an abstract, theoretical system architecture. Large amounts of additional research, revisions, specifications, development, and testing are necessary to realize the proposed system and to achieve broad federation support. As
described in the Conclusion chapter, many of these issues will be addressed in future work related to this project.
VI. Conclusion

Summary

This thesis proposes an architecture that enables dynamic federation among web services. Through the use of three primary concepts, trust levels, trust groups, and trust authorities, it suggests extensions to current web service technologies that help establish trust relationships, federate user identities, and exchange information across trust domains. Consistent with our research group’s existing project, the thesis discusses these concepts in the context of healthcare systems.

Specifically, trust levels allow services in one domain to recognize and to trust users and services with authenticated identities from other domains. They allow entities to map security practices from one domain to comparable practices in another domain. Even though various domains’ internal security practices and procedures may differ, policies supported by trust levels account for these differences in establishing trusted communications. Trust groups consolidate multiple trust level constraints and other security requirements, and they may include lists of group members that meet these requirements. Trust groups are also useful in establishing trust relationships dynamically. Trust authorities maintain definitions of trust levels and trust groups, and they verify and sign these definitions for services. Administrators can reference definitions hosted by trust authorities when formulating policies and specifying security requirements for potential partners.

Interpretations

The proposed architecture incorporating trust levels, trust groups, and trust authorities provides a framework for establishing federated networks of trust among web
services. Previously proposed specifications, such as WS-Federation, suggested how requestors with existing security tokens could obtain new tokens from a resource’s local domain to gain potential domain access. However, they did not specify how such security token exchanges could occur given potential differences in the various domains’ methods for authenticating users and services. Furthermore, the specifications did not suggest strategies for establishing trust relationships between entities, mapping differences in the trustworthiness of security tokens across domains, and designating authorities to maintain and to publish security definitions and requirements. This thesis proposes solutions to these issues.

As discussed below, much additional work is needed to incorporate these fundamental concepts into a complete architecture that can realize all of the possible benefits of a fully web-service-based system. The proposed architecture does, however, suggest a substantial step towards this goal. Trust levels, trust groups, and trust authorities provide mechanisms that allow a variety of interactive communications. Our project group ultimately aims to use our research results to contribute to the future development of web service technologies and specifications. We envision a system that could realize the following scenario:

Patients can login to a medical portal from their home personal computer to access all of their healthcare information from a central source, even if this information is stored at various locations, including hospitals, doctor’s offices, and pharmacies. During this process, patients only login once as they browse information from various sources. Furthermore, doctors can access similar important information about patients from various locations or devices, even using mobile technologies
while meeting with a patient in his or her office. Likewise, computer services such as billing and insurance systems can dynamically gain restricted access to portions of patients’ medical records for account processing purposes, etc. Furthermore, rural healthcare providers are able to access information and resources provided by large medical facilities. In all of these contexts, interactions should respect and preserve patient privacy.

In general, we hope that our research efforts will have a positive impact on the future of web services, which we believe could significantly improve the way society communicates. An important part of this goal involves improving communications in the healthcare industry to enhance the quality of medical care. At the same time, we also hope that these advances will benefit communication among all types of communities, organizations, businesses, governments, and individuals. With improved communication means, we can better learn about each other’s needs, share experiences and services, collaborate, and ultimately enhance our ability to contribute to society.

**Recommendations and Future Work**

This thesis project and the efforts of our research group are only the beginning of what could become a large-scale, multiyear research project. With the close support of our sponsor, Microsoft, we aim to make significant contributions to the future of web service technologies, especially regarding security, trust, and federation. This fourth-year thesis has the potential to develop into a master’s thesis. Presently, I plan to continue working on the project for the remainder of the semester and during the summer. Future research will involve designing an entire multi-system architecture that incorporates trust levels, trust
groups, trust authorities, context-aware role-based access control, biometric authentication mechanisms, and other technologies. In addition, our group plans to implement a multi-system prototype using new web service technologies and our own proposed solutions that tests the security and communications performance of our designs. Ultimately, we hope that this research will positively contribute to future adoption and use of web service technologies, standards, specifications, and implementations. A large, dynamic system of web services supporting security and trust federation could substantially improve society’s ability to communicate, to share ideas, to conduct business, and to exchange information effectively.
BIBLIOGRAPHY

<http://www.hl7.org/about/hl7about.htm#WhatIsHL7>.


Bratt, Steve. “W3C: Leading the Web – and Web Services – to their Full Potential.” Online

Cantor, Scott. “Assertions and Protocols for the OASIS Security Assertion Markup
Language (SAML) V2.0.” Online article, 7 March 2004. OASIS Open. 22 March
core-2.0-draft-07.pdf>.

More Secure.” Online article, April 2003. Microsoft MSDN Magazine. 22 March
2004. <http://msdn.microsoft.com/msdnmag/issues/03/04/ws-
security/default.aspx>.

Christensen, Erik, et al. “Web Services Description Language (WSDL) 1.1.” Online
article, 15 March 2001. Ariba, International Business Machines Corporation, and


and Roadmap.” Online whitepaper, 7 April 2002. IBM Corporation and Microsoft
dnwssecure/html/securitywhitepaper.asp>.


APPENDICES

Appendix I

Excerpt from WS-SecurityPolicy
Security Token Element


/SecurityToken
This identifies a security token assertion.

/SecurityToken/@wsp:Preference
This optional attribute specifies the preference of this particular alternative. The preference is expressed as an xsd:int. The higher the value of the preference, the greater the weighting of the expressed preference. If no preference is specified, a value of zero is assumed.

/SecurityToken/@wsp:Usage
This mandatory attribute indicates the usage of this assertion (e.g., required, optional, etc.) per WS-Policy.

/SecurityToken/TokenType
This mandatory element expresses the type of the security token for this assertion specified by a QName. This is extensible, but the following types are predefined:

<table>
<thead>
<tr>
<th>QName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsse:X509v3</td>
<td>X.509 v3 certificate</td>
</tr>
<tr>
<td>wsse:Kerberosv5TGT</td>
<td>Kerberos V5 Ticket Granting Ticket</td>
</tr>
<tr>
<td>wsse:Kerberosv5ST</td>
<td>Kerberos V5 service ticket</td>
</tr>
<tr>
<td>wsse:SAMLAssertion</td>
<td>SAML Assertion</td>
</tr>
<tr>
<td>wsse:XrMLLicence</td>
<td>XrML Licence</td>
</tr>
</tbody>
</table>

/SecurityToken/TokenIssuer
This optional element’s contents are interpreted as the name of a trusted issuer (or names of trusted issuers).

/SecurityToken/Claims
This optional element contains data that is interpreted as describing type-specific claims that are expressed in the security token. TokenType-specific descriptions, such as required extensions in an X509 certificate, MUST be specified using this mechanism. Some of the TokenType-specific extensions are defined in Appendix I of this document.

/SecurityToken/{any}
This is a general extensibility mechanism to allow additional elements to be specified.

/SecurityToken/{@any}
This is an extensibility mechanism to allow additional attributes to be specified.

Proposed Extensions to the SecurityToken Element of WS-SecurityPolicy:

/SecurityToken/TrustLevel – defined in WS-TrustLevel
/SecurityToken/TrustGroup – defined in WS-TrustGroup
/SecurityToken/Expiration
/SecurityToken/TrustAuthority
/SecurityToken/TrustAuthority/Name
/SecurityToken/TrustAuthority/URI
/SecurityToken/TokenIssuer/URI
Appendix II

WS-TrustLevel v1.0
Trust Level Element

/wstl:TrustLevel/
(can be nested – children inherent fractional IDs of parents)
/wstl:TrustLevel/Children (e.g., 1.1 – the value must always be higher than base)
/wstl:TrustLevel/Base (all base trust levels are implicitly required and inherited – e.g. allows
the combination of trust level 1 and 2 to make trust level 3)
/wstl:TrustLevel/@Name
/wstl:TrustLevel/@ID – numeric identifier
/wstl:TrustLevel/@TargetNamespace (Domain or Realm)
/wstl:TrustLevel/DefinitionDate
/wstl:TrustLevel/Expiration
/wstl:TrustLevel/Version
/wstl:TrustLevel/Authority
/wstl:TrustLevel/TrustLevelReference – combine trust levels; reference existing trust level
definitions
/wstl:TrustLevel/AuthenticationMechanism
  Table with list of mechanisms (extensible – other mechanisms can be defined)
    Fingerprint
    IRIS
    Username / Password

    Example content in mechanism definition:

    /Mechanism specific fields
    /Device
      /Version
      /Production date
      /Manufacturer
    /Degrees of Freedom
    /Others
      e.g., Live fingerprint detection support, password characteristics
      (letters, numbers, symbols, min length)
/wstl:TrustLevel/Security
  Encryption Algorithm
  Keysize
  Support for other WS-Security features
Appendix III

**WS-TrustGroup v1.0**

Trust Group Element

/wstg:TrustGroup/

/wstg:TrustGroup/Base (all base trust levels are implicitly required and inherited – e.g. allows the combination of trust level 1 and 2 to make trust level 3)

/wstg:TrustGroup/@Name
/wstg:TrustGroup/@ID – numeric identifier
/wstg:TrustGroup/@TargetNamespace (Domain or Realm)
/wstg:TrustGroup/DefinitionDate
/wstg:TrustGroup/Expiration
/wstg:TrustGroup/Version
/wstg:TrustGroup/Authority
/wstg:TrustGroup/TrustGroupReference – combine trust groups; reference existing trust group definitions

/wstg:TrustGroup/TrustLevel

/wstg:TrustGroup/Security
  Encryption Algorithm
  Keysize
  Support for other WS-Security features

/wstg:TrustGroup/CustomMembershipPolicy
/wstg:TrustGroup/CustomMembershipPolicy/DefinitionDate
/wstg:TrustGroup/CustomMembershipPolicy/Expiration
/wstg:TrustGroup/CustomMembershipPolicy/Version
/wstg:TrustGroup/CustomMembershipPolicy/Authority
/wstg:TrustGroup/CustomMembershipPolicy/CustomMembershipPolicyReference
/wstg:TrustGroup/CustomMembershipPolicy/Claims/{any}

/Ranking
/Reliability
/Privacy
/Language
/Namespacedomain
/Membership
/Members
/Accuracy
/References
/UpdateFrequency
/InfoDelay
/ProtocolSupport
/OSPlatform
/SecurityPolicy
Appendix IV

Additional Proposed Elements and Procedures

Proposed Contents of Signed Custom Authentication Security Token
- ID (Medical System ID) / Pseudonym
- Trust Level
- Trust Groups
- Issuing Authority (contact for look-up, discovery, challenge response, pseudonym/attribute service reference)
- Expiration
- Date and Time Issued
- List of Referring Authorities (potentially embed previous tokens)
- Existing / New Account Flags

Proposed Process That a Local Domain’s STS Should Follow When Mapping UserIDs
- If a user arrives with a security token from the locally trusted STS
  - The user may access services in the local domain using this token
- If a user arrives with a security token from a trusted STS (i.e., a policy exists for mapping users authenticated with this STS)
  - If this is the first time that the user has used this security token to access a service within the local trust domain (i.e., no previous mapping for the UserID listed in this security token exists in the user database referenced by the STS)
    - Ask the user if this is a first-time visit to this STS
    - If yes, no local account exists for this user
      - Ask the user for additional security/authentication and/or attribute information
      - Create a new account for the user
      - Following policies established for trusting security tokens from the STS, add necessary mapping information for this account (maps the user’s original security token UserID and issuer with the newly created local account)
      - Issue a local security token to the user
    - If no, ask the user to authenticate locally
      - Determine the user’s local account
      - Following policies established for trusting security tokens from the STS, add necessary mapping information for this account (maps the user’s original security token UserID and issuer with the local account)
      - Issue a local security token to the user
  - If this is not the first time that the user has used this security token to access a service in the local trust domain
    - Perform a mapping look-up between the user’s current security token and local UserID. Follow established policies for trusting security tokens from the STS when performing this operation.
    - Issue a local security token to the user
- User has the option to request a higher trust level than the one issued initially; Appropriate authentication technologies, procedures, and policies should be used and followed in this process.
## Description of the System Architecture Components

**Client**
The machine through which a user accesses the web portal. Authentication devices may be connected to this machine.

**Outside Web Portal Requestor**
A web portal outside of Domain A that attempts to access Domain A’s web services.

**Outside Web Service Requestor**
Requestor web services outside of Domain A that attempt to access Domain A’s web services.

**Web Portal**
The web portal of Domain A. Clients inside and outside Domain A may attempt to access this portal.

**Security Token Service (STS) – Authentication Service**
The service that validates credentials and issues security tokens. It references the Trust Authority for policies and definitions. Using this information, the STS maps trust levels and security requirements to values consistent with those of the local domain.

**Trust Authority**
The service that maintains Domain A’s trust level and trust group definitions. It also manages the domain’s token-granting and mapping policies.

**User Database**
This database stores information about users in Domain A.

**Authorization Service**
This service authorizes authenticated users for access to a domain’s web services. Authorization may be based on roles, permissions, and contexts.

**Pseudonym Service**
A pseudonym service is a Web service that maintains alternate identity information about principals within a trust realm or federation. The term principal, in this context, can be applied to any system entity, not just a person.*

**Attribute Service**
An attribute service is a Web service that maintains information (attributes) about principals within a trust realm or federation. The term principal, in this context, can be applied to any system entity, not just a person.*

**Web Service Resources**
This category includes general resource services, including the medical records web service.

*Definitions from*
HIPAA: Security Standards and Privacy Goals

The U.S. federal government created the Health Insurance Portability and Accountability Act (HIPAA) to establish conditions for healthcare reform regarding “administrative simplification” and the use of private patient information. Following these terms, the U.S. Department of Health and Human Services (HHS) developed standards, including security requirements, that healthcare institutions must implement in accordance with HIPAA. These standards aim to streamline healthcare system practices using IT solutions and to protect the integrity of personal health information.

HIPAA: Patient Privacy – The Final Privacy Rule

The following outline summarizes highlights of the U.S. Department of Health and Human Services Final Privacy Rule under HIPAA:

- Privacy Rule aims to ensure patient privacy protection without interfering with access to medical care or the quality of care. Most covered entities have until April 14, 2003, to comply; under the law, certain small health plans have until April 14, 2004, to comply
- Privacy Rule regulation covers health plans (insurance companies), healthcare clearinghouses (billing services, health management systems), and healthcare providers (hospitals, doctors’ offices)
- Medical records or other individually identifiable health information used or disclosed by a covered entity in any form (electronic, written, and oral) are covered by the Privacy Rule
- Providers and health plans must give patients clear written explanations of how covered entities may use and disclose their health information (i.e., the Notice Requirement)
- Prior written consent is optional for uses and disclosures related to treatment, payment, and healthcare operations by covered entities, but providers must make a good faith effort to obtain patients’ written acknowledgement of the notice; this eliminates barriers to patients’ access to care (i.e., during emergency situations)
- Patient authorization (using a common form) is required for each use or disclosure of information for any purpose unrelated to healthcare (i.e., those defined as “non-routine purposes”); the Privacy Rule recognizes the possibility of incidental disclosures, if reasonable safeguards exist and the minimum necessary requirements are being followed
- Covered entities must obtain written authorization before sending patients marketing materials, except during a face-to-face encounter or a communication involving a promotional gift of nominal value; doctors and covered entities may communicate freely with patients about treatment options
- Patients may access and request corrections of their medical records
- History of an entity’s most recent non-routine disclosures must be available to patients on request
- Parents generally have access to their children’s medical records; specifics are based upon state law; the provider has discretion in special cases
- Individuals have the right to file formal complaints regarding Privacy Rule violations
- Uses or disclosures of information should be limited to the minimum necessary for the purpose of the use or disclosure (for “non-routine purposes” only, excluding those for which authorization exists); health providers need access to complete patient records to provide quality medical care (for healthcare purposes); oral communication and consultation about patient medical records is permitted (i.e., conversations between doctors).

- For specific public responsibilities in limited circumstances, covered entities may disclose health information without individual authorization (i.e., to the FDA for public health purposes related to the quality, safety, and effectiveness of FDA-regulated products and for situations involving law enforcement and national security).

- Covered entities must have contracts with business associates to ensure that they follow the Privacy Rule’s requirements.

- Medical information that is not de-identified may be used for research purposes in the following circumstances (see Privacy Rule for more information):
  - Explicit patient authorization via a single, common consent form
  - Waiver receipt from an established review board
  - Under certain conditions and precautions, researchers may obtain limited data sets (not including directly identifiable information).

- An electronic signature standard must be followed when electronic signatures are used for secure communications of health data.

The complete Privacy Rule is available at the following web address: [http://www.hhs.gov/ocr/hipaa/finalreg.html](http://www.hhs.gov/ocr/hipaa/finalreg.html)

**HIPAA: Security Implementation Standards**

Security standards outlined by the HHS comprise four primary IT components: administrative procedures, physical safeguards, technical security services, and technical security mechanisms. The following excerpt from the HHS’s “Security and Electronic Signature Standards Proposal” issues a breakdown of the latter two HIPAA security components:


| Technical Security Services to Guard Data Integrity, Confidentiality, and Availability |
|--------------------------|-----------------------------|
| Requirement               | Implementation               |
| **Access control** (The following implementation feature must be implemented):** Procedure for emergency access. In addition, at least one of the following three implementation features must be implemented: Context-based access, Role-based access, User-based access. The use of Encryption is optional). | Context-based access. Encryption. Procedure for emergency access. Role-based access. User-based access. |
| **Audit controls**        |                             |
| **Authorization control** (At least one of the listed implementation features must be implemented). | Role-based access. User-based access. |
| **Data Authentication**   |                             |
**Entity authentication** (The following implementation features must be implemented: Automatic logoff, Unique user identification. In addition, at least one of the other listed implementation features must be implemented).

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic logoff.</td>
<td>Automatic logoff.</td>
</tr>
<tr>
<td>Biometric</td>
<td>Biometric.</td>
</tr>
<tr>
<td>Password</td>
<td>Password.</td>
</tr>
<tr>
<td>PIN</td>
<td>PIN.</td>
</tr>
<tr>
<td>Telephone callback.</td>
<td>Telephone callback.</td>
</tr>
<tr>
<td>Token</td>
<td>Token.</td>
</tr>
<tr>
<td>Unique user identification.</td>
<td>Unique user identification.</td>
</tr>
</tbody>
</table>

**Technical Security Mechanisms to Guard Against Unauthorized Access to Data That Is Transmitted Over a Communications Network**

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communications/network controls</td>
<td>Access controls.</td>
</tr>
<tr>
<td></td>
<td>Alarm.</td>
</tr>
<tr>
<td></td>
<td>Audit trail.</td>
</tr>
<tr>
<td></td>
<td>Encryption.</td>
</tr>
<tr>
<td></td>
<td>Entity authentication.</td>
</tr>
<tr>
<td></td>
<td>Event reporting.</td>
</tr>
<tr>
<td></td>
<td>Integrity controls.</td>
</tr>
<tr>
<td></td>
<td>Message authentication.</td>
</tr>
</tbody>
</table>

These technical security requirements necessitate secure, reliable information technology (IT) solutions and computer networking systems. Such security is essential for the storage and transmission of sensitive patient medical records. Although HIPAA and HHS mandate these security requirements, they do not explicitly specify implementations for attaining them. Each covered institution must select its own implementation of the rules.