SURROGATE TRUST NEGOTIATION:

SOLVING AUTHENTICATION AND AUTHORIZATION ISSUES IN DYNAMIC MOBILE NETWORKS

by

Tore L. Sundelin

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

July 2003

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Tore L. Sundelin

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Dr. Kent E. Seamons, Chair
Date	Dr. Charles D. Knutson
Date	Dr. Scott N. Woodfield

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Tore L. Sundelin in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

____________________	____________________
Date	Kent E. Seamons Chair, Graduate Committee

Accepted for the Department

____
David W. Embley
Graduate Coordinator

Accepted for the College

____
G. Rex Bryce
Associate Dean, College of Physical and Mathematical Sciences

ABSTRACT

SURROGATE TRUST NEGOTIATION: SOLVING AUTHENTICATION AND AUTHORIZATION ISSUES IN DYNAMIC MOBILE NETWORKS

Tore L. Sundelin

Department of Computer Science

Master of Science

This research describes a system that brings non-identity based authentication and authorization services to resource-constrained, wireless devices. This system extends the nascent technology of automated trust negotiation via a supple protocol that leverages both software proxies and autonomous security agents. The protocol allows portable devices from different security domains (i.e., with no pre-existing relationship) to securely interface with trust agents and centralized credential storage to make intricate, real-time access control decisions. The system is called surrogate trust negotiation (STN) as the sensitive and resource-intensive tasks of authentication are performed via proxy for the primary communicating devices by an out-of-band surrogate.

The viability of the proposed protocol is demonstrated by the results of a prototype system design and implementation. A performance benchmark of the prototype system executing a representative usage scenario provides preliminary evidence that the design is practical. This assertion is strengthened by a complimentary network simulation of a large-scale, STN-based system deployment involving several devices simultaneously requesting resource access and the requisite authentication and authorization services. The practical need of the technology offered in the STN protocol is asserted by its application in the healthcare industry, a large commercial sector currently exploring the increased digitization and transference of sensitive data.

ACKNOWLEDGEMENTS

I would like to thank my graduate advisor, without whom this work would have been infeasible, my graduate committee and research peers, without whom this would have been unrefined, my wife and family, without whom this would have been unimportant, and finally God, without whom this would have been impossible. I am humbled by and grateful for their cumulative guidance, support, and assistance.

Financial support for my research was provided by DARPA through AFRL contract number F33615-01-C-0336 and through Space and Naval Warfare Systems Center San Diego grant number N66001-01-18908.

The following research peers made concrete, substantive contributions to the noted sections of this research:

Adam Hess – Networking Protocol

Jason Holt – Security Provisions

Bryan Smith – System Design, Scalability Analysis

Dave Vawdrey – Applying STN to the Healthcare Industry

Tom Freestone – coined the phrase "surrogated trust negotiation"

TABLE OF CONTENTS

CHAPTER 1: FOUNDATIONAL MATERIAL 1

1.1. Motivation of Problem 1

1.2. Trust Negotiation 2

1.3. Reference Scenario 4

1.4. Thesis Outline 5

CHAPTER 2: SYSTEM OVERVIEW 7

CHAPTER 3: NETWORKING PROTOCOL 13

CHAPTER 4: SECURITY PROVISIONS 17

4.1. Secure End-to-End Protocol 18

4.2. Key Distribution 22

4.3. Threat Assessment 23

CHAPTER 5: SYSTEM IMPLEMENTATION 27

5.1. System Architecture and Class Design 27

5.2. Message Sequencing 30

CHAPTER 6: EMULATION DESCRIPTION AND TEST RESULTS 33

CHAPTER 7: SCALABILITY ANALYSIS AND TEST RESULTS 37

7.1. Scalability Scenario 37

7.2. Queuing Analysis 38

7.3. Network Simulation 41

7.4. Scalability Analysis Conclusions 46

CHAPTER 8: APPLYING STN TO THE HEALTHCARE INDUSTRY 47

8.1. Electronic Medical Records (EMRs) ... 47

8.2. Security Concerns for EMR Systems ... 48

8.3. Securely Transferring an EMR Using STN ... 49

8.4. Contributions of STN in EMR Systems ... 51

CHAPTER 9: RELATED WORK ... 53

CHAPTER 10: FUTURE WORK ... 55

CHAPTER 11: CONCLUSIONS ... 59

BIBLIOGRAPHY ... 61

TABLE OF FIGURES

Figure 1. A Medical Emergency 4

Figure 2. Surrogate Trust Negotiation 8

Figure 3. Topology Taxonomy 9

Figure 4. Networking Messages 13

Figure 5. Security Provisions 18

Figure 6. System Architecture 28

Figure 7. Primary Device Message Sequencing (Client Proxy) 30

Figure 8. Primary Device Message Sequencing (Server Proxy) 31

Figure 9. Secondary Device Message Sequencing 32

Figure 10. Prototype Trust Negotiation Exchange 33

Figure 11. Emulation Test Results 35

Figure 12. Alice goes to a football game 37

Figure 13. Queuing Analysis Calculations for the Main Gate 40

Figure 14. Queuing Analysis Calculations for the Secondary Gate 41

Figure 15. Network Topology for ns Simulation 42

Figure 16. Round Trip Times from Client Devices to their Respective Agents at the Main Gate 44

Figure 17. Average Round Trip Times from Client Devices to their Respective Agents 45

Figure 18. Dr. Jones Requests Sally’s EMR 50

CHAPTER 1: FOUNDATIONAL MATERIAL

1.1. Motivation of Problem

Interpersonal transactions are often contingent upon relevant attributes of the involved parties (e.g., nationality, age, job title, financial resources, etc.) and can be quite intricate and involved. In the digital world, such interactions have historically either been viewed as static identity-based schemes, handled out-of-band using alternative means, or simply avoided. One proposed solution for this problem of real-time, attribute-based digital interactions is called *automated trust negotiation* (see next section).

This research builds upon the foundation of trust negotiation to create an advanced authentication and authorization system compatible with the limited capabilities of many mobile computing devices. This application becomes particularly compelling in light of the recent proliferation of such devices, their associated usage models, and their intuitive contextualization as digital representatives of their respective users. Thus, the goal of this system is to enable people to utilize these devices to safely and efficiently perform sensitive transactions in their behalf in circumstances in which this was previously not possible.

The development of such a system presents significant obstacles, the first of which is the migration of trust negotiation and its various requirements to the mobile arena. Since trust negotiation is based on attributes within digital credentials (e.g., X.509 v3 certificates), it relies upon elements of public key cryptography that are often excessively burdensome on mobile devices. Also, because mobile network topologies are often unpredictable, this system must handle interactions between devices of mixed capabilities in varied infrastructure configurations. Furthermore, as a single entity or user can possess multiple mobile devices, the ability to securely identify specific devices with specific entities and moreover to allow these independent devices to safely share and access user credentials becomes a core issue. Additionally, since this research relies on the disclosure of credentials and other potentially sensitive resources, issues of confidentiality and authenticity must be addressed. Finally, as all networks are ultimately susceptible to compromise, the discovery of and recovery from potential breaches is also addressed.

1.2. Trust Negotiation

Historically, access control systems have predominantly targeted topologically static networks and closed systems. Therefore, access control has typically dealt with clearly defining a group of authorized users, establishing these users' capabilities with respect to system resources, and preventing unauthorized parties from accessing these resources. Yet the proliferation of dynamic, heterogeneous networks (such as the Internet) renders previous access control models invalid as they are incapable of governing resource access relative to a transient user base and resource cache.

This growing inability of existing access control models to meet the evolving needs of today's networks and open systems was the impetus for automated trust negotiation [35]. Trust negotiation is the bilateral, iterative disclosure of digital credentials between interacting parties in order to establish sufficient trust to complete a requested transaction. Unlike earlier identity-based models however, interaction is built upon the ability of the involved parties to demonstrate the possession of certain attributes as certified by digital credentials. This approach allows parties that have no preexisting relationship to confidently perform sensitive transactions in real-time as decisions are based on what a person (or device) has rather than who a person is. Furthermore, it also provides more semantically accurate decisions and increased privacy by focusing on the qualities that provide germane assurances concerning the parties involved rather than irrelevant or unnecessary information.

Credentials are digitally signed by a credential issuer and assert the veracity of the included attributes of the credential owner. They can attest to such varied characteristics as residency in a particular state, age, membership in a particular organization, employer, or any other conceivable descriptor. Therefore, transactions can trade semantics like "I will give you access to this file if you are John Jones" for semantics like "You can reserve this rental car if you have a current license, are insured, and are older than 25". Furthermore, due to the properties of public key cryptography, credentials are both unforgeable and verifiable. Thus, participants in a trust negotiation exchange can confidently validate the content of credentials as well as challenge for appropriate ownership by the presenting party.

If attribute credentials are the cogs of trust negotiation, *policies* provide the steam. Policies regulate access to sensitive resources such as services, data, credentials, or even policies themselves. Policies specify which credentials a party must possess in order to access a specific resource, thereby providing a means by which any user can be mapped to an authorized role in real-time. Thus, the definition of policies and their association with particular resources allow trust negotiation to automatically progress in a dynamic environment. Additionally, as all sides in a negotiation might have relevant policies, trust negotiation often occurs in a bilateral fashion with respective parties gradually fulfilling other parties' policies while iteratively making policy-based requests of their own.

A deeper discussion of trust negotiation is more involved than space permits. Past and present research deals with such topics as sensitive credentials [4][34][36], trust negotiation protocols [11], policy language requirements for trust negotiation [25], and negotiation strategy definition and interoperability [33][36].

1.3. Reference Scenario

The following scenario is intended to clearly illustrate both the core components of a STN system as well as its basic behavior and application. As such, it will be referenced throughout the body of the text to serve as context for the discussion of specific aspects or issues of the proposed system.

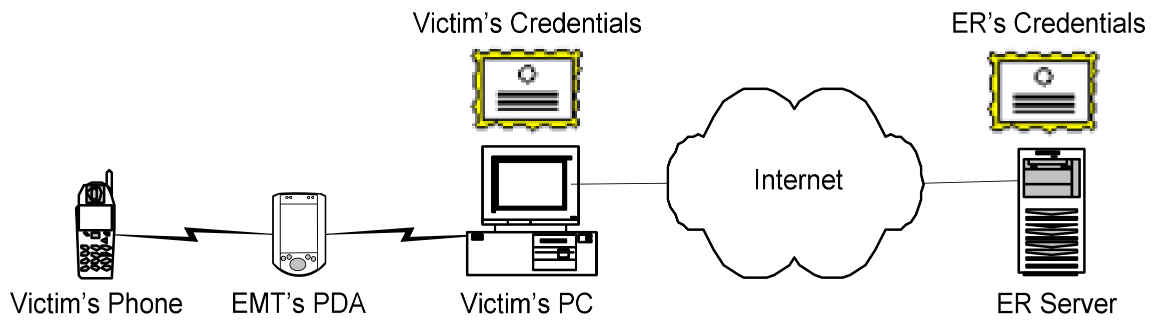


Figure 1. A Medical Emergency

Emergency medical personnel arrive at the scene of an automobile accident where they find a victim unconscious behind the wheel of a car. An EMT's PDA wirelessly discovers that there is a cell phone in the victim's car containing her emergency medical and physical identification information. The victim's cell phone transmits the location of authentication software located on her home PC. The EMT's PDA then contacts the victim's PC, pointing it towards an emergency room computer, which represents the EMT. These two machines establish mutual trust by requesting and exchanging digital credentials: the ER PC supplies the EMT's emergency medical credentials and the victim's PC supplies a credential asserting the authenticity of her medical data. The on-site medical personnel then access critical information contained in this data such as medical conditions and drug interactions while the hospital staff attemps to contact the victim's family.

1.4. Thesis Outline

The remainder of this thesis is organized as follows. The next chapter outlines the general architecture of the system, providing a lexicon of various components and a taxonomy of target usage models. Chapter 3 discusses in detail the network messaging that drives the system, and Chapter 4 presents an architecture that provides the necessary security characteristics. Chapter 5 describes the elements of an actual prototype implementation of a complete STN system. In Chapter 6, the empirical performance of this prototype is discussed; whereas, Chapter 7 analyzes the scalability and feasibility of large scale deployments using theoretical metrics. Chapter 8 describes the application of STN techniques to the evolving landscape of the medical industry. In Chapter 9 important related work is explored. And Chapters 10 and 11 present future directions, observations from the current research, and conclusions regarding the research area as a whole.

CHAPTER 2: SYSTEM OVERVIEW

Five major goals have shaped this research. The first, as discussed above, is the extension of automated trust negotiation to various mobile topologies. Building on this, the second goal is providing compatibility with the resource-constrained devices that typically comprise these networks.

The third is integration with a personal, centralized credential storage scheme, allowing each user to maintain a single repository for all of his credentials. All user devices share a pre-existing relationship (see section 4.2) with their personal credential server which allows them to authenticate to and request their associated security agent to negotiate trust on their behalf. This approach adds security and convenience to the system by avoiding the dangers of storing sensitive credentials on mobile devices and by allowing credential updates to be immediately accessible by all user devices.

The fourth goal of the system is the support of certain target usage models as discussed at the end of this section. The final goal is to produce a system independent of the idiosyncrasies of particular networking protocols or cryptographic mechanisms and thus one that is interoperable with the widest possible array of such choices for specific implementations.

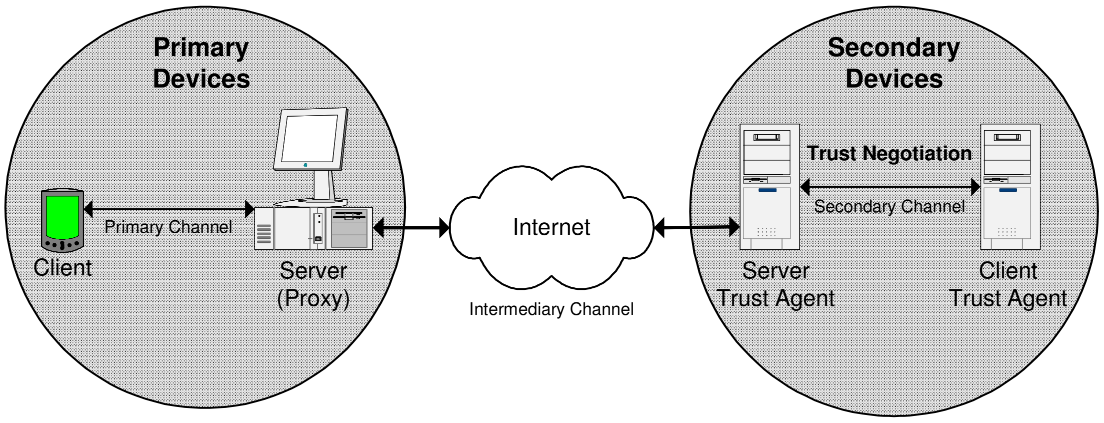


Figure 2. Surrogate Trust Negotiation

Any object for which access or use is regulated by a policy is referred to as a *protected resource*. A typical usage scenario (shown in Figure 2) is initiated when a user requests to access a resource belonging to another device. The requesting device is referred to as the *client;* the resource owner or steward, the *server*. These are not static designations however, and it is realistic to assume that both parties will routinely switch roles as one requests a resource from the other and vice-versa. Examples of protected resources are files, services, credentials, or even sensitive policies.

For the purposes of this research, those devices directly involved in requesting or providing resources (i.e., client and server) are called *primary devices*, and those indirectly involved in facilitating this exchange are *secondary devices*; moreover, the link connecting primary devices is called the *primary channel*, that connecting the secondary devices, the *secondary channel*, and that logically connecting a primary device and its associated secondary device, the *intermediary channel*. It is important to emphasize that these communication channels do not necessarily denote direct, physical links; these are strictly logical designations and there can be intervening devices such as proxies (which are explained below). Thus, once a resource request has been made via the primary channel, and the server determines the necessity of authenticating the client to a specific role, this request is forwarded via the intermediary channel to secondary devices. These secondary devices then negotiate trust in the secondary channel.

Secondary devices, or *trust agents,* are autonomous software modules that manage credentials, policies, and secret keys for use in trust negotiation for a specific entity. Within the context of the STN system, a *proxy* is any device that serves as an infrastructural intermediary between a primary device and its associated trust agent. All primary devices are configured to act as proxies; thus, whether acting as server or client in a particular negotiation, either can serve as a proxy should the need exist. Proxy selection is described in detail in Chapter 3.

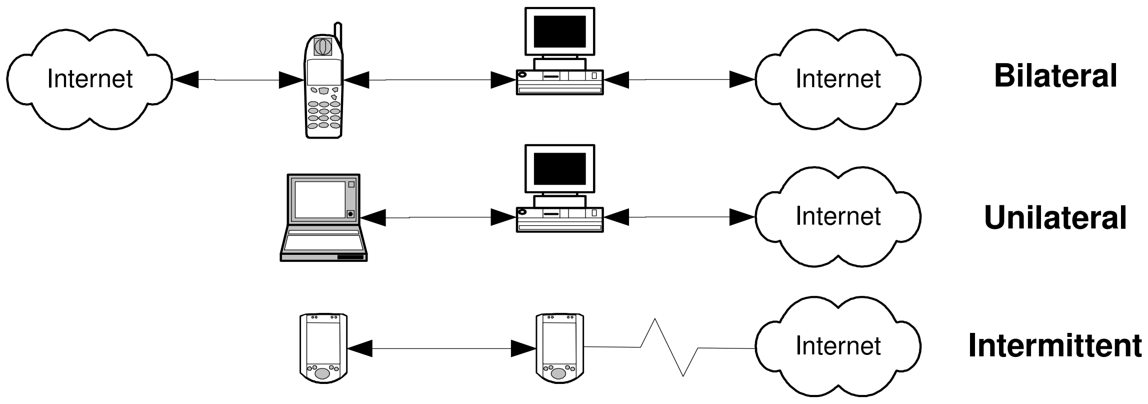


Figure 3. Topology Taxonomy

An analysis of the target usage models (see Figure 3) indicates that the most significant differentiating factor among them is Internet access, which is often necessary for a mobile device to communicate with its trust agent. Using this perspective, there are three general topologies that effectively categorize these models: topologies with *bilateral*, *unilateral*, or *intermittent* wired infrastructure access.

The first, *bilateral*, describes scenarios in which both primary devices have reliable, economical, and adequate bandwidth access to the Internet. The next, *unilateral*, describes any situation in which only one device has a consistent connection with sufficient bandwidth. This is the situation depicted in Figure 3, above. The final categorization, *intermittent*, or situations in which neither device has consistent access, is further dichotomized into two scenarios. The first is a situation in which the primary device has *foreknowledge* of a desired transaction before terminating or losing Internet access and the second is a completely *ad hoc* scenario. An example of an intermittent topology with foreknowledge is an airline passenger who negotiates trust with a gate agent using his PDA prior to boarding. Using surrogate trust negotiation, the passenger demonstrates ownership of sufficient age credentials to purchase in-flight liquor and provides vital medical information in the case of an emergency; then, the necessary cryptographic networking primitives are exchanged and cached for later in-flight use as necessary.

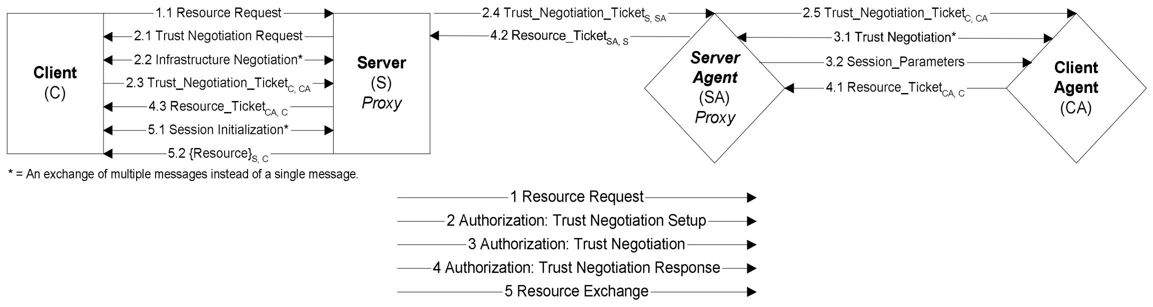
The remainder of this thesis explores the system in terms of a unilateral topology only due to its inherent simplicity and functional reducibility to the other topologies. That is to say, by having either primary device act as a proxy in a bilateral topology, it becomes functionally equivalent to a unilateral one; and, introducing a time lapse and crypto caching between trust negotiation and a resource exchange in a unilateral topology produces a functionally intermittent one. Thus, anything that is sufficient to accomplish the system goals in a unilateral topology theoretically addresses the needs of the other alternatives as well.

CHAPTER 3: NETWORKING PROTOCOL

This chapter describes a high-level, platform-independent approach to creating a surrogate trust system. Therefore, only the elements necessary to establish secure communications, perform trust negotiation, and exchange protected resources are outlined.

For simplicity, the networking messages are divided into in three distinct phases: *resource request*, *authorization*, and *resource exchange*. In the case of authorization, it is logical to further divide this phase into three sub-phases: *trust negotiation setup*, *trust negotiation*, and *trust negotiation response*. These phases and their composite messages appear in Figure 4.

An exchange begins with the resource request phase, in which the client simply requests a resource from the server, represented by the *Resource Request* message in Figure 4. The trust negotiation setup authorization phase begins when the server replies to the client by indicating that the requested resource is protected and that trust negotiation must be used for authentication (shown by the *Trust Negotiation Request* message). If the client is incapable of performing the trust negotiation protocol or chooses not to



Client (C)
Server (S) Proxy
Server Agent (SA) Proxy
Client Agent (CA)
1.1 Resource Request
2.1 Trust Negotiation Request
2.2 Infrastructure Negotiation*
2.3 Trust_Negotiation_Ticket C, CA
4.3 Resource_Ticket CA, C
5.1 Session Initialization*
5.2 {Resource} S, C
2.4 Trust_Negotiation_Ticket S, SA
4.2 Resource_Ticket SA, S
2.5 Trust_Negotiation_Ticket C, CA
3.1 Trust Negotiation*
3.2 Session_Parameters
4.1 Resource_Ticket CA, C
* = An exchange of multiple messages instead of a single message.
1 Resource Request
2 Authorization: Trust Negotiation Setup
3 Authorization: Trust Negotiation
4 Authorization: Trust Negotiation Response
5 Resource Exchange

Figure 4. Networking Messages

participate, this is communicated and the connection is broken. Otherwise, both hosts then communicate to decide which, if either, has the best access to the Internet in order to serve as a proxy. Figure 4 depicts this decision-making process as *Infrastructure Negotiation*.

A streamlined implementation of infrastructure negotiation can provide three choices that each primary device can use to express their negotiation desires and capabilities—*WANT*, *WILL*, and *WON'T*. WANT is used when a particular host has ready access to a reliable, efficient connection. WILL indicates that a host has infrastructure access, but there is some level of undesirability associated with its connection (e.g., high cost, low bandwidth, or unreliability). Hosts choose WON'T when infrastructure access is not available or when they are unwilling to act as the proxy.

Given a negotiation, there are 9 possible combinations of these values. Each individual choice is weighted as WANT = 2, WILL = 1, and WON'T = 0. Thus, whichever party has the maximum value serves as proxy for the exchange. In the case that both parties choose WON'T, the negotiation is abandoned and the link is terminated. When both parties indicate WILL or both select WANT, the server can choose which host will negotiate the infrastructure. For example, in scenarios where the server may benefit financially from the transaction, it may choose to handle the infrastructure to ensure completeness and timeliness. The case of intermittent access scenarios (e.g., expensive per minute access) can be signaled by sending a WILL response. Once the infrastructure decision has been made, the server ends infrastructure negotiation by sending the client a message identifying the selected proxy.

Following infrastructure negotiation, both client and server create a *Trust_Negotiation_Ticket* that is sent to their respective security agent. The ticket reliably notifies the security agent that its associated primary device desires to participate in trust negotiation for a specific resource. When the server acts as the proxy, the client sends a ticket to the server bundled with the location of the client's security agent. On the other hand, when the client serves as the proxy, the server creates a similar ticket but also includes an identifier for the requested resource and its associated policy. The nature of trust negotiation tickets is discussed further in Chapter 4.

Following the receipt of the appropriate trust negotiation ticket, the host that has been elected to function as the proxy connects to the server's trust agent. From the proxy host, the server's agent receives a message containing tickets from each primary device. The server's agent examines the ticket from its respective primary host and verifies its request to negotiate trust. Following this confirmation, the server's agent connects to the client's security agent and sends the appropriate trust negotiation ticket. The client security agent will then likewise verify the validity of its ticket.

After both security agents verify their associated primary hosts' intentions, the trust negotiation authorization phase begins. Since the server is the host that protects the resource, its security agent is responsible for initiating trust negotiation between the agents. As was briefly mentioned in section 1.2, the server's security agent begins the process by disclosing policies and/or credentials to the client's agent, which then responds likewise. This bilateral exchange continues until the server's agent deems that the policy – included in the Trust_Negotiation_Ticket – governing access to the resource has been satisfied or that the negotiation has failed. There are many factors that could contribute to a failure such as lack of necessary credentials, expired credentials, or the number of iterations exceeding a certain threshold .

Upon completion of successful trust negotiation, the client and server trust agents establish the cryptographic key material (see Section 7.2) necessary to create a secure link for releasing the protected resource in the primary channel. This key exchange is denoted by the *Session Parameters* message in Figure 4. Following this, the trust negotiation response authorization phase begins and this key material is sent back through the proxy to the respective primary devices in the form of *Resource_Tickets*. If trust negotiation was successful, the primary devices decrypt these tickets and use the session parameters that they contain to initialize the secure link (see Section 6.1), which is depicted by the *Session Initialization* exchange. When channel initialization is complete, the server can securely transmit the requested resource, and the sensitive transaction that began with the initial resource request is completed.

CHAPTER 4: SECURITY PROVISIONS

As discussed previously, a STN system can be logically separated into three communication channels – primary, intermediary, and secondary. In all three channels, integrity, authenticity, and confidentiality are necessary security goals. While a fourth often cited security characteristic, availability, is also significant, measures for its insurance are often so fundamentally different that it beyond the scope of the current treatment.

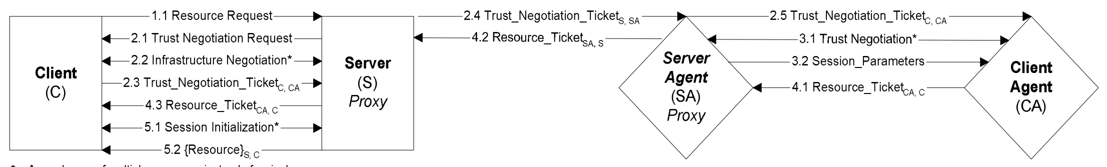
The reasons for these identified security goals are threefold. First, all are required to reliably initiate trust negotiation using the intermediate channel. Second, following trust negotiation, they ensure the safe delivery over the intermediary channel of key material for use in securing the primary channel. Finally, once this key material has been transported, it allows for the safe transmission of protected resources in the primary channel.

In order to provide these security characteristics to the networking protocol, it is proposed that a supporting architecture be comprised of two basic elements, which are covered separately in the next two sub-sections. The first is a protocol for creating secure, end-to-end communication channels. The second is the secure distribution of cryptographic key material requisite for the establishment of these channels. The correct establishment of these channels provides the system with all three target security characteristics.

4.1. Secure End-to-End Protocol

The end-to-end channels are defined as logical (see section 4), connectionless links between two hosts that guarantee integrity, authenticity, and confidentiality. The requirement of connectionless orientation is rooted in the volatility of wireless networks in general and potential exploits to the system specifically. It would be impractical, for example, to destroy an intermediary channel between a wireless primary device and its trust agent because of an unrecoverable packet resulting from either network conditions or malicious intervention.

To demonstrate what the establishment of these channels might look like, the case of establishing a secure link in the primary channel is considered. The end-to-end primary channel is initiated by first creating the necessary key material for the initialization parameters, which is generated and then transmitted in the *Session_Parameters* message in Figure 5. In this case, transmission actually occurs in the secondary channel at the conclusion of trust negotiation. Typical session parameters will often resemble the following:


Client (C)
1.1 Resource Request
2.1 Trust Negotiation Request
2.2 Infrastructure Negotiation*
2.3 Trust_Negotiation_Ticket$_{C, CA}$
4.3 Resource_Ticket$_{CA, C}$
5.1 Session Initialization*
5.2 {Resource}$_{S, C}$
Server (S) Proxy
2.4 Trust_Negotiation_Ticket$_{S, SA}$
4.2 Resource_Ticket$_{SA, S}$
Server Agent (SA) Proxy
2.5 Trust_Negotiation_Ticket$_{C, CA}$
3.1 Trust Negotiation*
3.2 Session_Parameters
4.1 Resource_Ticket$_{CA, C}$
Client Agent (CA)

* = An exchange of multiple messages instead of a single message.

{payload}$_{X, Y}$ = A message payload that is formatted for the end-to-end connection from host x to host y.
Trust_Negotiation_Ticket$_{C, CA}$ = CA_ID | {Resource_ID}$_{C, CA}$
Trust_Negotiation_Ticket$_{S, SA}$ = {Trust_Negotiation_Ticket$_{C, CA}$ | Trust_Negotiation_Role | Resource_ID}$_{S, SA}$
Session_Parameters = Sesssion_ID | Client_Write_Session_Key | Client_Authentication_Session_Key | Server_Write_Session_Key | Server_Authentication_Session_Key
Resource_Ticket$_{CA, C}$ = {Trust_Negotiation_Result | Session_Paramters}$_{CA, C}$
Resource_Ticket$_{SA, S}$ = {Resource_Ticket$_{CA, C}$ | Trust_Negotiation_Result | Session_Paramters}$_{SA, S}$

Figure 5. Security Provisions

Session_Parameters = {Session_ID,

Client_Write_Session_Key, Client_Authentication_Session_Key,

Server_Write_Session_Key, Server_Authentication_Session_Key}

Session_ID*: A value which uniquely identifies an open end-to-end protocol session. Thus, a single host can simultaneously maintain several open end-to-end links.

Client_Write_Session_Key*: A key used by the client to encrypt messages for the server, and in turn used by the server to decrypt those messages.

Client_Authentication_Session_Key*: A key used by both client and server to validate messages from the client.

Server_Write_Session_Key*: A key used by the server to encrypt messages for the client, and in turn used by the client to decrypt those messages.

Server_Authentication_Session_Key*: A key used by both client and server to validate messages from the server.

* While a generalized session ID and keys are shown, the actual composition and type are dependent upon the communication protocol and cipher suite used on the link. The prevailing convention of using distinct cryptographic keys is depicted as it ameliorates the effects of single key compromise and prevents reflection attacks.

Once the necessary session parameters have been generated and transmitted in the secondary channel, they are disseminated via the intermediary channel to the corresponding primary devices using resource tickets as explained in section 6.2. Using this key material, a secure session is then initialized in the primary channel as specified by the selected transmission protocol. This exchange is depicted by the *Session Initialization* messages in Figure 5.

Following session initialization, all transmitted messages are formatted according to the security provisions of the selected transmission protocol. For the purposes of this discussion, a simplified version of IPSec's Encapsulating Security Payload (ESP)[12] protocol is presented as it is capable of providing all of the target characteristics (i.e., connectionless sessions, integrity, authenticity, and confidentiality). These generalized cryptographic elements are also referenced later in a discussion of threats to the system (see section 6.3). However, an actual implementation of the system can use any established protocol which fulfills the proposed definition of an end-to-end link.

In general, end-to-end messages are formatted by wrapping the payload data with the necessary header information and then encrypting and authenticating the result. In Figure 5, messages formatted according to the end-to-end protocol are denoted by the syntax $\{\text{payload}\}_{\text{sender, recipient}}$. A formatted message typically had the following general structure:

$$\mathbf{End\_To\_End\_Message = Session\_ID \mid Sequence\_Number \mid Ciphertext\_Length \mid (Message\_Length \mid Message \mid Padding)_{Key\_Name} \mid Authentication\_Data_{Key\_Name}}$$

Session_ID: An identifier that allows a node to distinguish messages from different connections, as described above.

Sequence_Number: A counter used to prevent replay attacks. Note that all messages in a connectionless link need not arrive in order nor do they need to arrive at all. Furthermore, the sequence number, as well as an associated receive window, is typically maintained independently for each end of the connection.

Ciphertext_Length: The length of the ciphertext produced by $(\text{Data})_{\text{Key\_Name}}$.

$(\text{Data})_{\text{Key\_Name}}$: The output of the message encryption function on *data*.

Message_Length: The length of the actual message, excluding padding.

Message: The message payload.

Padding: Any padding required by $(\text{Data})_{\text{Key\_Name}}$.

$\text{Authentication\_Data}_{\text{Key\_Name}}$: The output of the keyed authentication function. The authentication function is called on the concatenation of the message elements listed above.

Thus, using the procedure described above, the security characteristics of the implemented connectionless protocol – in this case ESP – can be leveraged to protect communication and secure the channels. However, as the secondary channel is assumed secure by virtue of the utilized trust negotiation protocol (e.g., [11]), only the primary and intermediary channels need adhere to the proposed definition of an end-to-end link. Furthermore, the only necessary technical differences between these two channels then are the key distribution methodology (described next) and the hosts involved.

4.2. Key Distribution

In the case of the primary channel described above, the keys are generated and transferred in the secondary channel at the close of trust negotiation. Hence, this transmission leverages trust negotiation's provisions for authenticity, integrity, and confidentiality. Following this, the keys are distributed to their respective primary devices in the form of *Resource_Tickets*, which use the security provisions of the previously established end-to-end intermediary channels.

Therefore, in order to enable the secure exchange of a protected resource, the protocol depends upon the a priori existence of secure intermediary channels. Thus, while not a direct result of this research, their establishment also merits brief mention and discussion.

As an intermediary channel logically links a primary device and a trust agent, it represents a pre-existing relationship with certain unique characteristics that key distribution mechanisms can effectively leverage. A viable approach could follow the recommendations established by Stajano and Anderson in [30] or Balfanz et. al. in [3]. Both parties discuss protocols in which the desired communication is preceded by a form of pre-authentication during which cryptographic key material is either exchanged or pre-committed to. Hence, ensuing communications in the main wireless channel can then be both authenticated and confidential. Stajano advocates direct physical contact (e.g., via serial cable) in a process he calls “imprinting”, while Balfanz simply maintains that the transaction be with a physically identifiable party and authenticatable, such as via a point-to-point IrDA infrared device.

Either of these (or similar) approaches is acceptable and ostensibly rather convenient for most users wishing to imprint a primary device with the necessary keys to access its centralized trust agent and credential server. For example, given that the primary device and trust agent are both controlled by the user, session parameters can be generated by the trust agent as cryptographically strong random numbers and transferred directly to the primary device over a secure connection (e.g., using the USB cable between a Palm and a workstation). Then, once the necessary key material has been exchanged in this fashion, the session can easily be initiated following the provisions of the end-to-end link. Furthermore, following this same procedure for each individual imprinted device is also advisable in order to generate unique keys, which limits the potential damage caused by a single, compromised device (as explained in the following section).

4.3. Threat Assessment

All network systems are potentially vulnerable to four general types of attacks: interruption, interception, modification, and fabrication. As the issue of availability is elided from this thesis, interruption attacks (as they result in availability issues) are likewise excluded from this discussion.

Specific types of concerns for the STN system that are addressed include replay attacks and man-in-the-middle attacks. Due to the nature of the protocol, these attacks are examined in the context of abuse of the initially insecure primary channel. In addition to these more traditional concerns, the security implications of the loss/theft of an imprinted mobile device are also considered.

All data sent across the primary channel prior to initialization of the end-to-end session should be assumed to have been overheard by an attacker; thus, no sensitive exchanges should be conducted while the link is in this state. Furthermore, as data is sent in the clear at this point, it can also be maliciously modified or fabricated. An example of such an exploit would be for a man-in-the-middle to attempt to modify a resource request by a client or to masquerade and insert a counterfeit request. However, such attacks are thwarted since Trust_Negotiation_Tickets are formatted via the end-to-end link on the intermediary channel; hence, modified or counterfeit requests will be ignored by the associated trust agent upon an attempt to verify message integrity. Another major threat concerns various types of replay attacks. For example an attacker may try to replay a Trust_Negotiation_Ticket to gain unauthorized access to a protected resource or a Resource_Ticket to abuse resource access that was previously granted appropriately. However, the combination of sequence numbering and receive windows defeats such attacks as any message numbered less than the index of the last authentic message minus the window size is rejected. One interesting variation on this attack is a delayed Resource_Ticket that attempts to force a request exchange in an invalid timeframe. However, this attack could be averted by the proper utilization of a security protocol that includes support for message timestamping.

A final type of vulnerability to consider is that of a compromised mobile device, using such exploits as those described in [13]. An ameliorating factor on the extent of potential damage is the storage of credentials and their associated private keys on a centralized, geographically secure server. These keys never leave this machine, and thus, even if one mobile device is compromised, these private keys as well as imprinted relationships with other devices remain secure. Furthermore, termination from the security agent's side of an end-to-end link is trivial if a user suspects a device has been compromised and re-initialization likewise in the case that the device is later recovered.

One approach to the detection of a compromised device is the introduction of expiration on the end-to-end link of the intermediary channel. This requires periodic re-imprinting of devices and prevents extended exploitation of this relationship, presenting a tradeoff between security and convenience. Moreover, this approach allows for fine-grained control over the valid connection period of individual devices based on their characteristics and typical usage models. Alternative localized security measures on the mobile devices are the use of PINs, passwords, or biometric identification routines. A more unorthodox possibility is the association of a self-canceling imprinted key that allows a tamper-aware device to communicate compromise to the trust agent upon the acquisition of Internet access or at the first attempt to initiate trust negotiation. However, these systems are themselves vulnerable to an enterprising attacker and just provide additional lines of defense.

CHAPTER 5: SYSTEM IMPLEMENTATION

The project goal of interoperability with various networking or security protocols (see Chapter 2) played a significant role in many of the system design and implementation decisions. This focus is most clearly illustrated in the areas of object-oriented system architecture design and message sequencing. This chapter examines the characteristics of a STN system implementation from the perspective of these two design areas.

5.1. System Architecture and Class Design

The core of the STN software is comprised of three classes: the STNPrimaryDevice class, the EndToEndChannel class, and the STNPrimaryProxy class. These three classes cumulatively represent the basic functionality of a primary device in an STN negotiation. The behavior of a trust agent, on the other hand, is largely implemented in a modified TrustBuilder class. As previous literature has covered the design of the basic TrustBuilder class [35] for use in standard trust negotiation, its description is not included in this text. However, STN-specific modifications to the pre-existing software are noted in Figure 6.

As STN has application in a variety of conceivable wireless interactions, the class structure incorporates abstract classes for the modules that handle communication and security. Thus, a single primary device might have multiple derivatives of these basic classes to accommodate various negotiation scenarios and provide for future extensibility. For example, a single device might have both HTTP and FTP proxy classes as well as ESP and WTLS security implementations.

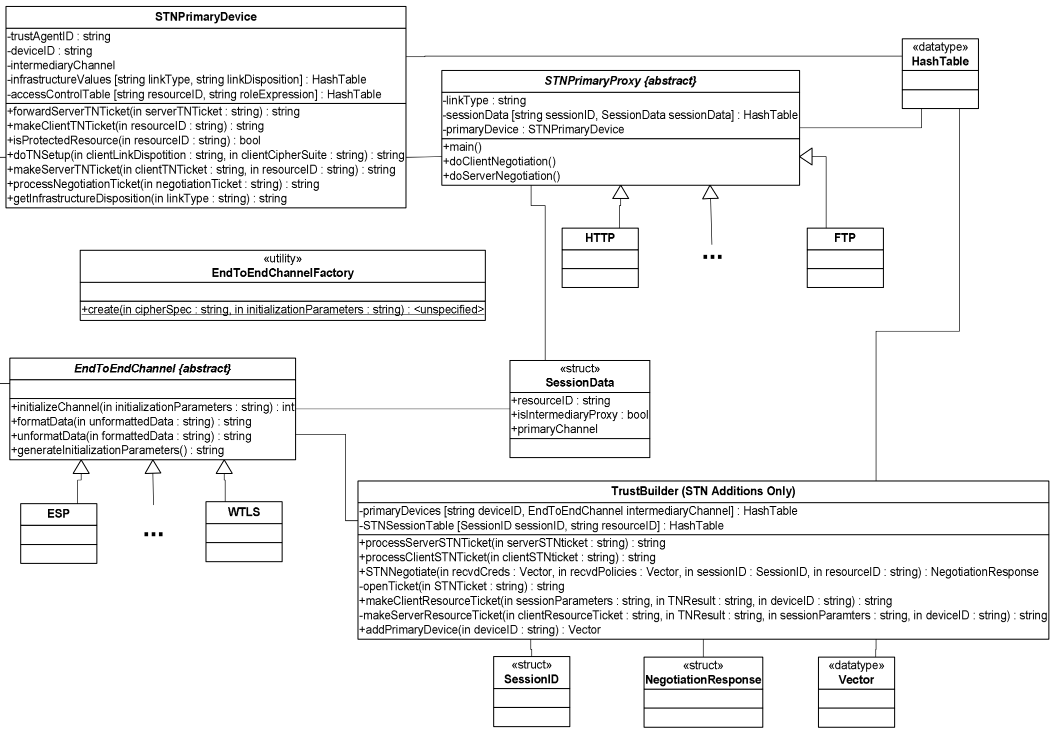

STNPrimaryDevice
-trustAgentID : string
-deviceID : string
-intermediaryChannel
-infrastructureValues [string linkType, string linkDisposition] : HashTable
-accessControlTable [string resourceID, string roleExpression] : HashTable
+forwardServerTNTicket(in serverTNTicket : string) : string
+makeClientTNTicket(in resourceID : string) : string
+isProtectedResource(in resourceID : string) : bool
+doTNSetup(in clientLinkDispotition : string, in clientCipherSuite : string) : string
+makeServerTNTicket(in clientTNTicket : string, in resourceID : string) : string
+processNegotiationTicket(in negotiationTicket : string) : string
+getInfrastructureDisposition(in linkType : string) : string
«datatype»
HashTable
STNPrimaryProxy {abstract}
-linkType : string
-sessionData [string sessionID, SessionData sessionData] : HashTable
-primaryDevice : STNPrimaryDevice
+main()
+doClientNegotiation()
+doServerNegotiation()
HTTP
...
FTP
«utility»
EndToEndChannelFactory
+create(in cipherSpec : string, in initializationParameters : string) : <unspecified>
EndToEndChannel {abstract}
+initializeChannel(in initializationParameters : string) : int
+formatData(in unformattedData : string) : string
+unformatData(in formattedData : string) : string
+generateInitializationParameters() : string
«struct»
SessionData
+resourceID : string
+isIntermediaryProxy : bool
+primaryChannel
ESP
...
WTLS
TrustBuilder (STN Additions Only)
-primaryDevices [string deviceID, EndToEndChannel intermediaryChannel] : HashTable
-STNSessionTable [SessionID sessionID, string resourceID] : HashTable
+processServerSTNTicket(in serverSTNticket : string) : string
+processClientSTNTicket(in clientSTNticket : string) : string
+STNNegotiate(in recvdCreds : Vector, in recvdPolicies : Vector, in sessionID : SessionID, in resourceID : string) : NegotiationResponse
-openTicket(in STNTicket : string) : string
+makeClientResourceTicket(in sessionParameters : string, in TNResult : string, in deviceID : string) : string
-makeServerResourceTicket(in clientResourceTicket : string, in TNResult : string, in sessionParamters : string, in deviceID : string) : string
+addPrimaryDevice(in deviceID : string) : Vector
«struct»
SessionID
«struct»
NegotiationResponse
«datatype»
Vector

Figure 6. System Architecture

At the core of the system design is the *STNPrimaryDevice* object. This class encapsulates the behavior and data that is common to all STN modules running on each respective primary device. Thus, it maintains a relationship with its respective trust agent (secondary device) as well as with an instance of the EndToEndChannel class, for secure communication with this agent. Furthermore, the STNPrimaryDevice class maintains a table of all local protected resources and their respective policies. It utilizes this data to make decisions about when and how to initiate trust negotiation based on incoming resource requests.

The *EndToEndChannel* class is designed to encapsulate all details concerning the secure communication channels that primary devices use during surrogate trust negotiation. Thus, it is aware of the cryptographic material utilized by these channels and the stipulations of their respective protocols (e.g., packet structure). Also, it is able to format and unformat incoming and outgoing data according to these details. These channels are capable of mixed pairings between the security protocol used and the selected cryptographic algorithms for implementing these protocols as arranged by the communicating devices (e.g., there could be one instance of an ESP channel using AES encryption and another using Triple DES). Thus, in order to accomplish this desired flexibility, these objects have been designed to inherit from a parent abstract EndToEndChannel class and to be instantiated from an associated factory class that creates them using the parameters supplied by the calling application.

Primary devices (i.e., STNPrimaryDevice objects) communicate with other devices through the use of *STNPrimaryProxy* objects. This class is designed to serve as a proxy for a specific networking protocol. For example, a primary device that serves web pages maintains a relationship with an HTTP-aware proxy that monitors all incoming HTTP requests and communicates with the STNPrimaryDevice object to determine if these requests require trust negotiation. Furthermore, this proxy also formats any outgoing primary device responses into valid HTTP-compliant messages.

Similar to the EndToEndChannel class, the root STNPrimaryProxy class is abstract and is intended to exist in various incarnations of subclasses according to system needs. It is important to note that each proxy is capable of handling multiple sessions (like in the case of a web server) and maintains and updates the necessary data for each. Moreover, each proxy can serve the role of either an STN client or server as necessary. Thus, if in addition to serving web pages, the primary device is browsing pages and receives an HTTP message stating that trust negotiation is required to access a page, the proxy interprets the request and makes the necessary calls to fulfill the request via the public methods of the STNPrimaryDevice class.

5.2. Message Sequencing

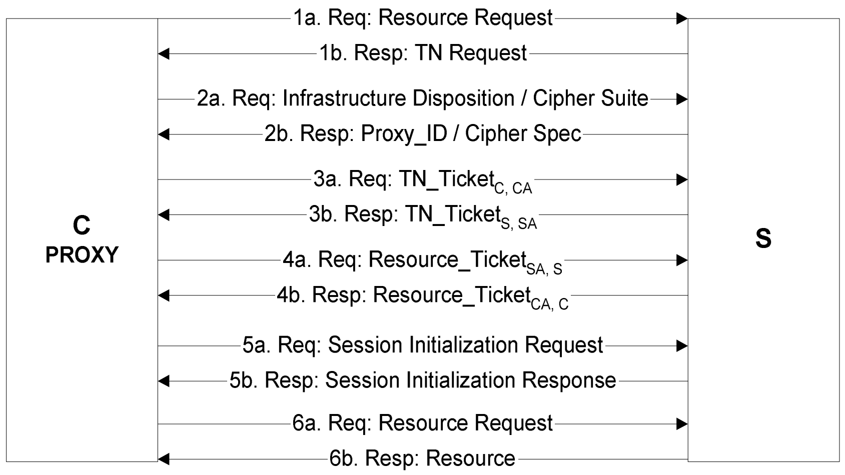

C
PROXY
S
1a. Req: Resource Request
1b. Resp: TN Request
2a. Req: Infrastructure Disposition / Cipher Suite
2b. Resp: Proxy_ID / Cipher Spec
3a. Req: TN_Ticket$_{C, CA}$
3b. Resp: TN_Ticket$_{S, SA}$
4a. Req: Resource_Ticket$_{SA, S}$
4b. Resp: Resource_Ticket$_{CA, C}$
5a. Req: Session Initialization Request
5b. Resp: Session Initialization Response
6a. Req: Resource Request
6b. Resp: Resource

Figure 7. Primary Device Message Sequencing (Client Proxy)

Many of today's most popular networking protocols adhere to a rigid request/response framework for message passing (e.g., HTTP, SOAP, SMTP, FTP, Java RMI, RPC, etc.). Thus, for compatibility, the STN message passing protocol also adheres to this convention. To illustrate this, Figure 7 depicts the message sequencing for primary channel scenarios in which the client device has been selected to serve as the proxy, such as the medical emergency example from section 1.3.

However, as Figure 7 demonstrates, while the request/response pattern does provide compatibility with several important protocols, compliance can lead to overhead. One example of this is the third message pair involving the Trust_Negotiation_Ticket that the client must send to the server agent to initiate trust negotiation. In this scenario, the client sends his TN ticket to the server, only to have that data inserted into the server ticket and then sent right back to him; the client then forwards this combined ticket on to the server agent. However, to both ensure message integrity in the intermediary channel (between the server and its agent) and to adhere to the request/response model, this inefficiency is unavoidable. This is not the case in unilateral scenarios in which the server acts as the primary channel proxy. Figure 8 illustrates this, as the third and fourth message pairs of the client proxy scenario are combined into a single third message pair.

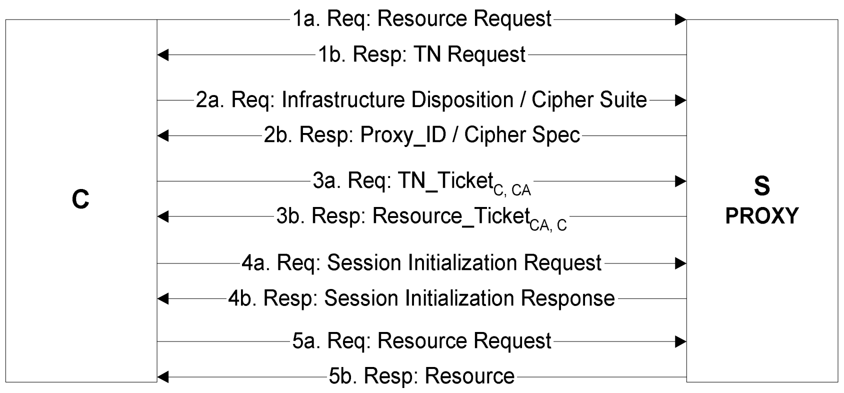

C
S
PROXY
1a. Req: Resource Request
1b. Resp: TN Request
2a. Req: Infrastructure Disposition / Cipher Suite
2b. Resp: Proxy_ID / Cipher Spec
3a. Req: TN_Ticket$_{C, CA}$
3b. Resp: Resource_Ticket$_{CA, C}$
4a. Req: Session Initialization Request
4b. Resp: Session Initialization Response
5a. Req: Resource Request
5b. Resp: Resource

Figure 8. Primary Device Message Sequencing (Server Proxy)

For completeness, a message sequencing diagram for the secondary channel is presented in Figure 9 below.

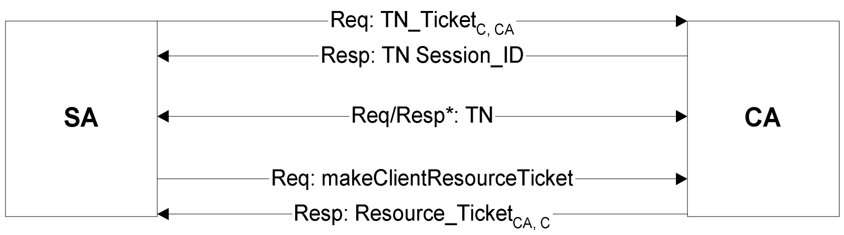

Req: TN_Ticket$_{C, CA}$
Resp: TN Session_ID
SA
Req/Resp*: TN
CA
Req: makeClientResourceTicket
Resp: Resource_Ticket$_{CA, C}$

Figure 9. Secondary Device Message Sequencing

CHAPTER 6: EMULATION DESCRIPTION AND TEST RESULTS

In order to analyze the performance of surrogate trust negotiation, a prototype system which emulates the functionality of the protocol was designed and implemented. Then, using this prototype, execution time measurements were observed and evaluated. These results provide a performance benchmark and support the feasibility of a commercialized STN-based system implementation.

The hardware core of the prototype system is comprised of two WiFi-enabled iPAQ handhelds running Microsoft's Pocket PC operating system, which serve as the primary devices. Thus, the physical and link layer of the primary channel is 802.11b. On top of this, basic TCP/IP sockets are used for communication between the primary devices.

The intermediary and secondary channels were implemented by extending the existing TrustBuilder [35] software. Essentially TrustBuilder (TB) was modified to behave as a secondary device, or trust agent. This was accomplished by extending TB to allow for direct communication among instances of TrustBuilder instead of requiring intervening network proxies. For simplicity, both the server trust agent and client trust agent were simultaneously run on a single Pentium 4 machine running Windows XP. The SOAP RPC protocol was used as a means of communication between negotiating trust agents as well as between primary devices and their respective trust agents.

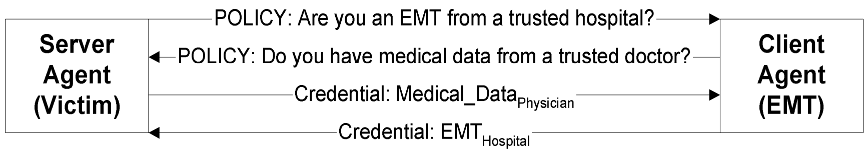

Server Agent (Victim)
POLICY: Are you an EMT from a trusted hospital?
POLICY: Do you have medical data from a trusted doctor?
Credential: Medical_Data$_{Physician}$
Credential: EMT$_{Hospital}$
Client Agent (EMT)

Figure 10. Prototype Trust Negotiation Exchange

The usage scenario was an adaptation of the medical emergency example from section 1.3. One handheld device acted as the victim's cell phone, or the STN server; the other, the EMT's PDA, or client. The trust negotiation for the scenario was the simple two-round, bilateral negotiation shown in Figure 10.

In order to get a fairly accurate estimate of the time required for a negotiation in a commercialized system, the prototype executed all five rounds of the STN protocol as stipulated in Chapter 3 – resource request, trust negotiation setup, trust negotiation, trust negotiation response, and resource exchange.

As the prototype was intended to merely serve as a proof-of-concept implementation and an empirical metric for performance analysis, certain delimitations on the integration of cryptographic primitives and system design were established. Since feasibility and performance (as opposed to functionality or robustness) were the focus of the analysis, an established security protocol was not integrated on any of the channels, but instead the inclusion of such was merely emulated. Thus, to gauge the performance impact of integrating a security protocol on the primary and intermediary channels, the steps of encrypting and authenticating the data were performed at each end of a secured exchange (such as in ESP for example), but the specific stipulations of ordering and message format of a formalized protocol were ignored and actual data was sent as cleartext. This was also true of the secondary channel, which just used SOAP over HTTP (as opposed to HTTPS for example). Furthermore, as the benefits of the OO design presented in the previous chapter (e.g., extensibility, code reuse) are only realized in a true development environment and do not impact performance, this structure was eschewed for a more simple procedural design in the prototype implementation. Hence, while the system is inappropriate for a true commercial implementation, its design does accurately simulate key performance characteristics while reducing the upfront implementation costs.

For the purpose of testing, the described scenario was performed ten times and the average execution time over these runs was calculated. In order to take into account the potentially misleading effects of caching, the STN applications and SOAP servers on the secondary devices were restarted as were the STN applications on the primary devices in between each run in the first five tests. Those results were then averaged with five additional runs during which only the STN applications on the primary devices were restarted. As can be seen from the results, these variations had no discernable effect in the overall execution time. The results of the tests are listed in Figure 11.

Test #	Execution Time (s)
1	3
2	7
3	6
4	4
5	3
6	4
7	4
8	7
9	4
10	6
AVE	**4.8**

Figure 11. Emulation Test Results

These results are very encouraging and good supporting evidence of the feasibility of a commercial implementation of a surrogate trust negotiation system from a performance standpoint. Given the currently accepted alternative, a mean execution time of 4.8 seconds for the emulation of the medical emergency scenario should be considered excellent performance. Even taking into account the additional overhead of the requisite changes for creating a commercially robust implementation – i.e., transitioning to Bluetooth-based networking, increasing the sophistication of the trust negotiation and credential verification, adding latency due to geographically disparate trust agents, and integrating an established security protocol - should keep the execution time well within comfortable operating parameters for the proposed scenario. That is to say, that even if an EMT arriving at the scene of an accident can automatically receive critical medical data within a minute or two (instead of five seconds), it is still more than an order of magnitude better than what is currently possible under existing mechanisms for accomplishing the same task.

CHAPTER 7: SCALABILITY ANALYSIS AND TEST RESULTS

This chapter explores the feasibility of a large scale deployment of an STN system (as opposed to the small scale emulation presented in the previous chapter) using a two-pronged scalability analysis. This study was performed by using statistical network queuing analysis [2] as well as by developing and running a STN-based network simulation on the ns-2 simulator [31].

7.1. Scalability Scenario

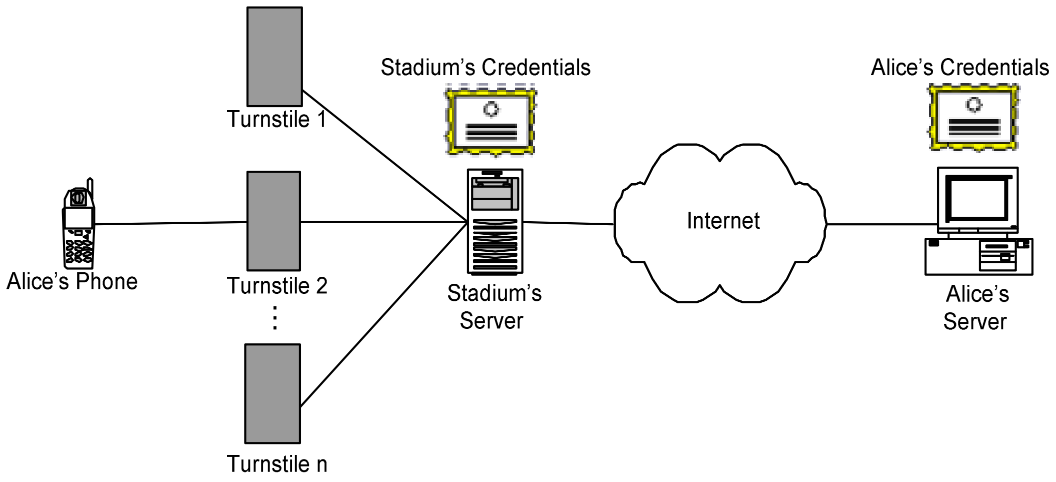


Figure 12. Alice goes to a football game

Alice, a State College student, is going to an away football game at State University. When she reaches the entrance to the stadium, the automated ticket machine requests a valid ticket and a student ID. In response, Alice takes her cell phone out of her purse and uses its Bluetooth radio to communicate with the turnstile. The turnstile communicates with the Trust Negotiation service on her phone and is immediately directed to her home computer, where all of Alice's digital credentials are stored and protected. Alice's home PC and the turnstile's trust agent negotiate trust, the PC verifying that it is talking to a State U approved trust agent and the turnstile verifying that the cell phone is owned by a current college student, who has sufficient funds to purchase a ticket. Upon verification, the turnstile unlocks and transmits a seating assignment to the phone, and Alice is quickly on her way.

7.2. Queuing Analysis

A common tool of network analysis is queuing theory. In the broadest terms, queuing theory is the theoretical analysis of waiting lines. Thus, as many performance scenarios in the computing world can be modeled in an identical mathematical fashion to people waiting in line for service (e.g., instruction evaluation in a CPU or packet forwarding in a network switch), queuing theory has become a common, well-recognized tool of the trade.

In this section, the queuing theory model is applied to the aforementioned scenario in order to theoretically evaluate the performance of an STN implementation which deals with large numbers of parties simultaneously requesting access to a protected resource in a single network. In order to achieve a level of realism, network condition data is based upon attendance figures collected at an actual University football stadium. Thus, for this test, it is assumed that Alice is a Utah State University student going to an away football game at Brigham Young University's LaVell Edwards Stadium. According to BYU's published data, this stadium has 65,000 seats, 42 ticket gates (turnstiles), and six entrances. It is assumed that there are seven ticket turnstiles at each of the six entrances. The calculation used for service requests, or in this case the number of spectators wishing to purchase tickets and enter the stadium, was taken from the average game attendance from 2000 – 2002, which was 60,376. It is presumable that the spectators entering the stadium were not evenly distributed over all the entrances. As a result, it is assumed that the main, or highest volume, entrance admits about 1/4 of the total spectators and the least-used entrance admits only 1/10 of the total spectators. Finally, traffic is modeled using a statistically random arrival pattern of spectators at the 7 turnstiles of each gate with an exponential service time for the spectators entering the stadium (i.e., an M/M/7 queue using Kendall notation). This is admittedly a simplification of actual game attendance patterns, in which large numbers of fans typically arrive at or around kickoff.

To perform the calculations, the T_S, or average service time, from emulation testing results (see Figure 11 above) was rounded up to an approximation of 5 seconds. In order to perform a comparative analysis of the entire stadium, an average arrival rate, λ, was calculated and then utilized in separate queuing analyses of both the highest volume (main gate) and lowest volume (secondary gate) entrances to the stadium as described below.

$T_S = 5 \text{ s}, \ \lambda = 1.39, \text{N} = 7$

$$\rho = \frac{\lambda T_S}{N} = .992$$

$$\text{K} = \frac{\sum_{I=0}^{N-1} \frac{(N\rho)^I}{I!}}{\sum_{I=0}^{N} \frac{(N\rho)^I}{I!}} = \frac{1+6.94+24.08+55.71+96.66+134.16+155.18}{1+6.94+24.08+55.71+96.66+134.16+155.18+153.85} = .75$$

$$\text{C} = \frac{1-K}{1-\rho K} = .98$$

$$T_r = \frac{C}{N}\frac{T_S}{1-\rho} + T_S = 92.5 \text{ s}$$

$$r = \text{C}\frac{\rho}{1-\rho} + \text{N}\rho = 128.46 \approx 129$$

Figure 13. Queuing Analysis Calculations for the Main Gate

For the average arrival rate at the main entrance, it is assumed that 15,094 spectators (or 25% of all attendees) arrive over a 3 hour span, making λ equal an average of 1.39 spectators per second (see Figure 13). Based on T_S and λ, average utilization, or ρ, is 0.992, or nearly at full capacity. Thus, using the number of servers, N (i.e., number of turnstiles at each gate), and utilization, the mean time a spectator spends waiting and being serviced (T_r) equals 92.5 seconds or roughly 1.5 minutes.

In a similar fashion, these same calculations can be applied to a queuing analysis of the least used (secondary) entrance gate. However, for this gate, it is assumed that only 6,038 spectators, or 1/10 of the total spectators, arrived over a 3 hour span, making λ equal 0.559 spectators per second. Thus, using these numbers, the time required to gain admittance (T_r) is calculated to be a little over 5 seconds at this low volume gate (See Figure 14).

$T_S = 5$ s, $\lambda = .559$, N = 7

$$\rho = \frac{\lambda T_S}{N} = .399$$

$$K = \frac{\sum_{I=0}^{N-1} \frac{(N\rho)^I}{I!}}{\sum_{I=0}^{N} \frac{(N\rho)^I}{I!}} = \frac{1 + 2.79 + 3.89 + 3.62 + 2.52 + 1.41 + .66}{1 + 2.79 + 3.89 + 3.62 + 2.52 + 1.41 + .66 + .26} = .98$$

$$C = \frac{1-K}{1-\rho K} = .033$$

$$T_r = \frac{C}{N} \frac{T_S}{1-\rho} + T_S = 5.04 \text{ s}$$

$$r = C\frac{\rho}{1-\rho} + N\rho = 2.81 \approx 3$$

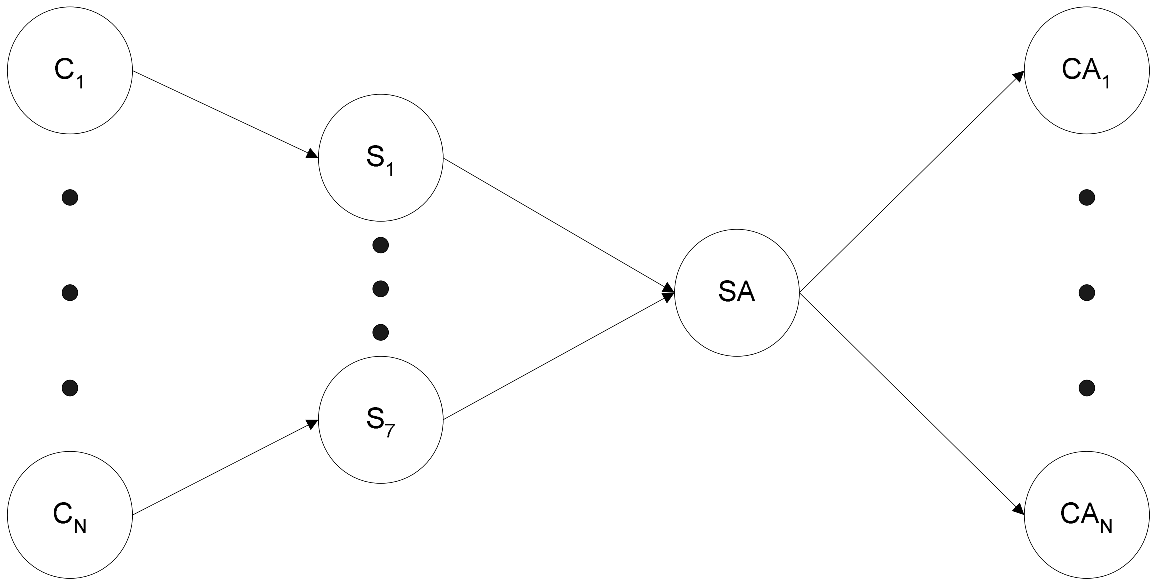
Figure 14. Queuing Analysis Calculations for the Secondary Gate

7.3. Network Simulation

Another common form of theoretical network analysis is a network simulation test. While there are many options for such a test, it is often desirable to use a widely known and thoroughly tested simulation environment so that the behavior of the test is well understood and reproducible by independent researchers. Thus, the network simulation of STN was conducted using the popular ns-2 Network Simulator. This simulator package is a byproduct of 14 years of correlated effort spearheaded by the USC School of

Engineering in conjunction with efforts at U.C. Berkeley and Carnegie Mellon and funded by both DARPA and NSF. ns-2 is a widely used simulation environment and is often utilized in publicized research.

The football scenario of the previous section was simulated by the network topology shown in Figure 15. It consists of N clients, 7 servers, and N client agents. In these simulations, N corresponds to *r* from the queuing analysis, which corresponds to the average number of spectators waiting and being served at each of the gates as calculated above. Also, each of the client devices in the simulation has a respective client agent while all the servers share a single stadium server agent. To simulate traffic, a TCP agent



There are N clients (C), 7 servers (S), and N client agents (CA). Each client has a different client agent, and all the servers share the same server agent. Also, each client is sequentially serviced by an open server in a round robin fashion upon requesting service.

Figure 15. Network Topology for ns Simulation

was associated with each client, and each client agent was attached to a TCP/Sink to receive the generated packets. Additionally, an application layer was simulated by an FTP module attached to each TCP agent.

The link between the clients and servers is a single-slot Bluetooth connection with a data rate of 115.2 Kbps and a delay of 1.3 seconds. This delay corresponds to an approximation of the time required to process a resource request, perform the infrastructure negotiation, and generate Trust_Negotiation_Tickets when data flows in the upstream direction (i.e., from the primary devices to the secondary devices). This delay additionally simulates the processing of Resource_Tickets, initialization of a secure channel, and transmission of the requested resource in the downstream direction.

The link between the servers and the server agent is a T1 connection with a data rate of 1.54 Mbps and a delay of 10 ms. The delay on this link simulates the processing of a Trust_Negotiation_Ticket upstream and the formation of a Resource_Ticket downstream. The link between the server agent and the client agents is a DSL connection with a data rate of 800 Kbps and a delay of 1 second. The delay on this link simulates the processing of a Trust_Negotiation_Ticket, performing trust negotiation, and generating session parameters upstream and creating a Resource_Ticket downstream.

A battery of six simulations was performed using this topology. In each of these tests, the number of servers remained constant at seven, but the number of clients and client agents was linearly increased with each successive test. The first was run with five clients, which would simulate the volume expected at the least-used entrance (*r* was actually calculated to be three in the queuing analysis, but was approximated as five for the simulation in order to maintain a pattern of using multiples of five for client numbers). The results of this first simulation were rather unsuprising. Each client was caculated to have a round trip time of approximately 4.5 seconds, which was two times the sum of the described delays – in other words each client encountered no additional queuing delays when requesting service. This is as expected as each server has at most one simultaneous client associated with it (i.e., the number of servers is greater than the number of clients requesting service).

In ensuing simulations, the number of clients and client agents continued to increase until a maximum of 120 was reached – which was an approximation of the calculated *r* of 129 at the main gate. The round trip times from this final simulation are shown in Figure 16.

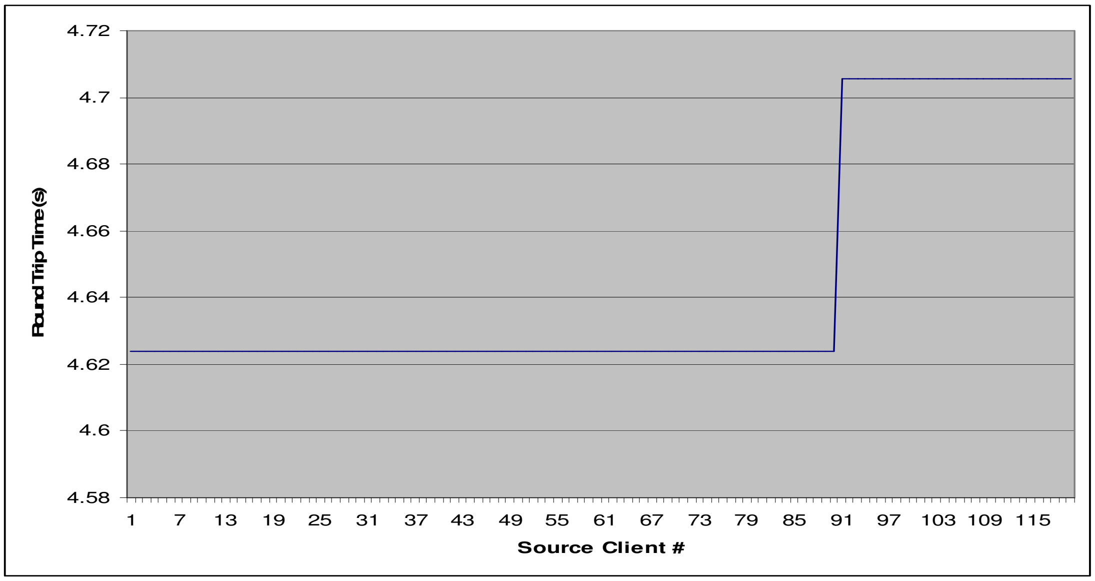


Figure 16. Round Trip Times from Client Devices to their Respective Agents at the Main Gate

In this final simulation (Test #6), Figure 16 shows that clients #1 - 90 acted identically to the clients from the secondary gate simulation (Test #1), while clients #91-

120 possessed a larger round trip time. Thus, at the time that client #91 sent its request, either the server or the server agent began to queue requests, which added queuing delays to the simulation. From this point, the introduced queuing delay then continued to affect the remaining service requests.

Test #	Average Round Trip Time (s)
1	4.62
2	4.62
3	4.62
4	4.62
5	4.62
6	4.64

Test #1 had 5 clients and 5 client agents. Test #2 had 10 clients and 10 client agents. Test #3 had 15 clients and 15 client agents. Test #4 had 30 clients and 30 client agents. Test #5 had 60 clients and 60 client agents. Test #6 had 120 clients and 120 client agents. Each test also had 7 servers.

Figure 17. Average Round Trip Times from Client Devices to their Respective Agents

The average round trip times for all six simulations are shown in Figure 17. This shows that there were no perceived queuing delays until the sixth test, in which the total number of clients and client agents was 120. The original hypothesis anticipated the queuing delay gradually increasing as the number of clients exceeded the number of servers. However, the experimental data reveals that the load placed on the servers was not sufficient to cause queuing delays until there were more than 60 clients and 60 client agents simultaneously present in the test system.

7.4. Scalability Analysis Conclusions

The results of the scalability analysis provide further support for the performance feasibility of a commercial implementation of a surrogate trust negotiation system. A mean residence time of 92.5 seconds for the queuing analysis of the football scenario at a high volume entrance is desirable (i.e., the average non-ticket holding fan only has to wait only about 1.5 minutes for entrance into a crowded football game). In contrast, existing schemes can typically yield at least an order of magnitude worse performance as a spectator waits several minutes in a ticket purchasing line at a large sporting venue. Additionally, the simple network simulation further corroborates the positive results of the queuing analysis – yielding a maximum spectator wait time of less than 5 seconds for the outlined test conditions.

However, these positive results are neither conclusive nor comprehensive. Such assertions require further, more in-depth analysis of large scale scenarios with high numbers of simultaneous clients and servers. The presented tests and results are intended only to serve as preliminary, foundational conclusions, as well as to act as touchstones for ensuing testing and analysis.

CHAPTER 8: APPLYING STN TO THE HEALTHCARE INDUSTRY

In modern healthcare delivery systems there is a critical need for timely access to accurate patient medical information. Comprehensive and cost-effective patient care depends on the provider's ability to readily access a patient's test results, prior treatment notes, current prescriptions, and so forth. However, under current systems, the lack of such access to this information may delay diagnosis and result in improper treatment and increased costs [25].

8.1. Electronic Medical Records (EMRs)

Electronic medical record (EMR) systems promise to solve many of the existing problems associated with information flow among healthcare providers. Advances in technology are enabling communication of medical record information among provider institutions, record repositories, and individual practitioners on a global scale [14].

While global access to patient record information is necessary to the future of healthcare, strict legal and ethical responsibilities exist for protecting patient privacy. Specific security requirements include establishing protocols to guarantee confidentiality and integrity of identifiable patient information and implementing authentication and access control schemes to prevent unauthorized disclosures of sensitive information [4]. Unfortunately, while most facilities are equipped with up-to-date medical technology, many rely on antiquated communication networks lacking the security measures required to protect sensitive patient information [20].

However, trust negotiation can address current authentication and authorization limitations in traditional security systems by creating a framework in which two unrelated parties may establish the trust sufficient to perform sensitive transactions. Moreover, surrogate trust negotiation is useful for meeting the security requirements of systems in which patients and healthcare providers use portable devices to act as their digital representatives in the exchange of personal EMR data. This chapter briefly outlines the issues involved in safely exchanging digital medical records and presents STN as a viable solution.

8.2. Security Concerns for EMR Systems

When it comes to personal medical records, patients have a high expectation of confidentiality and often an implicit belief that such information will be used exclusively to facilitate effective care. These records contain some of the most sensitive information about an individual, including data on fertility and abortions, emotional problems and psychiatric treatment, substance abuse, physical and sexual abuse, and genetic predispositions to certain diseases) [26]. Naturally, patients are inherently distrustful of storing and communicating such information electronically and making it available somewhere in “cyberspace” [17]. While various delivery networks and legal jurisdictions have differing regulations regarding access rights to medical information, it is generally agreed that patients’ care providers must be allowed consensual access to any information relevant to a patient’s treatment [27].

One of the fundamental obstacles in meeting legal requirements for privacy is the lack of a comprehensive security framework that addresses the need for authentication (i.e., mapping a party to its attributes) and authorization (i.e., mapping attributes to resource access) [8]. The most common threat to the security of electronic patient records does not come from outside attackers as some have supposed, but from the inappropriate accessing of information by “authorized” providers [25]. These internal security risks include accidental disclosures and poorly controlled secondary usage [26]. One of the major implications of HIPPA – the U.S.’s Health Insurance Portability and Accountability Act - is that healthcare institutions must track all instances of access to sensitive data, including who was involved, under what circumstances, and for what purpose [8]. Thus, reliable authentication mechanisms are essential to make these requirements feasible [17].

Similar to non-healthcare based authentication systems, a significant problem arises when no prior relationship exists between an access-granting service and a party requesting EMR data. For example, consider the common situation of healthcare provider *A* requesting a patient’s EMR from hospital *B,* where *B* cannot authenticate *A’s* request because they are strangers (i.e., they have no foreknowledge or preexisting relationship). Classical, identity-based access control schemes are inherently incapable of handling such situations.

8.3. Securely Transferring an EMR Using STN

Trust negotiation solves the problems associated with classical authentication and authorization schemes by allowing individuals outside a local security domain to safely access sensitive data and services. Moreover, as described in section 1.2, a successful trust negotiation provides certain guarantees about the attributes of the interacting parties, an important provision for protecting against the unauthorized access or accidental disclosures of sensitive information as discussed above.

To illustrate a common situation in which surrogate trust negotiation could be utilized to provide a secure framework for the transference of an EMR, see Figure 18 below. This depicts a scenario in which a physician, Dr. Jones, wishes to access the EMR of a new patient, Ms. Sally White, who is visiting from out of town.

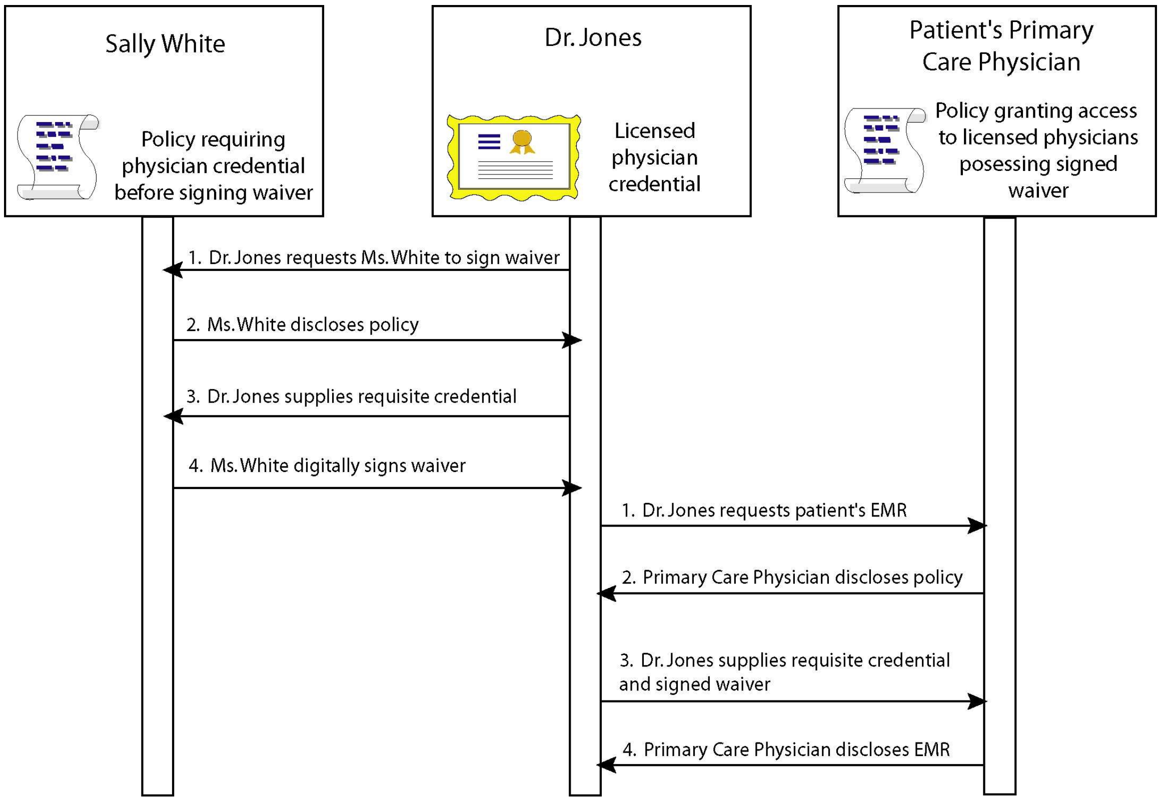

Sally White
Policy requiring physician credential before signing waiver
Dr. Jones
Licensed physician credential
Patient's Primary Care Physician
Policy granting access to licensed physicians posessing signed waiver
1. Dr. Jones requests Ms. White to sign waiver
2. Ms. White discloses policy
3. Dr. Jones supplies requisite credential
4. Ms. White digitally signs waiver
1. Dr. Jones requests patient's EMR
2. Primary Care Physician discloses policy
3. Dr. Jones supplies requisite credential and signed waiver
4. Primary Care Physician discloses EMR

Figure 18. Dr. Jones Requests Sally's EMR

Before providing care, Dr. Jones requires a waiver authorizing treatment and allowing access to Sally's EMR. To accomplish this, Dr. Jones' office computer sends a request to Sally's PDA for a digitally signed waiver expressing her consent. Sally's PDA communicates via the Internet with the trust agent on her home computer, which responds to Dr. Jones' request by disclosing a policy stating that the PDA will not sign a waiver without verifying the requestor's physician credential. Once Dr. Jones' respective trust agent supplies the necessary credential, Sally's PDA digitally signs the waiver.

Next, Dr. Jones' computer requests Sally's EMR from the office of her primary care physician. Before releasing the record, the primary care physician must not only recognize Dr. Jones as a licensed physician, but also confirm his "need to know" [9] – this need can be verified by the signed waiver previously created by Sally's PDA. As explained in previous sections, this authentication is accomplished through the provisions of the surrogate trust negotiation protocol. Thus, this relatively complicated exchange can happen in an automated, secure fashion, greatly expediting the issues of patient consent, physician identification, and record authentication that are inherent to the situation.

8.4. Contributions of STN in EMR Systems

The global expansion of EMR systems among healthcare institutions is absolutely essential for improving patient care, medical research, and public health. In the past, the implementation of EMR systems has been hindered by inadequate security mechanisms, including deficiencies in controlling access to sensitive data. Trust negotiation is a new approach for authentication and authorization among healthcare information systems with no pre-existing relationship. Surrogate trust negotiation extends these security benefits to systems involving resource-constrained mobile computing devices that may be used to exchange individual patient medical records. Thus, surrogate trust negotiation has enormous potential to improve the current state of security in healthcare information systems by providing an efficient, flexible, and secure communication framework for the exchange of highly sensitive data in a variety of desirable usage models.

CHAPTER 9: RELATED WORK

Given that an autonomous, remote trust agent is central to the STN protocol, research involving software agents within the context of security issues has proven germane. One such example is that proposed by He et al. [8] in which they implement a novel, agent-based PKI (Public Key Infrastructure) utilizing a proprietary communication language dubbed KQML (Knowledge Query and Manipulation Language). In the paper, the authors create autonomous agents that are capable of rudimentary certificate management tasks as well as inter-agent communication and interaction. The needs of the STN protocol extend beyond the scope of this research, however, as it does not broach access control issues such as authentication to roles in open networks comprised of transient, unaffiliated users.

In the field of mobile security research, there is a surprising dearth of viable protocols providing authorization capabilities for resource-constrained, mobile devices that supported diverse usage models and a flexible infrastructure. However, excellent treatments of isolated issues related to the core space do exist. The closest approximation of the overall STN system is described by Burnside et. al in [6]. Their research outlines a three-tiered, proxy-based SPKI/SDSI network that provides support for heterogeneous, wireless devices. In their implementation, the authors opt to force all communication to flow through all three tiers of the network and to require persistent infrastructure access. While this is valid for their target usage models, it is inadequate for many of today's mobile networks such as those in which only some parties have access to wired infrastructure or those in which access was not persistent.

On the other end of the networking spectrum, the research by Balfanz, et al. in [3] outlines an ad-hoc security scheme with important similarities to the STN system as well as a plethora of great ideas. The authors establish a confidential, authenticated link by performing out-of-band pre-authentication in an auxiliary channel. While the notion of pre-authentication (first proposed in the wireless arena by Stajano in [29]) is crucial to the efficiency and agility of STN, the proposed architecture's independence from network infrastructure is insufficient for the form of real-time, dynamic authorization and centralized credential management critical to the desired results of this research.

In an interesting proposal concerning access to persistent online storage by mobile devices, Villate et al. [32] speak of the combined use of proxies and agents—a pairing which also proved beneficial in the design of the STN system. Additionally, similar to Neuman [20], the convention of utilizing cryptographic tickets for cross-realm identification is an important inclusion in this research. However, Kerberos' connectivity requirements and identity-based authentication are unworkable in the identified target networks of the STN system.

CHAPTER 10: FUTURE WORK

There are many performance-centric topics that can be explored in future research. One such is session resumption in the primary channel. This has the potential to reduce the overhead of creating a new trusted session between two hosts by reusing parts of a previous session. One target scenario for this update is one in which the server and client of one negotiation switch roles in another. In addition, hosts may also see performance gains by caching authenticated roles and/or session parameters from previously negotiated sessions. A parallel concept is the ability to pre-authenticate to a set of roles. This is particularly useful in situations where a host is aware that in the near future trust will be required but Internet access will not be available—such as intermittent topologies with foreknowledge.

Issues involving the software interface and user interaction could also be addressed by future research. For instance, consider scenarios in which a user desires to release a protected resource even though its access control policy has not been satisfied or trust negotiation was not attempted. The ability for a human to override the system may be desirable when the user implicitly trusts the other entity. An example of this would be a husband releasing a newly drafted family finance spreadsheet that doesn't yet have an updated policy to his wife's PDA. In such cases, the model should allow the server's user to acknowledge that trust negotiation has failed and then allow him to manually override the system and proceed given the possession of proper identifying information such as a PIN. A popular analogue to this is the detection of an invalid credential (e.g., one that has expired) during an SSL connection negotiation. This detection triggers a browser dialog box that indicates to the user that the server's SSL certificate is not valid, but gives him the ability to proceed with the transaction nevertheless.

Another interesting problem to be addressed is resource policy placement. In the current model both the server and its security agent act as repositories for policies. Since a single user may have resources that are replicated across several mobile devices, storing a central policy for these in the security agent would seem ideal. A key benefit to a monolithic, centralized approach is the absence of possibly conflicting policies of differing strictness that present points of compromise to the system. On the other hand, consider a user creating a new resource on a mobile device. If the mobile device does not possess the capacity to locally generate access policies, trust negotiation for access to this resource is unworkable. In such cases, a decentralized policy approach appears to be best. For this research, issues of policy development, distribution, and storage have not been developed; instead, previous research in the wired paradigm has been leveraged to assert the basic feasibility of associating a role with a resource locally and associating related policies with this role centrally. In the future, the tradeoffs inherent in a hybrid approach to mobile policy management can be evaluated; such a system would allow each device and its respective security agent to generate and maintain a list of access control policies and resources in tandem.

A few key development areas also remain to be more fully explored prior to releasing a commercially viable STN implementation. First, the current prototype does not incorporate an established cryptographic protocol for protected data transfer in the primary channel. As was explained in Chapter 6, the key provisions of ESP were simply emulated for performance-based testing, but the complete protocol was not utilized.

Furthermore, security in the secondary channel was omitted in the current release. Thus, an updated release is currently under development which utilizes a proprietary ESP distillation in the primary and intermediary channels and runs SOAP over HTTPS in the secondary channel. Additionally, this updated release also adheres to the object-oriented design described in Chapter 5. A full-blown commercial implementation will require incorporation of a tested ESP implementation or an alternative protocol with the identified security characteristics outlined in Chapter 4. Secondly, as is stated in Chapter 7, further in-depth testing of performance benchmarks and scalability simulations would more definitively demonstrate the viability of a large scale deployment of an STN system such as the one described in the simulation scenario section of the same chapter.

CHAPTER 11: CONCLUSIONS

At its core, this research addresses the issue of a single entity being securely represented by multiple resource constrained devices in various sensitive exchanges and sundry usage models. Therefore, a user possessing a cell phone, a PDA, a laptop, and a portable MP3 player that are all capable of wireless communication can effectively leverage these devices to carry on sensitive transactions in a number of conceivable environments. Moreover, these transactions can occur with other devices with which the user's device has no pre-existing relationship.

In order to facilitate these transactions, a flexible model is presented that effectively leverages the combined capabilities of network proxies, software agents, and modern cryptographic systems. Central to this solution are portable devices that are individually capable of filling the role of a streamlined networking proxy in order to compensate for topological idiosyncrasies and deficiencies often present in today's mobile networks. This model can furthermore perform dynamic, role-based authentication and authorization in mobile infrastructures as well as in the stationary environments addressed by previous research in automated trust negotiation systems. Additionally, the highly sensitive and resource intensive task of public key cryptography that is integral to a centralized, credential-based system is offloaded to trust agents located on more secure, sedentary devices; thus, the system allows even computationally lightweight devices to effectively participate.

The results of a simple emulation and simulation system testing begin to lay the empirical foundation necessary to assert that STN be a viable commercialized protocol capable of the aforementioned applications and usage models. Moreover, the presentation of STN's integration into the emerging field of electronic medical records points to an expanding field with significant need for the existence of such a system. Thus, the combination of the development of a surrogate trust negotiation protocol, performance testing, and analysis of desirable potential applications establish this research as significant progress in the maturation of effective, new technology in the rapidly evolving research space of secure mobile transactions.

BIBLIOGRAPHY

[1] M. Abadi, M. Burrows, C. Kaufman, B. Lampson, "Authentication and Delegation with Smart-cards," *Technical Report 67,* DEC Systems Research Center, October 1990.

[2] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, P. Haldar, M. Handley, A. Helmy, J. Heidemann, P. Huang, S. Kumar, S. McCanne, R. Rejaie, P. Sharma, K. Varadhan, Y. Xu, H. Yu, and D. Zappala, "Improving Simulation for Network Research," *Technical Report 99-702*, University of Southern California, March 1999.

[3] D. Balfanz, D. Smetters, P. Stewart and H Wong, "Talking to Strangers: Authentication in Ad-Hoc Wireless Networks," *Network and Distributed System Security Symposium Conference Proceedings*, San Diego, CA, February 2002.

[4] R. Barrows, Jr., P. Clayton, "Privacy, Confidentiality, and Electronic Medical Records," *Journal of the American Medical Informatics Association (JAMIA)*, vol. 3, no. 2, pp. 139–148, March/April 1996.

[5] P. Bonatti, P. Samarati, "Regulating Service Access and Information Release on the Web," *7th ACM Conference on Computer and Communications Security*, Athens, Greece, November 2000.

[6] M.Burnside, D. Clarke, T. Mills, S. Devadas, and R. Rivest, "Proxy-Based Security Protocols in Networked Mobile Devices," *Proceedings of the 17th Symposium on Applied Computing (SAC 2002)*, March 2002.

[7] T. Dierks, C. Allen, The TLS Protocol Version 1.0, *RFC 2246*, January 1999.

[8] A. Dwivedi, R. Bali, M. Belsis, R. Naguib, P. Every, and N. Nassar, “Towards a Practical Healthcare Information Security Model for Healthcare Institutions,” *Proeedings of the. 4th Conference on Information Technology Applications in Biomedicine (ITAB 03)*, Birmingham, United Kingdom, pp. 114–117, April 2003.

[9] M. Epstein, M. Pasieka, W. Lord, S. Wong, and N. Mankovich, “Security for the Digital Information Age of Medicine: Issues, Applications, and Implementation,” *Journal of Digital Imaging*, vol. 11, no. 1, pp. 33–44, February. 1998.

[10] Q. He, K. Sycara, T. Finin, “Personal Security Agent: KQML-Based PKI,” *International Conference on Autonomous Agents*, Minneapolis, MN, May 1998.

[11] A. Hess, J. Jacobson, H. Mills, R. Wamsley, K. Seamons, B. Smith, “Advanced Client/Server Authentication in TLS,” *Network and Distributed System Security Symposium*, San Diego, CA, February 2002.

[12] S. Kent, IP Encapsulating Security Payload (ESP), *RFC 2406*, July 2002.

[13] Kingpin and Mudge, "Security Analysis of the Palm Operating System and its Weaknesses Against Malicious Code Threats," *10th USENIX Security Symposium*, Washington, D.C., August 2001.

[14] I. Kohane, P. Greenspun, J. Fackler, C. Cimino, and P. Szolovits, “Building National Electronic Medical Record Systems Via the World Wide Web,” *Journal of the American Medical Informatics Association (JAMIA)*, vol. 3, no. 3, pp. 191–207, May/June 1996.

[15] H. Krawczyk, “The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is {SSL}?),” *21st Annual International Cryptology Conference*, Santa Barbara, CA, August 2001.

[16] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," *RFC 2104*, February 1997.

[17] K. Mandl, P. Szolovits, I. Kohane, "Public Standards and Patient Control: How to Keep Electronic Medical Records Accessible but Private," *British Medical Journal (BMJ)*, vol. 322, pp. 283–287, February 2001.

[18] S. Miller, C. Neuman, J. Schiller, and J. Saltzer, "Kerberos Authentication and Authorization System," *Project Athena Technical Plan*, Section E.2.1, Massachusetts Institute of Technology, October 1988.

[19] R. Needham, M. Schroeder, "Using Encryption for Authentication in Large Networks of Computers," *Communications of the ACM*, vol.21 no.12, pp.993-999, December 1978.

[20] NetMotion Wireless, Inc., "HIPAA Security for Wireless Networks," http://www.netmotionwireless.com/assets/netmotion security hipaa.pdf, 2001.

[21] C. Neuman, "Proxy-Based Authorization and Accounting for Distributed Systems," *13th International Conference on Distributed Computing System, IEEE*, Pittsburgh, PA, May 1993.

[22] B. Patel, J. Crowcroft, "Ticket Based Service Access for the Mobile User," *Proceedings of the Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, Budapest, Hungary, September 1997.

[23] R. Pereira, R. Adams, "The ESP CBC-Mode Cipher Algorithms," *RFC 2451*, November 1998.

[24] G. Poupard, J. Stern, "Security Analysis of a Practical 'on the fly' Authentication and Signature Generation," *International Conference on the Theory and Application of Cryptographic Techniques (Eurocrypt 1998),* Espoo, Finland, May 1998.

[25] D. Rind, I. Kohane, P. Szolovits, C. Safran, H. Chueh, G. Barnett, "Maintaining the Confidentiality of Medical Records Shared Over the Internet and the World Wide Web," *Annals of Internal Medicine*, vol. 127, no. 2, pp. 138–141, July 1997.

[26] T. Rindfleisch, "Privacy, Information Technology, and Healthcare," *Communications of the ACM*, vol. 40, no. 8, pp. 92–100, August 1997.

[27] R. Schoenberg, C. Safran, "Internet Based Repository of Medical Records that Retains Patient Confidentiality," *British Medical Journal (BMJ)*, vol. 321, pp. 1199–1203, November 2000.

[28] K. Seamons, M. Winslett, T. Yu, B. Smith, E. Child, J. Jacobson, H. Mills, L. Yu, "Requirements for Policy Languages for Trust Negotiation," *Third International Workshop on Policies for Distributed Systems and Networks (POLICY 2002)*, Monterey, CA, June 2002.

[29] F. Stajano. "The Resurrecting Duckling – what next?" *Security Protocols, 8th International Workshop Proceedings,* April 2000.

[30] F. Stajano, R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks," *Security Protocols, 7th International Workshop Proceedings*, April 1999.

[31] W. Stallings, "Queuing Analysis," http://www.williamstallings.com/StudentSupport.html, 2000.

[32] Y. Villate, A. Illarramendi, E. Pitoura, "Keep Your Data Safe and Available While Roaming," *Mobile Networks and Applications,* vol. 7, no. 4, August 2002.

[33] W. Winsborough, K. Seamons, V. Jones. "Automated Trust Negotiation," *DARPA Information Survivability Conference and Exposition*, Hilton Head, SC, January 2000.

[34] W. Winsborough, N. Li, "Towards Practical Automated Trust Negotiation," *Third International Workshop on Policies for Distributed Systems and Networks (POLICY 2002)*, Monterey, CA, June 2002.

[35] M. Winslett, T. Yu, K. Seamons, A. Hess, J. Jacobson, R. Jarvis, B. Smith, L. Yu, "Trust Negotiation on the Web," *IEEE Internet Computing,* vol. 6, no. 6, November/December 2002.

[36] T. Yu, M. Winslett, K. Seamons, "Supporting Structured Credentials and Sensitive Policies through Interoperable Strategies for Automated Trust Negotiation," *ACM Transactions on Information and System Security (TISSEC),* vol. 6, no. 1, February 2003.