Decentralized Trust-Based Access Control for Dynamic Collaborative Environments

William Joseph Adams

submitted to the faculty of

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Engineering

March 31, 2006

Blacksburg, VA

Keywords: Access Control, Dynamic Collaboration, Trust Management, Wireless Security

Decentralized Trust-Based Access Control for Dynamic Collaborative Environments

William J. Adams

Abstract

The goal of this research was to create a decentralized trust-based access control (TBAC) system for a dynamic collaborative environment (DCE). By building a privilege management infrastructure (PMI) based on trust, user access was determined using behavior grading without the need for pre-configured, centrally managed role hierarchies or permission sets. The PMI provided TBAC suitable for deployment in a rapidly assembled, highly fluid, collaborative environment.

DCEs were assembled and changed membership as required to achieve the goals of the group. A feature of these environments was that there was no way of knowing who would join the group, no way of refusing anyone entry into group, and no way of determining how long members would remain in the group. DCEs were formed quickly to enable participants to share information while, at the same time, allowing them to retain control over the resources that they brought with them to the coalition.

This research progressed the state of the art in the fields of access control and trust management. The Trust Management System developed through this research effectively implemented a decentralized access control scheme. Each resource owner independently evaluated the reputation and risk of network members to make access decisions. Because the PMI system used past behavior as an indication of future performance, no a priori user or resource configuration was required.

Dedication

This work is dedicated to my wife, Lisa, and my two daughters, Analise and Amanda. The sacrifices they have made over the course of my career in the military led to me being selected for school. The support they have given me over the last three years enabled me to complete my studies. Without their love and encouragement, I would not have been able to overcome the challenges, negotiate the obstacles, and accomplish the deeds that I have. I will always love you all and will always strive to be worthy of you.

Acknowledgements

I would like to acknowledge the many people whose support and encouragement made it possible for me to complete this research. First, I must thank my advisor, Dr. Nathaniel J. Davis, IV for his guidance and help throughout my three years at Virginia Tech. I would also like to thank Dr. Scott Midkiff for stepping up and agreeing to be my co-chairman, as well as the other members of my committee Dr. William Baumann, Dr. Edward Green, and Dr. Jung-Min Park for their advice and support of this work.

Second, I would like to highlight the support I received from George Hadjichristofi as I took a vague idea and pounded it into the research presented here. Ryan Thomas also deserves a note of thanks for helping work out the kinks in the details and by being a good sounding board when my idea machine got stalled. Another big thank you goes to the other Army students at Virginia Tech who gave me encouragement, hints, tips, and course corrections as I took my journey to a Ph.D.

Finally, this section would not be complete without acknowledging the support of the United States Army and the Department of Electrical Engineering and Computer Science at the United States Military Academy for their funding support and for allowing me this opportunity.

Table of Contents

Dedication ..... **iii**

Acknowledgements ..... **iv**

Table of Figures ..... **viii**

List of Tables ..... **x**

List of Acronyms ..... **xi**

1 Introduction **1**

1.1 Problem Statement 2

1.2 Background and Motivation 2

1.3 Research Contributions 3

1.4 Methodology Overview 4

1.5 Results and Conclusions 4

2 Literature Review **6**

2.1 Dynamic Coalitions 6

2.2 Access Control 11

2.2.1 Role-based Access Control 11

2.2.2 Credential-based Access Control 16

2.2.3 Role-based Access Control in Dynamic Networks 19

2.2.4 Access Control Challenges in Dynamic Environments 20

2.3 Incentives 22

2.4 Trust 26

2.4.1 Trust Types 28

2.4.2 Reputation-based Trust Management 29

2.4.3 Trust-based Access Control in Wireless Networks 32

2.5 Conclusion 34

3 System Design 35

3.1 System Overview 35

3.1.1 Access Control in a Dynamic Collaborative Environment 36

3.1.2 Nodal System Security Architecture 37

3.1.3 Identifying Trust Management System Requirements 38

3.1.4 Elements from the Key Management System 41

3.1.5 Elements from the Intrusion Detection System 43

3.2 Implementing Trust 45

3.2.1 Trust Profiles 48

3.2.2 The Trust Store Module 50

3.2.3 The Reputation Scaling Module 56

3.2.4 The Risk Assessment Module 60

3.2.5 Trust Thresholds 62

3.3 Simulation Design 64

3.3.1 Behavior Modeling 64

3.3.2 Gaming the System 67

3.4 System Metrics 68

3.5 Summary 70

4 Analysis 72

4.1 Verification: Goals and Objectives 73

4.2 Trust Store 73

4.2.1 General Testing 74

4.2.2 Environmental Impacts 78

4.2.3 Contributions and Conclusion 79

4.3 Reputation Scaling 80

4.3.1 General Testing 81

4.3.2 Inter-network Mobility Testing 84

4.3.3 Environmental Impacts 86

4.3.4 Contributions and Conclusion 90

4.4 Risk Assessment 91

4.4.1 General Testing 92
4.4.2 Environmental Impacts 93
4.4.3 Conclusion 95
4.5 Validation: Goals and Objectives 95
4.5.1 Operational Scenario 96
4.5.2 Test Environment 98
4.6 System Testing 102
4.6.1 Success Metric 103
4.6.2 Cost Metric 106
4.6.3 Environmental Impacts 107
4.7 Conclusions 109
5 Conclusions 112
5.1 Significant Contributions 113
5.2 Research Accomplishments 114
5.3 Future Work 116
5.4 Concluding Thoughts 117
Appendix A: Simulation Environment 118
References 121
Vita 133

Table of Figures

Figure 2-1 High Assurance Guards in a Coalition Environment 7
Figure 2-2 Role-based Access Control Model Hierarchy 12
Figure 2-3 Basic Role-based Access Control Model 13
Figure 2-4 Transitive and Associative Trust 27
Figure 2-5 Trust Constructs 28
Figure 3-1 System Security Architecture 37
Figure 3-2 Registering a Security Association. 42
Figure 3-3 Dissolution Process 43
Figure 3-4 Composition of a Behavior Observation 44
Figure 3-5 Implementation of Trust Types and Constructs 45
Figure 3-6 Trust Constructs with Information Flows 46
Figure 3-7 Trust Management System Architecture 47
Figure 3-8 The Relationship of Trust Profiles to Trust Tendencies 49
Figure 3-9 Format of an Atomic Behavior Record 52
Figure 3-10 Introduction Process 54
Figure 3-11 Examples of Trust Threshold Use 63
Figure 3-12 Finite State Machine Behavior Model 65
Figure 3-13 Analysis of Behavior Model Settings 66
Figure 3-14 Example Comparison of Scenario Efficiencies versus Costs 70
Figure 4-1 Trust Store Management Algorithm Comparison 75
Figure 4-2 Trust Store Sizing Test 77
Figure 4-3 Effects of Trust Store Size on Efficiency and Cost 78
Figure 4-4 The Hysterisis Effect of the 3Win Method 81
Figure 4-5 Performance of Reputation-Scaling Mechanisms in an “Unreliable Node” Scenario 83
Figure 4-6 Performance of Reputation-Scaling Mechanisms in a “Smear” Scenario 83
Figure 4-7 Inter-Network Mobility Effects on Reputation Scaling 84

Figure 4-8 Inter-Network Mobility with Collusion 86
Figure 4-9 Reputation-Scaling Sensitivity to Network Density 87
Figure 4-10 Reputation-Scaling Sensitivity to General Peer Behavior 88
Figure 4-11 Reputation-Scaling Sensitivity to Network Mobility........ 89
Figure 4-12 Risk Assessment Module Efficiency in Completely Connected Networks........ 93
Figure 4-13 Risk Assessment Efficiency in Changing Behavior Situations........ 94
Figure 4-14 Operational Scenario Map........ 97
Figure 4-15 Format of Trust Information Transactions........ 101
Figure 4-16 Components of the Success Metric 104
Figure 4-17 Success Metric Components of the Base System Test........ 105
Figure 4-18 Comparison Base and Trust Management System Success Rates 105
Figure 4-19 Comparison of Cost Factor Between TMS and Base System........ 106
Figure 4-20 Comparison of TMS Success Rate to Base Case System 107
Figure 4-21 Sensitivity of System Efficiency to Network Density 108
Figure 4-22 Effects of Peer Behavior on Efficiency and Cost........ 109

List of Tables

Table 3-1 Example Initialization Values Based on Trust Profiles... 50
Table 3-2 Behavior Model State Change Probabilities.. 65
Table 3-3 Behavior Reporting Based on Behavior Type and State 67
Table 4-1 Coalition Locations and Resources 98
Table 4-2 Simulation Attraction Point Parameters 99
Table 4-3 Simulation Parameters.. 100
Table 4-4 Group Mobility Simulation Parameters.. 100
Table 4-5 Probability and Duration of Resource Requests in a Simulated Collaborative Environment.. 103
Table A-1 Mapping of Test Vignettes to Objectives 120

List of Acronyms

AA – Authorization Authority

ABR – Atomic Behavior Record

ACL – Access Control List

ARNG – Army National Guard

CA – Certificate Authority

CRL – Certificate Revocation List

DAC – Discretionary Access Control

DCA – Distributed Certificate Authority

DCE - Dynamic Collaborative Environment

DCP - Dynamic Coalition Problem

EFS – Ebay Feedback System

FI – Feedback Item

FIFO – First In First Out

GRI – Global Risk Index

HAG – High Assurance Guard

ICS – Incident Command System

IDS – Intrusion Detection System

IO – Information Officer

IRT – Incident Response Team

KMS – Key Management System

LDAP – Lightweight Directory Access Protocol

LRU – Least Recently Used

MAC – Mandatory Access Control

NGO – Non-governmental Organization

OSC2 - On-Scene Command and Control

PDA – Personal Digital Assistant

PGP – Pretty Good Privacy

PIC – Public Information Center

PKI – Public Key Infrastructure

PMI – Privilege Management Infrastructure

PVO – Private/Volunteer Organization

RAM – Risk Assessment Module

RBAC – Role-based Access Control

RI – Reputation Index

RIW – Reputation Index Window

RPG – Reference Point Group (mobility model)

RSM – Reputation Scaling Module

RW – Random Waypoint (mobility model)

SDSI - Simple Distributed Security Infrastructure

SPKI – Simple Public Key Infrastructure

TBAC – Trust-based Access Control

TMS – Trust Management System

TOC – Tactical Operations Center

TP – Trusted Peer

TS – Trust Store

WLAN – Wireless Local Area Network

WMA – Weighted Moving Average

1 Introduction

Groups of people collaborate to gain mutual benefit from exchanging information and sharing resources. In emergency incident response situations, organizations rapidly form dynamic coalitions that seek to leverage each other's expertise and resources to solve large scale, potentially lethal problems. The structure and composition of these dynamic coalitions is dynamic and tightly constrained by the limited amount of time available to configure the information systems that the coalition needs to be effective.

In such a dynamic collaborative environment (DCE), classical methods of access control are not effective in securing information while allowing users to fulfill their duties and responsibilities in the coalition. Rigid, role-based access controls could keep a DCE from achieving its mission by prohibiting or delaying authorized users access to the information and resources they require. Trust-based systems, which monitor a user's behavior as an indication of their future actions, provide the flexibility that a DCE requires.

This research shows how trust can be used to control access and describes a trust management framework for access control in a DCE. This chapter provides an overview of this research effort. Section 1.1 poses the problem statement, followed by a brief background of the problem and the motivation to seek a solution in Section 1.2. Section 1.3 highlights the solution's specific research contributions. The implementation and validation methodology used in this research are outlined in Section 1.4. Finally, Section 1.5 summarizes the results of the research and describes the organization of the remainder of this document.

1.1 Problem Statement

"A PMI is to authorization what a PKI is to authentication."[1]

This research developed a decentralized trust-based access system, which resided on each node of the network. The trust management system (TMS) provided a framework for aggregating behavior grading information and applying the resultant reputation value to control access. The TMS used behavior grades to create a reputation for peer nodes and to assess the general uncertainty in the network. The use of reputation and risk assessment tools to allocate and enforce access control provided flexibility in a DCE.

1.2 Background and Motivation

Disasters can happen at any time. When they occur, emergency service organizations from local, state, and federal governments respond as quickly as they can to reach the site of the incident. Once on site, the various organizations begin the arduous task of providing assistance to stricken civilians and repairing the damage. Only after the initial response can organizations begin to address the specific requirements of the emergency.

In an interview with the Roanoke Times, the Alleghany Region Emergency Services planner, Rick Moorer, stated that the greatest obstacle the responders faced was communications (Sturgeon 2004). Effective coordination between the organizations relied on timely information exchange. In effect, the people responding to emergencies formed DCEs.

The ad-hoc nature of the response, the broad range of interested organizations, and the presence of "interested parties" posed interesting problems in access control. The formation of DCEs gave rise to a specific scenario called the Dynamic Coalition Problem (DCP). In the DCP, each member required that it retained complete access control over the resources they provided to the coalition. The current emergency services doctrine proposed the use of a hierarchical framework called the Incident Command System (ICS). The ICS might have provided an excellent opportunity for Role-based Access Control (RBAC) but the structural fluidity in the participating organizations required configuration time and expertise that has not existed to date.

[1] (Chadwick and Otenko, et. al. 2003), p 62.

This scenario required an access control system that was decentralized and enabled discretionary control to the coalition's resources in a secure, non-structured, and non-reputable manner. Trust management provided the granularity and dynamism to the access control that ICS required by removing the requirement to pre-configure a hierarchy of roles and permissions.

1.3 Research Contributions

The over-riding contribution of this research was to design a trust-based access control system that combined reputation and risk (representing two distinct types of trust) and a memory to sustain them both to implement a dynamic access control mechanism. Creating a trust-based access control mechanism enabled coalitions to form quickly so that the group could cooperate to work toward shared goals through the fusion of credential-based identification and reputation-based access control system research.

Contributing to this research goal was the development of several key components of trust management. A unique method of reputation aggregation and computation called reputation scaling was developed. Complementary to this, methods for assessing global trust conditions for the establishment and maintenance of risk analysis were similarly implemented. Finally, system memory structures and management procedures were created to store and maintain the behavior information that was the fuel for the access control machinery.

This research answered the following questions in Section 5.2:

1. What were the benefits of trust-based access control?
    (a) In terms of efficiency.
    (b) In terms of accuracy.
2. What were the costs and vulnerabilities of trust-based access control?
    (a) In terms of performance impact.
    (b) In terms of workload for small, mobile nodes.
3. How effective was trust-based access control in reacting to different network risk conditions?
4. At what point did the trust management system fail to either:
    (a) Protect users and resources from malicious peers?
    (b) Isolate good users through overly protective controls?

By answering these questions, this research made the following contributions to the state of the art of access control and trust management:

1. That the concept of trust-based access control actually was beneficial and effective in DCEs.
2. That the design of the trust management system was flexible enough to be applied to a wider array of computing platforms and real-world situations.
3. That the concept of trust management could become the foundation for future implementations of access control.

1.4 Methodology Overview

The TMS was constructed and tested in C++. Each module of this design (see Figure 3-7) was verified individually before combining the components and validating the entire system. Computer models were constructed to test each module's ability to identify and react to a variety of situations correctly and quickly.

Once the modules had been verified, they were combined into a working system. These items resided on a mobile node and, when interacting with a server or other node in the network, were able to judge and be judged by reputation to allow access to shared resources. The system test gauged the TMS's ability to combine environmental indicators, varying mobility patterns, and behavior grades to determine access control.

The system's performance was evaluated against its ability to accurately grant or deny access to users in a coalition. Component modules of the trust management system were expected to detect and react to network conditions. Part of the reaction included adjusting security parameters and protecting from unauthorized access while continuing to allow access from authorized users. Simulation scenarios modeled a mobile ad-hoc network environment, including selfish and malicious node behavior.

1.5 Results and Conclusions

This research contributed a means to retain and process behavior history in a DCE so that decisions could be made that allowed nodes to interact safely. User-specific behavior information was combined with a general risk assessment to produce a judgment of the trustworthiness of a prospective collaborative partner. The TMS implemented an innovative combination of risk and reputation to create an access control decision based on past performance and current network conditions. The results of this research demonstrated that trust management provided accurate and effective access control in a DCE.

The investigations of reputation aggregation and calculation showed that trust-based systems could track the behavior of users, a criterion that was then applied to access control decisions. The unique 3Win method of assessing reputation weighted behavior grades dynamically to reflect changes in reporting nodes' trust levels before calculating the potential associate's reputation.

The TMS used an innovative risk assessment process, basically assessing the trust of the network as a whole, to adjust the trust thresholds that granted or removed trust. This process gauged uncertainty and adjusted trust thresholds to protect against the presence of risk. The dynamically tuned thresholds were more effective than systems constrained to fixed thresholds, which was the norm. The tuning process resulted in more accurate access control decisions. Reinforcing the increased accuracy was the fact that this improvement was implemented without increasing the cost associated with making those decisions.

Finally, the trust store was implemented to provide a memory of current and past trust associations. The trust store utilized a unique behavior record to store behavior grades and provide referrals to trusted peers. The development of the trust store required the creation of an adaptive technique for estimating network density and applying this estimate to size system memory. The trust store also enabled a process of credential exchange that promoted the use of evidentiary performance assessment rather than the more usual trading of unsubstantiated values.

The following chapter provides a brief overview of the dynamic coalition problem and forms of access control. A system architecture framework is presented in Chapter 3 that discusses the modules that comprise the trust management system. The verification and validation test results are provided in Chapter 4. Finally, Chapter 5 concludes this document and offers thoughts on future work that could be undertaken to expand the fields of trust management and access control.

2 Literature Review

This research focused on the application of behavior and trust to the problem of access control in a decentralized environment. This chapter provides both the background of dynamic coalitions and a review of research in the areas of trust management and access control. Section 2.1 provides an overview of the dynamic coalition problem (DCP) and early solutions that attempted to control access in Dynamic Collaborative Environments (DCE). After setting the stage for this research, Section 2.2 reviews access control theory and practice, including an explanation of Role-based Access Control (RBAC).

Once a background in pre-configured access controls has been established, Section 2.3 discusses the use of incentives to stimulate participation and positive behavior in a DCE. A basic theoretical background in trust follows in Section 2.4 and provides examples of trust management. Section 2.5 summarizes both access control and trust management and shows how they can be combined to meet the challenges of dynamic coalitions.

2.1 Dynamic Coalitions

Large problems have often required large commitments of resources. Often, the required resources were above and beyond the capabilities of a single organization. In these situations, autonomous organizations formed coalitions to share resources and achieve common goals (Li and Mitchell 2003). We call these goal-oriented collaborative efforts *dynamic collations* because organizations joined and departed the coalition based on individual internal goals and objectives. While a participating organization desired cooperation, it also required that it retained ultimate authority over the resources it controlled prior to joining the coalition. The two characteristics of decentralized resource control and independent membership were considered hallmarks of dynamic coalitions.

Research into this area began in the late 1980s as businesses formed temporary alliances to work in large, often government funded, projects (Estrin 1986). In these projects, access control was recognized as a key concern. Systems that implemented controls that were too rigid or required extensive administration delayed projects. On the other hand, companies were often unwilling to participate in partnerships when they could not be assured that their information would be protected from release to unauthorized parties. Under these circumstances, research focused on distributed file sharing and database applications. These applications required non-discretionary access controls to safeguard sensitive information. Like the military applications that followed, access controls were enforced at specific network locations, called gateways, and relied on the verification of a user's identity to pass information through the appropriate gateway.

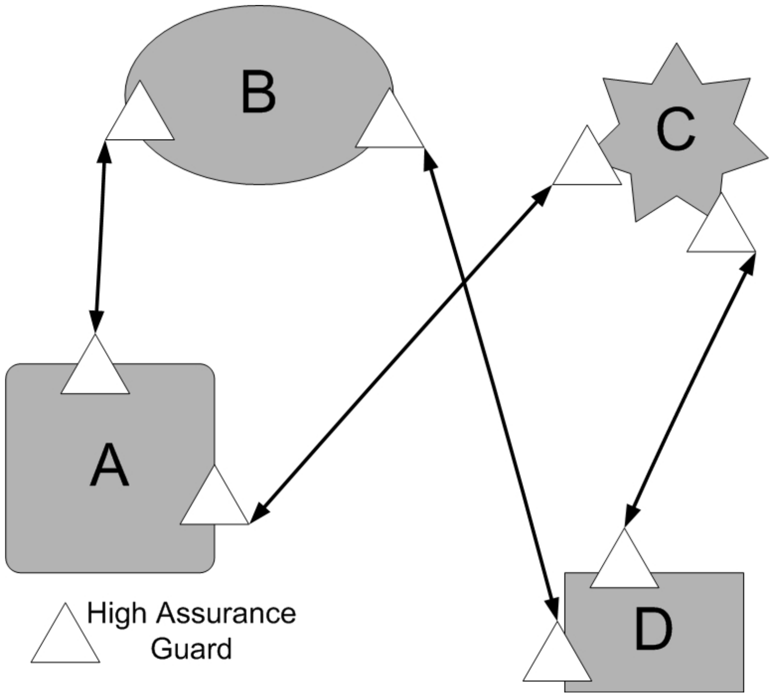


Figure 2-1 High Assurance Guards in a Coalition Environment

In the early 1990s, the military devoted vast amounts of research energy and resources to the problem of inter-organization (i.e., inter-service or multi-national) information exchange with mandatory access controls. Formatted messages allowed gateways to apply mandatory access controls (MACs) by evaluating user identity and data classification. Global, static networks were constructed by each nation, sometimes by individual services within the nations. To facilitate communication with allies and coalition partners, extremely expensive "High Assurance Guards" (HAGs) were placed at key locations within the network (Gibson 2001). The HAGs performed as secure routing devices, checking packets as they flowed between a pair of coalition partners. They were very laborious to configure and maintain, as each bilateral link required its own set of HAGs, one for each coalition partner, as shown in Figure 2-1.

Even as information digitization increased and the scope of coalitions broadened, the precepts of gateways and the reliance on individual user identity and consistent data labeling persisted. The assumptions that these concepts relied upon may have been valid in an era of formatted, centralized, low-volume communication but they have been shown to be invalid in today's age of ubiquitous email and web services (Binnendijk and Johnson 2004). The volume and informality of these communications media meant that HAGs were reduced to operating as super-expensive firewalls, as the users often had no verifiable identity and the information was not labeled. The experiences gained in the coalition operations of the late 1990s highlighted the fact that the foundational assumptions that enabled gateway-based MAC-only networks to work were no longer applicable. These foundational elements hindered the execution and scalability of coalitions and research has since focused on information exchange in a more dynamic, less homogeneous environment. This type of environment is called a Dynamic Coalition and the associated challenges with information exchange are grouped under the DCP.

The DCP was defined as the security, resource, and information sharing risks that were inherent in coalitions that formed quickly yet needed to share resources and information to resolve a crisis (Spring, Gormley et al. 2000). Although the authors specifically addressed military applications of DCP, it was noted that many of their concerns were also found in the civil coalitions that formed for crisis or emergency response. The concern of organizations intending to join a coalition was, in the words of the Chairman of the Joint Chiefs of Staff, that an information sharing capability did not "imply access to information without constraints."[2] He stressed that sensitive information, although not expressly classified, must be protected.

Most, if not all, of the previous research in the DCP focused on the challenges surrounding military applications. In these settings, rigidly hierarchical organizations shared information but not physical resources. Mandatory and role-based access controls were natural fits in this environment (Phillips, Ting et al. 2002). Information received classification labels and users received security labels. The network architecture focused on validating the labels that a role might access. Security was enforced through rigid application of identity and role checks using specialized, pre-configured hardware (Gibson 2001), as discussed previously.

[2] Shelton, Joint Vision 2020, page 16.

In some cases, joint administration of resources was assumed. This led to discussion of the application of threshold cryptography to create joint certificate and attribute authorities. The threshold cryptography technique split a secret among several members of a coalition and requires that a certain number *k* members of the coalition's population of *n* organizations, commonly signified by *k of n*, agree to issue a credential (Byrd, Gong et al. 2001). This technique was also applied to the publication of authorization policies (Khurana, Gligor et al. 2002). In both cases, previous knowledge of the required number of secret shareholders and membership in the coalitions was predetermined (i.e., configured off line). As discussed previously in this document, these assumptions were not applicable in a more dynamic environment.

Neither the military nor the threshold approaches were acceptable for use in a civil coalition that lacks hierarchy, pre-defined members, and the ability to pre-configure resources. Military (AFSC 1997; USCG 1999) and government (FEMA 2004) planning doctrine attempted to address some of the challenges of civil coalitions by layering a military planning process and role hierarchy over the civil coalition environment. Built largely around military staff procedures and communications systems, these documents formed a foundation for interaction during a disaster incident.

The Federal Government attempted to address the issue of interagency communications through the development of the Incident Command System (ICS) (FEMA 2004). ICS was a standardized on-scene incident management concept designed specifically to allow responders to adopt an integrated organizational structure equal to the complexity and demands of any single incident or multiple incidents without being hindered by jurisdictional boundaries. The challenge was in building the tools and systems that would have enabled ICS to be put into the hands of the responding organizations on the scene.

In 1999, the United States Coast Guard, in a partnership with the United States Army Corps of Engineers, began a project to implement a system called the On-Scene Command and Control (OSC2) system. While the prototype of this system was trialed in mid-1998 (USCG 1999), future capabilities of the OSC2 were to include wireless local area networks (WLANs,) indicating that the Coast Guard and other emergency service agencies were planning to deploy wireless networks during emergency response situations. These WLANs were expected to provide connectivity between organizations across the broad range of information requirements set out by the ICS.

While the ICS concept provided a framework of positions and responsibilities (FEMA 1994a), the exact composition of an incident response team was very ad-hoc. Volunteer organizations and individuals who contributed to the disaster relief operation assisted professional emergency service organizations. These volunteers were incorporated into the ICS framework in a multitude of capacities and positions. Their presence presented the requirement for multiple levels of access and security that needed to be combined. Organizations maintained volumes of sensitive information that they required to execute their role in the coalition but that was not releasable to outside agencies due to legal constraints.

Additionally, during an ICS-managed coalition operation, the Incident Response Team (IRT) set up a Public Information Center (PIC) under the responsibility of an Information Officer (IO)[3]. The IO was responsible for developing and releasing information about the incident to the news media, to incident personnel, and to other appropriate agencies and organizations. As such, the PIC provided the connectivity for media organizations, concerned citizens and family members, as well as other interested organizations. In many multi-jurisdiction incidents, an agency or jurisdiction might send a representative who was not on direct tactical assignment but who was there to assist in coordination efforts. The demand for information from journalists and concerned citizens represented the requirement to provide information to external organizations while at the same time protecting operational data before it was cleared for release. In crisis situations, enforcing rules of information exchange was critical to reduce confusion and focus relief efforts. Following the September 11, 2001 terrorist incidents, CNN.com reported that "…duplicate reports and confusion in the hours and days immediately following the attack" inflated the estimated death toll from 3,040 to over 6,000.

While inter-organizational information exchange was never easy, the dynamic membership, informal communications, and short timeframes associated with the sort of emergency response coalitions that desired to use ICS clearly ruled out the use of static, label-based access control. Not only was there the inability to gain consensus over any security labeling policy, but also the lack of defined membership made seeking this consensus an exercise in frustration. The presence of the ICS structure did offer an alternative, however, and the next

[3] FEMA 1994, ICS Position Descriptions and Responsibilities, pp. 2 – 3.

section discusses the theory and possible application of access control based on organizational position, duties, and responsibilities.

2.2 Access Control

Two schools of thought have long dominated the topic of access controls. Both emerged from military-oriented, mainframe-based applications. MAC was developed as part of the implementation of multi-level secure messaging systems in the 1970s (Feiertag, Levitt et al. 1977). MAC restricted access based on security labels assigned to users and objects (e.g., clearance and compartment caveat). The mandatory aspect of MAC was that the access controls could not be bypassed. In other words, users accessed objects through the MAC system and were not able to change or reassign permissions.

Discretionary Access Control (DAC) allowed an object's owner to determine the access rules for that object. Several types of DAC existed and were implemented giving a wide range of ownership transferal and delegation properties (Osborn, Sandhu et al. 2000). While yielding more flexibility than MAC systems, DAC lost the ability to provide provable security to resources. Both MAC and DAC have been implemented using role-based access control as a means to assign permissions to users.

2.2.1 Role-based Access Control

Assigning access control permissions for each resource directly to specific individuals did not scale well. This method was used in centralized, mainframe environments where the user population was closed and hierarchically related to each other, and therefore their set of attributes. With more open communication environments like the Internet, the difficulties in issuing, revalidating, and revoking attribute certificates became unworkable. To combat this problem, Sandhu (Sandhu, Coyne et al. 1996) introduced the concept of RBAC.

In RBAC, role administrators assigned users roles that described job functions or specific responsibilities required to perform their duties. The roles defined the accessible resources and the actions that could be invoked on these resources. A single individual might be assigned several roles and these roles might change. While the role might change, the identity of the individual remained the same. At the same time, the attributes that were defined for the role were also stable. The result was that the dynamism in an RBAC system was found in the mapping between users and roles (Ferraiolo, Kuhn et al. 2003).

RBAC relied on the critical relationship between identity assignment and role assignment. RBAC in any of its forms required a one to one correspondence between user identifiers and human beings. If a human user was allowed to function in the system with more than one identifier, constraints and role hierarchies could be circumvented.

The economies of RBAC were found by expressing the association of permissions to individuals as an ordered set *(U, P)*, where *U* was the number of individuals in a job position and *P* was the set of permissions for that particular position. Matching individuals to permissions required $|U| * |P|$ associations. If, however, *U* represented the set of users in a role that described the job functions and *P* the set of permissions associated with a role, the number of associations became $|U| + |P|$. This savings was the result of mapping multiple permissions to a single role that was subsequently mapped to the individual users. Thus, RBAC was more economical as long as $|U| + |P| < |U| * |P|$, which was satisfied when $|U|, |P| > 2$. Mathematically, in an organization with n job positions ($\forall n > 2$), Equation 1 expressed a rough approximation of savings as:

$$\sum_i^n \left(|U_i| + |P_i|\right) < \sum_i^n \left(|U_i| * |P_i|\right) \tag{1}$$

This equation provided only a rough approximation since users commonly fulfilled more than one role in the organization and those roles might be hierarchically related. Because RBAC was applied to many different types of systems to satisfy a variety of access control requirements, Sandhu developed a RBAC hierarchy that specified the relationship between four conceptual models, depicted in Figure 2-2.

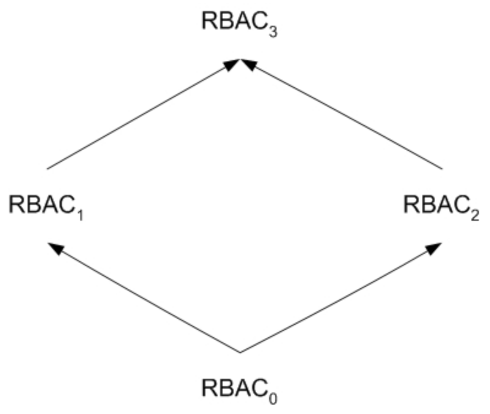


Figure 2-2 Role-based Access Control Model Hierarchy

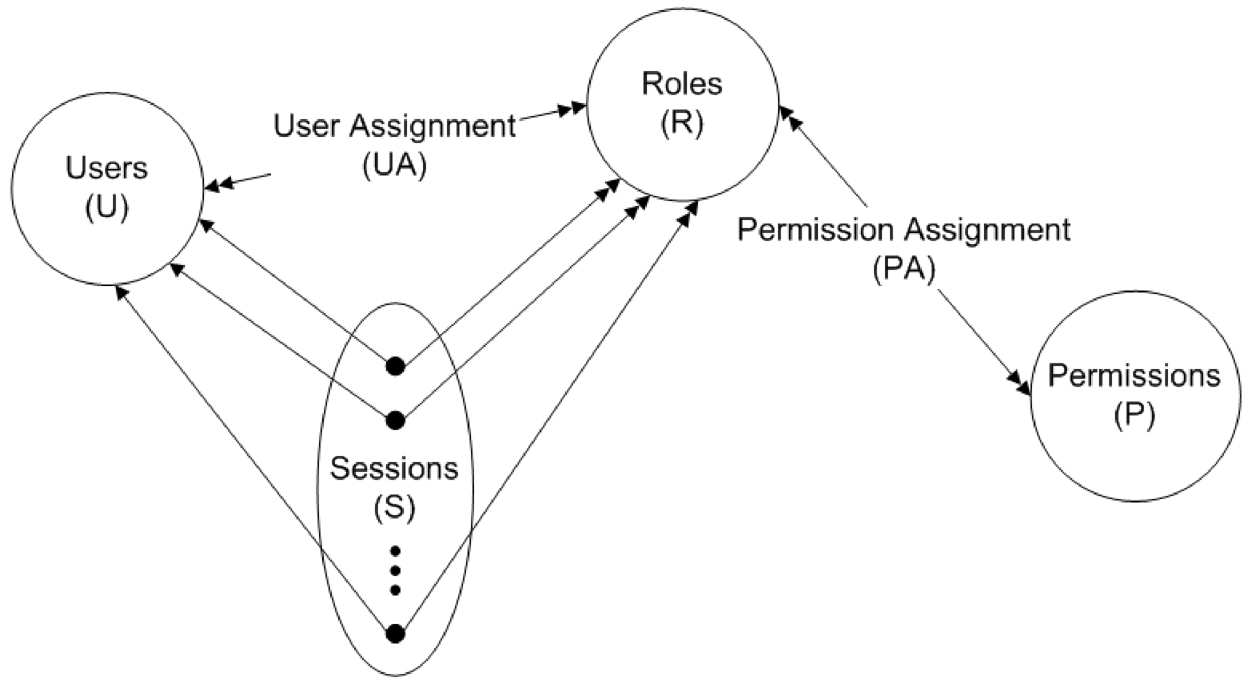


Figure 2-3 Basic Role-based Access Control Model

The core model, $RBAC_0$, was comprised of four basic elements, as shown in Figure 2-3. Users (*U*) were humans employed by an organization. Roles (*R*) were specific job functions within the organization. These functions described the authority and responsibility placed on a person in that role. Permissions (*P*) were composed of objects and the operations that can be applied to those objects. Permissions were, by convention, the positive authorization of a specific action on an object. In other words, permissions granted access. Negative permissions were considered under the discussion of constraints below (Sandhu, Coyne et al. 1996). The last element, Session (*S*), was established when a single user wanted to invoke a subset of roles that he was entitled to. This association between user and roles remained constant for the duration of the session, even though a user might have multiple simultaneous sessions.

$RBAC_0$ was defined mathematically as follows:

- U, R, P, S were users, roles, permissions, and sessions, respectively;
- $PA \subseteq P \times R$ was a many to many permission to role assignment relationship;
- $UA \subseteq U \times R$ was a many to many user to role assignment relationship;
- user: $S \rightarrow U$ was a function mapping session s_i to the single user user(s_i);
- Roles: $S \rightarrow 2^R$ was a function mapping a session s_i to a set of roles roles(s_i) $\subseteq$ (r | (users(s_i),r) $\in$ UA) (which can change with time) and session s_i had the permissions $\cup_{r \in roles(si)}(p|(p,r) \in PA)$

$RBAC_1$ inherited the basic functions of $RBAC_0$ and introduced the concept of a partially ordered set of roles called a role hierarchy. The role hierarchy was a means to structure roles, and, therefore, permissions to reflect the organization's internal structure. By convention, the most powerful roles were shown at the top of role hierarchies while the less powerful roles were placed toward the bottom. In addition to illustrating relative power, this ordering also demonstrated role inheritance. Roles could inherit permissions from lower level subordinates. This inheritance property was transitive, such that permissions flowed up the hierarchy from its lowest levels. Inheritance was also reflexive because a role inherited its own permissions. Thus when a user invoked a session, he selected a role r_i from the set consisting of his own roles (r_x) and any role subordinate to that role ($i \mid x \geq i$). The session was established with $S(r_i)$ and could utilize the permissions that were assigned directly to r_i and those roles that are subordinate to r_i.

$RBAC_1$ was defined mathematically as follows:

- $RBAC_1$ inherited the elements described in $RBAC_0$, above;
- $RH \subseteq R \geq R$ was a partial order on R, called a role hierarchy;
- Roles: $S \rightarrow 2^R$ was modified from $RBAC_0$ to require $roles(s_i) \subseteq (r \mid (\exists r' \geq r)[(users(s_i),r') \in UA])$ (which could change with time) and session s_i had the permissions $\cup_{r \in roles(si)}(p|(\exists r'' \leq r)[(p,r'') \in PA])$

The $RBAC_2$ model introduced constraints, which limited aspects of $RBAC_0$. Constraints enforced the concept of separation of duties, enabling the definition of mutually exclusive roles so that no single user gained too much power or could commit malicious acts unchecked. More specifically, constraints were placed on *PA* to limit the assignment of powerful permissions. Other commonly applied constraints included cardinality and prerequisites.

Cardinality was usually applied to establish the maximum number of users that could be assigned a specific role. Cardinality was also used to limit the number of roles a single user was assigned. Because roles were mapped to users in a many to one relationship, it was difficult to enforce minimum cardinality constraints and ensure that the system did not assign a role to more users than allowed.

Prerequisite roles were established to ensure competency and appropriateness. For example, Alice could be assigned role R_2 if she is already assigned role R_1. The prerequisite role was almost always subordinate to the role with the prerequisite constraint. This constraint applied to permission assignment as well. Thus, for example, Alice could only read a file if she had access to the directory where the file was stored.

The consolidated model, $RBAC_3$, combined both role hierarchies and constraints. Because of the transitivity of inheritance, the elements of the core model were present as well. Combining $RBAC_1$ and $RBAC_2$ allowed administrators to set constraints on role hierarchies. Constraints often affected hierarchies in unforeseen ways, as roles at the top of the hierarchy might violate separation of duty or cardinality constraints because of the inheritance property.

By applying the $RBAC_3$ model, a system could be configured to satisfy the requirements of both DAC and MAC (Osborn, Sandhu et al. 2000). The keys to realizing the requirements of DAC and MAC were in the configuration of the role hierarchies and application of the administrative roles. In MAC, the crucial principle of "read-down, write-up" was enforced by mapping the permission to read and write to objects (e.g., (o,r) and (o,w), respectively) to matching pair of roles, xR and xW. This assignment of permissions to specific roles set the sensitivity label of object o to x.

For DAC without ownership transferal, the arrangement of administrative roles relied on the implementation of role hierarchies. An owner of object o was assigned the role *Owner_o* and could authorize a set of eight permissions for each action, such as read or write, to a regular role (e.g., *Read_o*). Implementing ownership delegation or transferal made the administrative model more complex and each object required the creation of three administrative roles in addition to a regular role in a process described in detail by Osborn (Osborn, Sandhu et al. 2000).

Thus, in systems requiring either MAC or DAC, $RBAC_3$ simulated and enforced the access control requirements using role hierarchies and systematic mappings of permissions to roles. Constraints played a vital role in the assignment of roles, as the administrative roles became complex to the point that implementing RBAC reached the point of diminishing returns.

RBAC was implemented by embedding roles in Attribute Certificates (Acerts), just as other systems had embedded privileges in similar credentials (see Section 2.2.2). Like other privilege management infrastructure systems (PMI), RBAC systems faced the challenges of identity validation and revocation. They were also challenged by dynamic role assignment (i.e., how do we assign/reassign roles to users without offline configuration or a centralized authorization authority?). As discussed in the following section, these obstacles and the lack of a centralized access control management structure opened RBAC systems to attacks including privilege escalation, role impersonation, or malicious disclosure (Young and Aitel 2004).

2.2.2 Credential-based Access Control

Secure and verifiable identities were the key to creating and using trust to make decisions. Researchers attempted to satisfy this requirement by credentials for use as proof of identification. Credentials were digitally signed documents or certificates that asserted a binding between a cryptographic public key and some property.

Theoretically, this public key was linked to an identity or a role in an organization. The issuer of the key had the responsibility of ensuring that the appropriate key was embedded in each certificate. Assuming that everyone else in the network trusted the issuer, they could use this key to verify a user's identity, role, or organization.

Because the credentials provided a non-reputable and trusted link between the identity of a user and their permissions, they were often utilized for access control (Biskup and Wortmann 2004). Users confirmed a certificate's authenticity by verifying the issuer's digital signature. Given an authentic certificate (and therefore an authentic identity), a user then validated the asserted permissions by comparing the certificate against a Certificate Revocation List (CRL). If the certificate had not been revoked, the assumption was that the user had the correct identity and permission to perform the action it had requested.

RFC 3281 (Farrell 2002) established the basic set of capabilities and operational rules for certificate-based access control. The credentials, also called certificates, enumerated the actions and objects that the subject was permitted to access. Access or privilege certificates were issued to users (or more accurately, electronic identities.) A user presented his access certificate, along with the desired action, to the resource when he wished to invoke an action. Assignment and promulgation of access policy statements told the user what he was allowed to do. On the other hand, each resource maintained access control lists (ACL) that described what users were allowed to do. The user was permitted to complete their request only when they presented a policy statement that was allowed by the ACL. The maintenance of ACLs was crucial to the success of a credential-based system and was the subject of much discussion and investigation.

Where RFC 3281 proposed the general framework for access control using credentials, several schemes implemented this concept using Simple Public Key Infrastructure (SPKI) and the Simple Distributed Security Infrastructure (SDSI) (Ellison 1999a; Ellison 1999b). The SDSI used local names to establish the identities of nodes or users in a network. SPKI used these names and provided the ability to bind identities, authorizations, and delegation permissions with public keys. The SPKI/SDSI concept used name certificates to establish identities and CRLs to maintain and verify the validity of these identities. Authorization certificates were issued separately to bind the authorizations and delegations to the user's identity through the use of a public key. The link between the identity and the authorization was, therefore, cryptographically protected through the public key. In use, authorization certificates were verified in the same manner as the identity certificates, requiring comparison of the certificate and its key with Certificate Authorities (CA), CRLs, and ACLs.

The first system to attempt to establish trust between subjects and resources was Blaze's PolicyMaker system (Blaze, Feigenbaum et al. 1996). It was the first tool that introduced the use of trust into an access control decision. A subject gained access to resources simply by possessing credentials that authorized specific actions on the resource. Blaze later extended this work with the KeyNote tool (Blaze, Ioannidis et al. 2002) to make more use of his credential checking trust management engine.

In KeyNote, a user requested a resource by presenting a triplet of values: the security policy it wished to invoke, a list of credentials that supported the user's use of the specific policy, and the request. The trust management engine examined the credential and provided the application with a GO/NO GO response. As with this research, the PolicyMaker system did not make the access control decision but provided advice to the application that was charged with enforcing or allowing the decision.

While groundbreaking, there were two deficiencies to PolicyMaker and KeyNote. First, the credentials had to be issued as part of the role assignment or through pre-configuration. This was suitable in a static, centralized network and assumed that users will know their roles when they entered the system. Second, there was no protection for the credentials or the security policy. Both PolicyMaker and KeyNote addressed the access control problem without worrying about dangers of disclosing sensitive information. Because of this deficiency, a malicious user could obtain the policy, create fraudulent credentials, and then fool the trust engine into authorizing access.

Yu and Winslett's work in the TrustBuilder project (Yu, Winslett et al. 2003) utilized a system of credential exchanges and directed policy graphs to satisfy access requirements. In their system, nodes entered into an iterative process of trust negotiations. The requesting node provided credentials to satisfy the resource node's policy statements. If the resource was satisfied of the credentials' validity, it allowed access.

In their work, the authors pointed out several challenges to the implementation of policy graphs. First, there was the problem that both requester and resource had to disclose sensitive information without the assurance that their partner would meet the access criteria. Realistic access control policies contained sensitive information that disclosed the nature of the credentials, the user, and the resource - none of which may be desirable. Second, ensuring that the policy graphs could be satisfied was non-trivial. In other words, the system had to guarantee that there existed a path from the initial request so that a user could reasonably expect that he could access the desired resource if he presented the required credentials. There is the possibility that a long policy graph might be used to disguise a denial of service attack. This possibility forced the authors to define safe disclosure sequences, which limited any sequence of credential disclosures that led to the disclosure of the resource.

The X.509 standard (Housley, Ford et al. 1999) was promoted as a simple, scalable, and interoperable means to implement attribute certificates. Nykänen (Nykänen 2000) explored this line of research as a means to eliminate the need for ACLs for each resource. He compared the feasibility and performance of an SPKI system against an X.509 implementation and concluded that both were usable for the purpose of authorization. SPKI had the benefit that it did not require a global naming scheme, such as the one that X.509 relied on. On the other hand, X.509 offered a tighter bond between the identity and the authorization attributes. Using either SPKI or X.509 for authorization, he confessed, was a matter of "moving difficult things from one domain to another."

Shin and Ahn (Shin, Ahn et al. 2002) followed the same concepts as Nykänen. Although they admitted that using an X.509 certificate could raise difficulties such as shortening the certificate validity period, increasing the number of CRLs, and making certificate management more complex, they recommended that a separate Authorization Authority (AA) manage the attribute certificates. The AA bound the node identity to its permissions or other attributes. The expectation was that a node's identity remained stable for longer periods of time than its role or set of permissions. Having an AA freed the CA from having to issue, reissue, and revoke attribute certificates while still taking advantage of the standard format. The disadvantage to this scheme was that the AA needed a complete set of management functions and procedures to assign attributes. The AA's functions and procedures were separate from and in addition to those that the CA required. The net benefit, therefore, was questionable.

Several projects explored implementations of certificate or credential-based PMI. The University of Salford built the PERMIS system to implement access control using X.509 certificates (Chadwick and Otenko 2002). This research utilized Access Certificates (Acerts) in addition to Public Key Certificates (PKCerts) to build a role-based access control system. The Akenti project at the Lawrence Berkeley National Laboratory supported similar objectives and infrastructure design (Chadwick and Otenko 2003). Its development focused on controlling access to widely distributed resources in environments where resources had multiple owners, each desiring to impose conditions for access. Both PERMIS and Akenti utilized Lightweight Directory Access Protocol (LDAP) directories to store access certificates and policies, the exact implementation of these components being the primary difference between the two schemes.

PMI in static networks relied on access control policy statements and ACLs to enforce secure access to network resources. As had been found with Public Key Infrastructure (PKI) CRLs, management of these statements and lists required a robust, centralized infrastructure that was not present or even possible in a mobile wireless networking environment. Schemes that utilized the X.509 suite of standards were effective in a centralized, wired environment but are less effective in decentralized, wireless environments due to the lack of a single controlling organization and the possibility of network partitioning.

2.2.3 Role-based Access Control in Dynamic Networks

When SDSI (Ellison 1999b) established decentralized access control through the use of certificates, it relied on web of trust and delegation chain constructs that did not perform well in partition-prone networking environments. Two recent projects attempted to address this shortfall: RT (Li, Mitchell et al. 2002; Li and Mitchell 2003) and dRBAC (Freudenthal, Pesin et al. 2002). Both combined the advantages of RBAC with trust management to implement decentralized and scalable solutions.

RT was based on a set of logical rules that express attributes. Nodes presented attribute credentials to gain service from other nodes. These credentials were translated into rules, which were then verified using a logic engine. Once a rule had been checked, the engine traced a credential chain for proof of validity. RT implemented standard $RBAC_3$ functions (consolidated model) and included the separation of duty constraint. RT provided a constraint to limit the length of credential chains, but this appeared to be a "quick-fix" for what would otherwise be a glaring vulnerability to the system.

Another of RT's weaknesses was found in its logic engine, Datalog. Due to Datalog's structure, RT's designers were forced to construct another "work around" to allow them to implement third party role delegation. In third party delegation, Alice could delegate permission to assign a role *Alice.z* to Bob and Bob could delegate the same permission to Carl. In RT, however, Alice had to allow Bob to create another role called *Bob_Alice.z* that inherited *Alice.z*. Because Bob was the owner of *Bob_Alice.z*, he could delegate this to Carl. While functional, this solution presented administrative overhead (e.g., Alice has to create a "dummy" role) and cluttered Alice's role hierarchy with duplicate roles.

The dRBAC project (Freudenthal, Pesin et al. 2002) extended SDSI, RT, and KeyNote (Blaze, Feigenbaum et al. 1999) to provide an RBAC functionality in a distributed system and introduced support for multi-level access and system monitoring. The project linked roles to an issuing organization's public key, eliminating the need for additional policy roots. Cross organization collaboration was enabled by requiring the nodes to present support proofs with credentials. These proofs were certificates from trusted third party entities that attested to the user's identity, role, or access requirements. Credential attributes were allowed to have values, allowing multiple levels of access to a resource depending on the source of the credentials.

Credentials were managed locally by having each node equipped with a "wallet" of role certificates and proofs. A node presented its certificate and proofs when making a request. The peer node attempted to verify the certificate before granting the request. If the peer node could not verify the certificate or required additional proofs, it queried the CA and retrieved the additional proofs. As with other systems, the ability to make global queries to resolve authorization requests implied reliable communications media and nodes with the capacity to store large amounts of data.

2.2.4 Access Control Challenges in Dynamic Environments

Previous work in inter-organizational collaboration was based on Estrin's assumption that all information bore recognizable security classification labels and all the users were identifiable through a central or trusted authority (Estrin 1986). The combination of these two characteristics enabled the employment of the Bell-LaPadula Model of MAC (Ferraiolo, Kuhn et al. 2003), a perfectly valid assumption in a scenario where the bulk of the information was kept internal to the organization and a much smaller subset was "published" to the coalition. Further applications enabled MAC by establishing information gateways that implemented the label and identity checking (Gibson 2001).

These rigid requirements did not meet the wide variety of information needs of coalitions that were dynamic in both membership and duration. Modern coalitions used email and web to exchange information, as opposed to the centralized messaging systems of the Cold War. The ubiquity and informality of email and web-enabled applications shifted the balance of information from "internal to the organization" to "shared in common" with the coalition.

DCEs presented interesting design challenges for access control systems. Unlike static, wired environments, the expectation was that users could enter and leave the network environment at will. The design challenges were characterized into three broad categories: membership, mobility, and behavior. The challenge of membership arose, as alluded to earlier, when members joined and left the network based on their own goals and objectives. While they could physically enter and leave the network coverage zone, the user could also drop in and out of the network for reasons of energy conservation; turning his device on when he had to transmit or receive data and then "going to sleep" or turning the device off to conserve battery power (Michiardi and Molva 2002b; Molva and Michiardi 2003).

Compounding the membership problem was the challenge of identifying the members in a verifiable manner. Physical characteristics and short range, interpersonal certification were discussed (Hubaux, Buttyan et al. 2001) but these methods presented too many limitations and did not scale to large or sparse networks. The challenge in identifying users was to prevent a single user from dominating or co-opting the network by maliciously establishing multiple identities (Douceur 2002). Users were not expected to be vetted by a single trusted source and different schemes to allow network users to identify themselves to other members were developed (Ellison 1996; Seigneur, Farrell et al. 2003; Montenegro and Castelluccia 2004).

In addition to the identification obstacle, users were expected to be mobile. While some researchers tried to show that mobility aided security by building collaborative acquaintances (Capkun, Hubaux et al. 2003), others recognized that malicious users might use their mobility to try and hide their undesirable behavior (Michiardi and Molva 2002b). Rather than assume that all nodes were good or bad, systems in mobile ad-hoc networks needed to determine whether a user's behavior was intentional or a symptom of poor transmission capabilities. Nodes might cooperate when they needed something and then ignore the other nodes once their needs had been satisfied. This was called “freeloading” or “selfishness” in the literature (Aberer and Despotovic 2001; Moreton and Twigg 2003a). Detecting and discouraging this resource-draining behavior presented a non-trivial problem to dynamic networks.

These same authors developed incentive systems to stimulate participation in dynamic environments. Their rationale was to reward good behavior and participation with an incentive. Systems were developed in which peers would want (or need) to possess the incentive to gain access to resources.

2.3 Incentives

The concept of incentives came from examples in the real world. People were used to paying or offering a consideration for services. In a network where peers volunteered their services to help a common goal, incentives were considered to stimulate participation and prevent overloading specific resources (Buttyan and Hubaux 2000). By manipulating the price of a resource, peers were encouraged to spread their utilization around the network rather than overloading a single node. Incentive systems also discouraged nonbeneficial behavior, since nonparticipating peers would not have enough credit to impede legitimate users and would be effectively isolated. Moreton and Twigg (Moreton and Twigg 2003b) classified incentive systems into two types: token or trust systems.

Token systems introduced a form of notional currency, which nodes exchanged in return for services. A currency system was called a token system if the number of tokens was bounded. The existence of this bound allowed a node to calculate the value of the tokens that it possessed and negotiate for the services it needed. In unbounded systems, specific redemption schemes were implemented to defeat the possibility that a node could flood the network with its own currency. Moreton and Twigg (Moreton and Twigg 2003b) also introduced the notion that one node’s services might be deemed more valuable than another node’s. This gave rise to the subject of token exchange rates and a number of token trading schemes.

An example of an unbounded notional currency came from the Terminodes project, where the currency was called *nuglets* (Hubaux, Buttyan et al. 2001). Nodes paid to have their packet traffic forwarded by affixing a certain number of nuglets to their traffic. Intermediate nodes accepted a nuglet for passing the traffic on to the next hop. Unfortunately, this system allowed nodes to “mint” their own currency and the designers did not implement any of the redemption schemes mentioned in Moreton and Twigg, leading to inflation and the possibility of widespread currency flooding and fraud.

In addition to currency flooding and the challenge of attack-resistant redemption, there were many other problems with the use of notional currency. First, all of the researchers admitted that the implementation of a "counterfeit-proof" currency was extremely hard. Second, it was very difficult for a node to know how many tokens it needed to affix to the message to ensure delivery (i.e., "How much postage?"). This challenge was amplified in systems with exchange rates, as nodes had to determine the "postage," then calculate the exchange rate, and then negotiate for service. Third, currency redemption was difficult to police. Intermediate nodes might take payment and then not render services or might overcharge and leave the message with too few tokens to arrive at its intended destination. Finally, the authors were vague on the mechanism nodes used to set their prices. Nodes that offered services for low prices might become swamped while higher-priced nodes sat idle. Worse, nodes that could not afford their price were starved out.

The use of incentives also brought indirect complications. As mentioned in the discussion of notional currencies, the motivation to accumulate tokens could lead to an imbalance in the network. Nodes were encouraged to collect tokens but, if they then moved away or became disconnected, the network "economy" suffered from the lack of tokens. Worse, other factors like connectivity or communications capacity, might lead to an inflation scenario where certain nodes adjusted their prices in response to supply and demand. While the economic theories of Adam Smith were the foundation of capitalism, such pricing schemes would quickly ruin an ad-hoc network that relied on cooperation to operate.

More recently Buyya (Buyya, Stockinger et al. 2001) addressed pricing in a Grid computing environment. Member nodes fulfilled the role of consumer, producer, or resource broker. The resource broker represented consumers to producers in various transactions. Each producer interacted with a specific broker to manage his or her resources and schedule computational efforts within the Grid. Brokers operated through a market directory to advertise services. Resource pricing was negotiated because consumers "rented" the resources for a specific duration. In this system, the consumer nodes desired the greatest resource at the lowest price while the producers wanted the highest price for the least resource.

Buyya (Buyya, Stockinger et al. 2001) suggested a number economic models for price setting. The commodity market model employed either flat rate or supply and demand driven pricing schemes. Flat rate pricing models listed a set unit cost but could suggest "special" pricing offers that were only available during off peak usage. Yu and Singh (Yu and Singh 2003) described a payment scenario as follows: Alice (the consumer) listened for peers who offered a resource she wanted (e.g., a printer.) Bob, who was willing to share his printer, advertised the price of the printer's services (assuming a flat rate pricing scheme.) Alice took this price and matched it to what was called her "reserve price," which was what Alice was willing to pay for this specific service. Alice requested to use the resource of the first user she found whose price was under her reserve price. Alice tried to optimize her choice so that she used resources with lower prices or higher reputations but the payoff in credits saved versus the increased time, communication, and computation was doubtful.

A flat rate system could also ensure access to resources through a simple mechanism like the Paris Metro Pricing scheme (Odlyzko 1999). Basically, if a user decided if he or she could live with the "best effort" service, he paid only *x*. If he or she wanted better service or some sort of guarantee with the service, he paid *2x*. As the author pointed out, this was a simple and self-regulating system without the communication required in a bidding system. The problem was that the system provided no guarantee to the consumer that the producer could actually deliver what was promised.

Variable rate systems also existed. These systems relied on negotiation and concentrated more on the currency exchange than on the way prices were set. Levien (Levien 2004) described a "Stamp Trading" network. In this preliminary design, each node contained a set of on hand stamps, which were directly traded between users to acquire needed services. Consideration was given to the exchange rate of stamps redeemed for various available services within the system. In this economy, nodes had background processes that managed stamp distribution relative to available service buckets. The process attempted to fill empty buckets by redeeming stamps from full buckets with the aim of maintaining a general equilibrium amongst all the buckets. Interestingly, the aforementioned exchange rate was not based on economic factors of supply and demand but was based instead on a confidence factor (Levien 2004). This use of a confidence factor was based on credibility ratings of nearby nodes. As a node moved away, their rating decreased. As long as a trusted predecessor node stayed close and behaved satisfactorily, the confidence factor remained high for their respective stamps' valuation. Each stamp trade invoked a new calculation of stamp exchange rate value by examining the stamp issuer's credibility factor to produce a new stamp confidence factor. In this case, stamps were valued based on the age of the stamp and the credibility of the issuer.

Pricing and exchange rates were only two challenges faced by incentive systems. Credit redemption mechanisms had to address the possibility of cheating by either party in a transaction. Rivest (Rivest and Shamir 1996) invented a micro-payment scheme for packet forwarding. Similar to a threshold cryptography system, a producer needed to collect all of the parts of the payment to be able to use the credits. Buttyan (Buttyan 2000) extended this solution to address the issue of either partner defaulting before the transaction had completed. His system of "ripped payments" amounted to the consumer placing a deposit with the producer or a trusted third party and then paying the balance upon completion of the transaction. While this did not eliminate the possibility that the consumer would renege on the final payment, it did ensure that the consumer had paid something. Because the producer and the consumer might be more than one hop away from each other, Ben Salem (Ben Salem, Hubaux et al. 2004) suggested a situation where an entity could argue that they had assisted in providing a service, perhaps in a collusion with a confederate, and then can demand payment for services.

The other type of incentive system (Moreton and Twigg 2003b) was a trust-based system. Trust systems used node reputations as incentives. Nodes were motivated to provide services for other network members so that they could earn or maintain a good reputation. The desire to implement behavior grading in mobile ad-hoc networks led to the incorporation of sociological principles of trust into access control systems.

This requirement for a system to try to interpret a user's intent led to investigations into behavior, specifically to the application of behavior history as an indication of future actions (Datta, Hauswirth et al. 2003). The researchers' theory was that, given enough evidence of previous performance, a network member had a reasonable ability to determine which users were trustworthy. This determination was then used to guide the member when selecting partners with whom to interact (Hadjichristofi, Adams et al. 2005b). To fully understand why and how these principles could be applied, it was necessary to begin with a definition of the terms and constructs that deal with trust.

2.4 Trust

Trust, and more importantly decisions on trustworthiness, is omnipresent in life (Marsh 1994). Luhmann's sociological approach (Luhmann 1979) considered trust as "a means for reducing the complexity in society." This complexity was created as individuals interacted using their own perceptions, motivations, and goals. Solomon and Flores (Solomon and Flores 2001) contended that "trust forms the foundation, or the dynamic precondition, for any free enterprise society." They pointed out that what constituted freedom was the right to make promises and, more importantly, the responsibility for fulfilling them. Trust, therefore, was the basic underpinning of a cooperative environment. Trust was not an inherited trait but was learned as a member of the environment interacted with others. Another applicable definition of trust was provided by Gambetta (Gambetta 1988):

> "…trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects [our] own action."

Humans usually based the decision to trust on historical evidence that led them to predict another person or entities' future behavior (Aberer and Despotovic 2001). When this prediction was shown to be incorrect, the other person was trusted less, if at all. Rather than accept a philosophical betrayal, because "trust can only concern that which one person can rightly demand of another" (Hertzberg 1988), humans acknowledged the presence of selfishness in their environment (Michiardi and Molva 2002b) and took steps to avoid being victimized by self-centered peers. Any declaration of another's selfishness was dependent on establishing the context of the trust evaluation.

Time and context were two characteristics of the multi-dimensional nature of trust. The time aspect showed that trust was dynamic; a disreputable person could redeem himself through honest actions and a trusted person could become less reputable if he demonstrated deceit. Context was the situation in which trust was being considered. An example of context was that Alice may trust Bob to order wine at dinner but wouldn't trust him to fix her car.

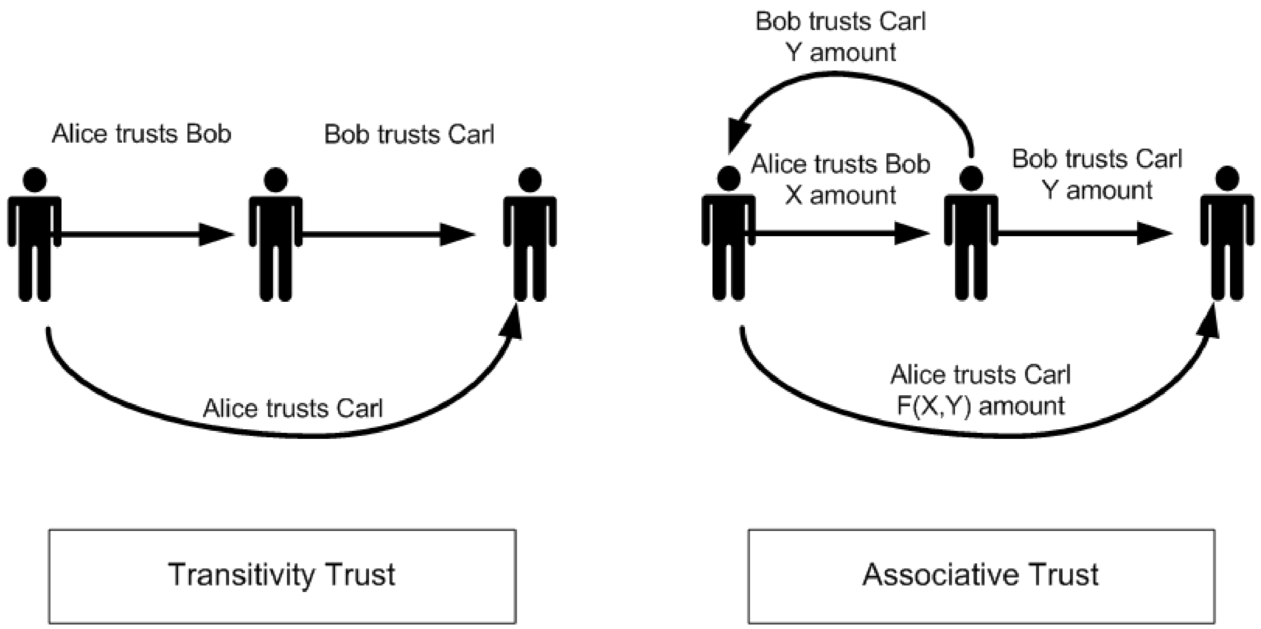


Figure 2-4 Transitive and Associative Trust

Trust could be transitive, as shown in Figure 2-4. If Alice trusted Bob to pick wine and Bob trusted Charles to pick wine, Alice might reasonably trust Charles in wine selection if she were applying transitive trust. Alice could also constrain this trust by context. The constraint meant that, although Bob might trust Charles to split the bill fairly, Alice might have been willing to risk Charles' wine choice but might not be expected to trust the way he divided the check.

Alice might choose to constrain her trust through association, illustrated on the right side of Figure 2-4. This type of trust required Alice to gauge the extent she trusted Bob before asking his opinion on Carl's trustworthiness. Bob would reply with a qualified expression of his estimate of Carl's trustworthiness. Once she had established her trust in Bob and his trust in Carl, Alice combined both trust levels to create her own initial impression of Carl's trustworthiness. Alice's guarded trust or cynicism allowed trust to be expressed in a continuous, rather than discrete, manner as it was in sociological settings.

Expressing trust in continuous terms qualified trust in terms of context (e.g., Alice trusted Bob's taste in wine) or acceptance of risk (e.g., since the bill was only $5, Alice was willing to see how Charles split the check). Individuals evaluated evidence of their peers' behavior, forming a perception of behavior through risking betrayal with each interaction. The means of determining trust was complicated by numerous definitions and applications of trust.

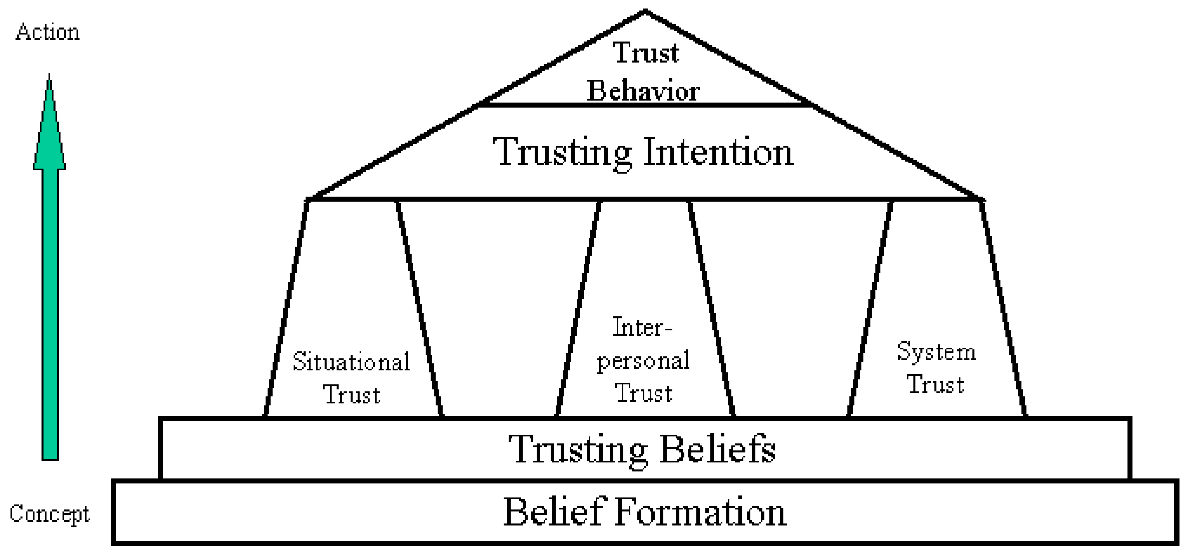


Figure 2-5 Trust Constructs

2.4.1 Trust Types

Given the many, sometimes contradictory definitions of trust, McKnight and Chevrany (McKnight and Chervany 1996) attempted to describe a framework that provided a taxonomy of three types of trust. From this taxonomy (shown in Figure 2-5), they were able to quantify and generalize the process an individual used to influence their behavior. Starting from the bottom of the figure, this model shows how the trust types combine with trust beliefs and are adjusted by trust intentions before becoming behavior; the expression of the trust decision. The way these types and constructs were implemented by this research will be discussed in Section 3.2

System Trust was the extent to which an individual placed trust in the environment around them. In a personal sense, this type of trust reflected a person's feeling of safety in their current location or present situation (e.g., Alice always locked her car doors when she drove through the downtown area.) System trust was built from both structural assurances and situational norms. The individual's belief that the system's rules and regulations would protect them was an example of structural assurance. Similarly, if a user's past experience was that a certain area was risky, the situational norm prompted his System Trust to provide appropriate protection.

Interpersonal Trust was user centric and described an individual's general willingness to extend trust across a broad range of situations to any number of people. This type of trust, also called Dispositional Trust, formed the basis of an individual's approach to interaction. It demonstrated an expectation of other people once their trustworthiness had been evaluated. Interpersonal trust was built from experience, either referred from other trusted individuals or from direct contact with the person in question. This was modified by a trait that McKnight and Cheverny call Trusting Beliefs, reflecting the general tendencies people had toward extending trust. Some people were trusting, believing in the goodwill of their fellow man. Other individuals were more cynical, requiring others to demonstrate their trustworthiness before risking interaction.

Situational Trust described the degree of trust that an individual was prepared to trust any other person in a given situation. This trust was formed upon the intention to extend trust in a particular situation, regardless of what the person knew or did not know about the other party in the situation. It was suggested (McKnight and Chervany 1996) that this type of trust occurred when the trusting party stood to gain with very little attendant risk. Situational trust was different than System trust because there was no implied structural or system safeguards. It was, in short, an individually conceived situational strategy and did not involve an evaluation of the trustworthiness of the other party.

2.4.2 Reputation-based Trust Management

Literature often confused trust with reputation and it was necessary to clearly differentiate between the two concepts. Trust was active; it was extended unilaterally from a node to a specific peer without the guarantee of reciprocity. Reputation was passive; it was the perception of trustworthiness that a node formed about a peer. Reputations were individual in the sense that peers formed different reputations about the same node, based on the fact that they had different experiences or had observed different behavior.

Humans developed the concept of reputation as an aggregation of trust information. They used this concept to predict the actions of others based on historical behavior information gained through personal interaction or the shared observations of peers (Abdul-Rahman and Hailes 2000). Researchers pointed out that reputation could be utilized in a virtual society, such as a DCE, to make up for the lack of the physical, interpersonal clues that humans use to determine trustworthiness.

Resnick (Resnick, Kuwabara et al. 2000) discussed the risks in exchanging valuable information with parties who are identified only through pseudonyms and self-descriptions in centralized systems. They also elaborated on the concept of reputation systems, a means by which the behavior of the participants in information exchanges was tracked and each contributor gained an expectation of the others' actions. The psychological concept that individuals would assist others in order to build and maintain a positive reputation helped instigate a line of research into the application of behavior grading and cooperation incentives in networked systems.

The Sporas system (Zacharia and Maes 2000) thoroughly reviewed the requirement for RM systems in the context of a Usenet-type mailing list (Cyprus-L) and identified the desirable features that a reputation system should have. In particular, the authors discussed the use of reputation in centralized systems such as e-commerce and online communities. These systems reinforced the concepts of trust by emphasizing the conditions of anonymity between participants (due to the pseudonyms) and the need for peer feedback. Through the application of a quantitative procedure, feedback on a specific network member was used to produce a value that attempted to capture the trustworthiness or the expected behavior of that member. The authors' conclusion was that a prospective participant in the system could use that trustworthiness indicator to determine if he should communicate with the other person.

Other researchers applied this conclusion to peer-to-peer file sharing networks (Aberer and Despotovic 2001). A typical problem in a peer-to-peer system was that a user lacked the means to determine the previous activity of other participants in the network before deciding to share files with them. One solution was to have the system collect performance feedback on each participant and transform the feedback into a reputation value, which was associated with each participant's identity. The nodes examined each other's reputation value to develop an expectation of how the other participant might act based on past performance. Trust, therefore, was reliant on a node knowing the identity of its peers.

Without a strict identity construct, peers could assume multiple personas or pseudonyms in the network for malicious purposes. Douceur (Douceur 2002) pointed out that the benefit of having a reputation was limited by the ability for users to adopt unlimited pseudonyms. Concern over anonymity and the Sybil attack spawned an interest in creating an atomic identity for each node in the network. Once a system created an atomic identity for each node in the network, it calculated a reputation and linked each node with its reputation. In addition to linking nodes with their reputations, availability of a global identity enabled the system to track the source of reports and defeat possible attacks by misbehaving nodes through non-repudiation of the reports. Distributed PKI schemes offer a means of affixing a persistent identity to each node in the network. Example PKI schemes include Pretty Good Privacy (PGP) systems, like the one presented by Capkun (Capkun, Buttyan et al. 2002), and X.509-based certificate schemes (DaSilva, Midkiff et al. 2004).

Three types of reputation systems have been studied: positive reputation, negative reputation, and a combination of both. Positive reputation systems only recorded positive behavior observations or feedback. An example of such a system was CORE (Michiardi and Molva 2002a). CORE used positive feedback to enforce node cooperation in a network. In the CORE system, nodes built positive reputations by collaborating (i.e., routing traffic) with their peers. The drawback of this system was its reliance on positive reports, without the facility to submit negative feedback.

Negative reputation systems only shared complaints or negative behavior observations. Peers were assumed to be trustworthy, so behavior feedback was used to modify a node's reputation negatively. Systems that utilized only negative feedback (Marti, Giuli et al. 2000) lacked an ingrained incentive for a node to participate in the network. Since a node's reputation could only decrease, the system promoted selfishness (i.e., a node only participated when it wanted something) or malicious behavior (i.e., a node actively pursued a poor reputation so that it would never be asked to expend resources to help its peers.)

The drawbacks of negative reputation systems were addressed in the CONFIDANT system (Buchegger and Le Boudec 2002b; Buchegger and Le Boudec 2002a), which extended the notion of reputation to include both positive and negative reports. This extension fostered the elimination of misbehaving nodes through isolation. Positive and negative reports had the same weight and the system differentiated between first hand observations (i.e., behavior observed by the node compiling the reputation value) and second hand observations (i.e., behavior observations shared between trusted peers). CONFIDANT policed the network through rating lists and a scheme of weighting observations. The effects of including second hand observations were deemed beneficial to the reputation system, as they broadened a user's knowledge base (Buchegger and Le Boudec 2003a).

Rather than use feedback to calculate reputations, researchers have constructed a directed, weighted graph to represent the relationship between nodes (Mui 2003). In this technique, an edge was placed between two nodes to symbolize an association. The edge weight was the measure of trustworthiness between the source and the destination of the edge. Because the weighted graph allowed for transitivity of trust, the trust between any two nodes in the network could be determined by summing the weights of the edges between the two nodes. The two major problems with this approach were its assumption of transitivity of trust and its implied assumption that a node knew (but does not necessarily trust) about all of the members of the network. The algorithms that were presented did not allow for mobility or transience in the network as these characteristics were expected to change the number, weight, or existence of edges in the graph.

As stated throughout this research, dynamic collaborative efforts were not suitable environments for the employment of rigid security controls, such as RBAC or MAC. Instead, the use of soft security (Rasmussen and Jansson 1996) used trust and reputation as social controls. These controls were used to determine when to share information between members of a dynamic community. Because soft security acted as a social control rather than a hard gate, like passwords, role assignments, or authorization certificates, it resisted the failures that occurred when a gate failed or a user did something unexpected. The benefits of soft security are discussed in more detail in Section 3.3.2.

This section has discussed how other researchers have formulated reputations. Throughout their work, however, it was left to the user to determine how the reputation value was applied. The remainder of this dissertation focuses on the application of trust and reputation to the area of access control.

2.4.3 Trust-based Access Control in Wireless Networks

Researching and evaluating trust-based access control systems was full of hidden challenges and obstacles. Some systems like the Vigil project at University of Maryland, Baltimore County (Kagal, Finin et al. 2001; Kagal, Undercroffer et al. 2002) claimed that they included a trust evaluation in the access control decision. Upon deeper investigation, however, this evaluation was performed within a centralized authorization server, robbing Vigil of any claim of being a distributed system.

In the Vigil system a node presented its credentials and resource requests to the authorization server. The server verified the node's identity and invoked a policy enforcing mechanism (which is confusingly called a "trust agent") to interpret the policy rules embedded in the node's credentials. These policy rules were implemented as PROLOG statements. The "trust agent" evaluated the statements against its knowledge base to produce an access control decision.

Researchers expanded the Vigil system to implement a basic weighted reputation scheme (Perich, Undercroffer et al. 2004). This reputation scheme was constrained by relying on first hand information only. In other words, it did not update the reputations of peer nodes based on input received from other peers or a network monitor. The reputation value was evaluated to produce a binary trust value (e.g., reliable or unreliable.) Whether due to time or scenario constraints, the reputation system was not illustrated in the context of the complete Vigil system. It also lacked any facility to allow for nodal rehabilitation or behavior history.

A better example of trust-based access control in a wireless environment was offered by the SECURE project (Gray, O'Connell et al. 2002). This project investigated the application of trust-based methods to enable short term, ad-hoc collaboration between handheld devices. In the SECURE project, nodes joined the network by requesting access from a self-appointed group leader. The leader then queried the other group members and they voted on whether or not to admit the new node. SECURE did not attempt to implement a behavior history because nodes did not possess a verifiable, non-reputable identity and, therefore, any behavior history would be rendered suspect by the possibility of a Sybil attack (Douceur 2002). Only after a node was admitted to the network could its peers track its behavior.

Group members recorded and evaluated behavior during each session. A node could be excluded if it demonstrated undesirable behavior, but SECURE's method for aggregating behavior was not responsive enough to catch any but the slowest or most persistent malefactor. The authors admit that their method was susceptible to the Sybil attack and did not implement trust thresholds due to the lack of a persistent behavior history. These deficiencies made the project's results unsuitable for implementation in a situation where the system was protecting sensitive data.

2.5 Conclusion

The DCP, discussed in Section 2.1, set out requirements for a fluid, decentralized system of access controls. Coalition members needed to be confidant that they would have access to the tools and information that they needed to fulfill their roles while, at the same time, were reassured that they would retain control over the assets that they provided to the coalition. Centralized solutions were unacceptable, as few coalition members would willingly subordinate themselves to any of the others. While RBAC did exactly what it was designed to do (Section 2.2) by providing a shortcut to assigning and managing permissions to users, its centralized architecture and pre-configuration requirement made it unwieldy in a dynamic environment. Researchers needed to break away from the old centralized RBAC applications and find new decentralized ways to protect information and assign privileges.

By analyzing the risks, threats, and use of incentives, researchers made preliminary proposals for the use of trust incentives to encourage positive behavior in virtual societies such as computer networks. Token-based incentives, discussed in Section 2.3 had the benefit of being intuitive but brought a host of economics-related issues, rendering the solution more challenging than the problem. Section 2.4 introduced the concept of trust and the possibility for its use in dynamic access control. When researchers tried to use trust concepts (Section 2.4.3) as determinants for access control, their work stopped short of either completely decentralizing access control decisions or of maintaining behavior history so that the system could sustain an operational life rather than a short-term sense of protection.

The following chapter discusses a proposed solution to these problems, developed through this research. A detailed design of a decentralized, trust management-based, access control system is set out as a component of a network's security mechanism. This design is then set into an experimental environment for testing.

3 System Design

This chapter presents the trust management system (TMS) design developed through this research. As stated in Section 1.3, the purpose of this research effort was to design, implement, and test a decentralized, trust-based access control system for use in dynamic collaborative environments (DCEs). By assessing user behavior and gauging the uncertainty in the neighboring user community, the system facilitated efficient and effective access control. The TMS design was the key contribution of this research, replacing the classical centralized security architectures and providing adaptable security in a DCE.

Section 3.1 describes the hybrid nature of a DCE's communications and processing environment, including a discussion of the system security architecture that is present on each DCE member's node. This section illustrates the position of the TMS as a decision-making layer that supports the key management system (KMS) with assessments of trustworthiness.

Section 3.2 describes modules that perform the functions of the trust constructs. The application of trust as a criterion for access control required the resolution of a number of issues. The terms and concepts presented in Section 2.4 formed the foundation of this research, as the TMS was the first access control system to implement and apply all three types of trust that were described by McKnight and Chervany. This discussion is followed by a simulation design, presented in Section 3.3, which is specifically adapted to test a DCE.

Section 3.4 explains the metrics used to evaluate the TMS's performance. This section also discusses the testing development and describes the method used to construct the tests. Having detailed the groundwork of the TMS, Section 3.5 summarizes the system design.

3.1 System Overview

This section investigated access control challenges that were specific to a DCE. The concepts of guarded collaboration, as discussed in Sections 2.1 and 2.2.4 were developed into system requirements. These requirements were then framed as a node's security architecture, of which the TMS formed the middle layer.

3.1.1 Access Control in a Dynamic Collaborative Environment

A basic challenge in coalition information systems was that the coalition did not own any organic resources. All of the resources needed by the coalition were contributed by participating organizations. An aspect of collaboration was that nodes participated willingly to achieve a collective goal. In a DCE, nodes contributed resources such as printing, file storage, file sharing, and processing power. Resource-owning nodes shared to help other members but might join and depart the DCE based on internal goals and objectives (Li and Mitchell 2003). While a participating member desired cooperation, it also required that it retain ultimate authority over the resources it controlled prior to joining the DCE.

The following notional coalition provided an outline of a DCE's general characteristics. In particular, this description highlighted the disparity in resources and requirements that drive the access control requirements of a DCE. The resources, training, and structure of the organizations that contributed to the resolution of a crisis spanned a wide range.

Some coalition members arrived as self contained, self-sustaining units (the National Guard or the Coast Guard), each complete with a stable hierarchy, operating procedures, and sufficient resources to perform their intended role. Other organizations (like police, rescue squad, and fire departments) had limited or specialized resources that they were willing to integrate into a DCE but, in turn, needed to leverage other members' capabilities to fulfill their increased responsibilities during the incident response. Additionally, non-governmental organizations (NGOs) and private volunteer organizations (PVOs) like the Red Cross and the local church's Doughnut Squad arrived to assist the coalition with lots of enthusiasm but few resources to support them. Since these organizations played a tremendous role, they could not be turned away but were usually the most outspoken in their refusal to subordinate themselves to any central (especially military) authority.

Because the coalition had no set structure or membership, it was feasible that members of the coalition did not implicitly trust each other. Judging a coalition member's motivation for joining an emergency relief effort was fraught with dangers of misperception and emotion. What could be measured, however, was the demonstrated trustworthiness of the participants. The trust management system used trustworthiness to make access decisions, in effect deciding whom to trust before sharing or asking to share resources.

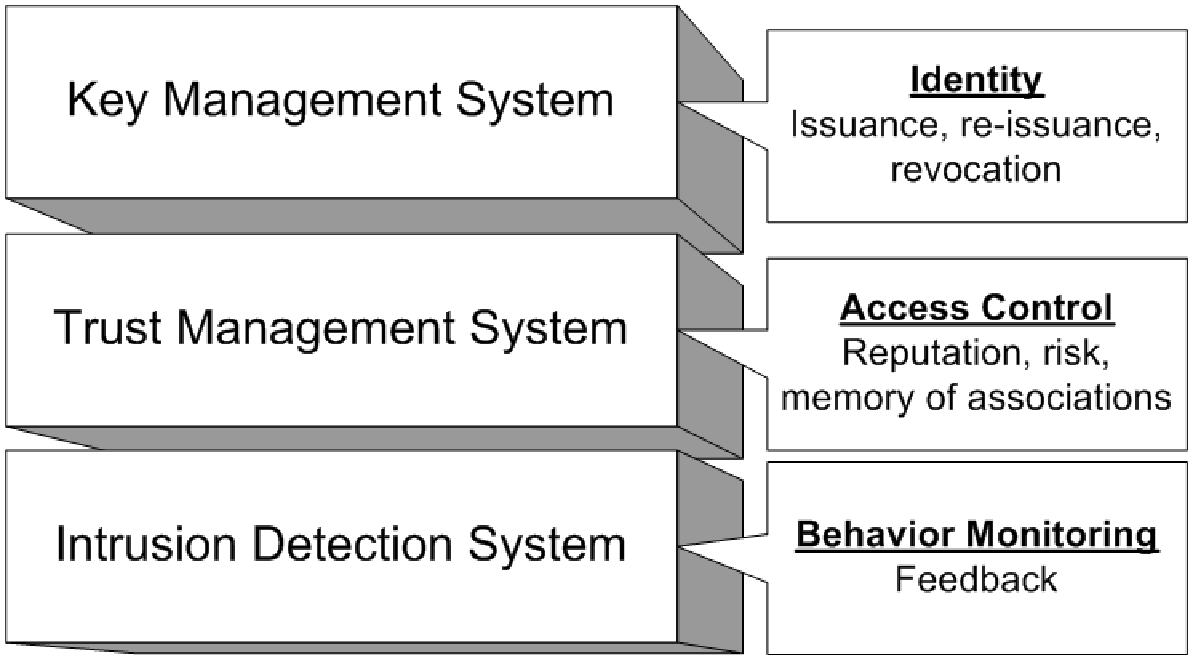

Key Management System
Trust Management System
Intrusion Detection System
Identity
Issuance, re-issuance, revocation
Access Control
Reputation, risk, memory of associations
Behavior Monitoring
Feedback

Figure 3-1 System Security Architecture

The resources were accessed through a hybrid network of wired and wireless communications links. DCE members used computing platforms that ranged from desktop workstations to handheld personal digital assistants (PDAs) to collect, process, and exchange information. Each DCE member was part of a system-wide security architecture that regulated access to resources and information. Members implemented this architecture as a set of autonomous agents on each node under their control. The next section describes this architecture and develops the TMS design.

3.1.2 Nodal System Security Architecture

Each member node contributed to the system security architecture, as shown in Figure 3-1. Each node executed a three-layered security agent that implemented this security construct. Some layers, like the KMS layer, contributed to the DCE at large, while others, like the Intrusion Detection System (IDS) layer, were focused more on the individual node. These agents were autonomous, in that the parameters were set by the operating node and not by network-wide security policies.

An agent-based approach was selected because of its suitability to a mobile collaborative environment (Perry, O'Hara et al. 2001; Bartram and Blackstock 2003). Each node possessed a complete security system and could operate independently based on peer nodes that were known to it or observations made first hand. A node could also join a coalition or collaborative group and take advantage of the group's information. The node retained this information when it chose to leave the coalition or the group's network area.

The KMS managed user identity certificates and established the rules for issuing, reissuing, and revoking certificates (Hadjichristofi, Adams et al. 2005a). In a centralized network, this KMS relied on directory replication and certificate revocation lists (CRLs.) In a decentralized environment, the goal was to provide the KMS with access control decisions based on the trustworthiness of the perspective peer node.

The TMS was implemented as a central data-processing layer of the overall system security architecture. The TMS provided the KMS with a layer of abstraction of the overall trustworthiness of nodes, based on the activity of the nodes in the network. As the central layer, the TMS determined whether to trust or distrust its peers based on its individual trust thresholds. The trust management system then reported its trust decisions to the KMS for its consideration.

At the lowest layer, an IDS or network monitoring scheme (Buchegger and Le Boudec 2002a; Michiardi and Molva 2002a) provided periodic performance observations to the network. These observations were distributed throughout the system in a modified epidemic routing algorithm, similar to the selective dissemination scheme proposed by Datta (Datta, Quarteroni et al. 2004). The architecture's lowest level was simulated, as its specification and construction was beyond the scope of this research.

The following sections develop the requirements for the trust management layer and detail the theoretical model underpinning its construction. First, we examine the requirements for building and using reputations in a virtual society or collaborative group. Then the TMS inputs and outputs are identified before the internal processes of the TMS are detailed.

3.1.3 Identifying Trust Management System Requirements

Having established the location and general function of the TMS in the system security architecture, we looked at the inputs and outputs the TMS will require. In particular, we needed to identify the information necessary to collect, construct, and utilize reputations of peers within a virtual society. The eBay Feedback System (EFS)[4] was examined as an example of a widely used reputation system to determine suitable system requirements (Keser 2003; Shmatikov and

[4] eBay, "eBay's Feedback System," available from http://www.eBay.com.

Talcott 2003). The EFS was chosen because it enabled a behavior grading system in a large, well-documented environment. Through the eBay website, the EFS aggregated positive and negative comments made by buyers and sellers to provide customers with some sense of the reliability or trustworthiness of a person they are considering doing business with.

The EFS introduced three features that were applicable to reputation management in general: positive and negative feedback, reputation aging, and identity. In the EFS, buyers and sellers left positive or negative feedback on each other's performance after conducting a transaction. Positive comments had the same weight as negative comments, meaning that a compliment had the same effect on a reputation as a complaint. A similar situation existed in a collaborative environment when two nodes participated in a file sharing or information exchange. Peers submitted positive feedback when a transaction was completed in line with their expectations. Transactions that were incomplete or unsatisfactory (e.g., the file was not as advertised, the service was too slow) resulted in the submission of negative feedback. The presence of both positive and negative feedback was deemed necessary for a complete reputation management system (Buchegger and Le Boudec 2003b).

The use of feedback raised the requirement for the EFS's second feature. This feature was the need to age or fade feedback (Buchegger and Le Boudec 2003b) to prevent reputation gaming, as discussed in Section 3.3.2. Aging feedback diminished the impact older behavior feedback items (FIs) had on the reputation calculation. If the system did not age feedback, a comment made years ago had the same weight as a comment made on a current transaction. A malicious individual could take advantage of this weakness to build up a high reputation over time and then default or cheat without incurring much damage to his reputation. Aging FIs made this sort of attack more difficult because a user's performance had to be constantly maintained to sustain his positive reputation.

The third requirement observed in the EFS was identity. In the eBay system, a member's reputation rating reflected the number of other distinct members that have left feedback. Because the source of the FI was recorded, the EFS discarded duplicate items, hindering the opportunity for a single node to have undue influence over another node's reputation. For example, even if Alice left four positive (or four negative) comments on four distinct transactions, only one positive (or negative) item was added to Bob's reputation because all four pieces of input came from the same buyer. The EFS used login-password combinations to identify the user submitting feedback but other distributed systems used a PKI-based KMS to provide each member with a persistent identity (Khurana and Gligor 2000; Shin, Ahn et al. 2002; Thompson, Essiari et al. 2003).

In addition to the requirements derived from the EFS, a decentralized environment posed additional challenges for reputation management. Member nodes were not restricted to a single location or access point to obtain network services. Nodes could enter or leave network coverage and continue to operate in peer-to-peer mode. This characteristic was called nomadic membership (Fenkam, Dustdar et al. 2002; Suryanarayana and Taylor 2003).

When a node decided to establish an association with a new peer, there needed to be a procedure for each side of the transaction to establish the partner's identity and gain preliminary trust information without relying on a central authority or directory. This procedure was similar to the way individuals introduce themselves in social situations. Some systems (Dewan and Dasgupta 2004) performed introductions by passing a reputation value or used a voting mechanism (Gray, O'Connell et al. 2002) to extend trust to new associates. The node soliciting the introduction could not determine how or why the prospective associate had established a particular trustworthiness because of their lack of evidence to support the reported trust level. We concluded, therefore, that an independent determination of trust required a node to examine evidence of the prospective associate's behavior.

A more effective introduction process included a mechanism for the two prospective associates to share observed behavior history in such a way that they could derive the reputation of their prospective partner by having the proof to substantiate the given value. In our target environment (DaSilva, Midkiff et al. 2004), a node polled the Delegated Certificate Authorities (DCAs) and its Trusted Peers (TPs) for the new associate's identity certificate and behavior history. As a result, an effective reputation management system had to keep a certain number of its behavior observations so that it could provide non-reputable evidence to other nodes.

The preceding analysis examined both centralized systems (e.g., the EFS) and decentralized environments to collect requirements for a trust-based system. The following sections will discuss the sources for identities and behavior evidence, as they are external to the TMS layer. Internal mechanisms, such as reputation aging and the introduction process, will follow as part of the discussion on the TMS design.

3.1.4 Elements from the Key Management System

In a well-connected and hierarchically organized DCE, the KMS had the ability to provide a control plane of authentication services. This ability was constrained by connectivity, lack of a naming policy, and the Dynamic Coalition Problem (DCP). Because of these constraints, there could be no expectation that all DCE members had verifiable identities, since not all members would be willing to surrender their autonomy to the DCE.

In many ways authentication presented the same requirements as authorization and was vulnerable to the previously mentioned constraints. Authentication required cryptographically verifiable credentials but the possession of identity credentials did not equate to verifiable permissions. The result was that the KMS was relied upon to handle identity credentials but that these credentials assumed less importance in a DCE than in a more controlled and organized environment.

The KMS, like the other layers of the system security architecture, resided on each node. Within each node, the KMS declared the node's identity to peers and established secure information exchanges with associates in the DCE. These associates were TPs and provided referrals to each other. Between associates, the KMS layer publicized the establishment and dissolution of associations to specially designated nodes called DCAs.

Within the security architecture, the KMS asked the TMS for trust assessments on specific users. The KMS provided the identity of a prospective associate and received a Go/No-Go assessment of that user's trustworthiness in return. These assessments allowed the KMS to accept or decline offers to associate with other users.

3.1.4.1 Identity Imprinting

The concept of imprinting an identity on a network peer was borrowed from the world of biological sciences (Stajano and Anderson 1999). The imprinting process required a node to declare its identity upon entering the network. This identity could have come from any one of three sources. First, the node's parent organization could have issued identity credentials before the node joined the DCE. Second, the node could have applied to another DCE member to issue credentials signed by a local DCA. Third, the node could have issued its own credential.

Because the KMS accepts the difficulty in verifying identity credentials, a node's declared identity was only used as an index for behavior grades. A user might create any number of aliases but these were all linked in some way to their declared identity. The user was discouraged from creating aliases by the reputation-scaling mechanism.

3.1.4.2 Reports

The KMS recorded instantiations and dissolutions of trust within the system. Because of our target environment's use of IPSec to secure communications, these extensions of trust were implemented as security associations between peer nodes. Nodes notified the KMS of the association's status. The KMS then shared this information throughout the network. The notification messages were called reports to differentiate them from observations gathered from other nodes and were treated as trusted, global, information

The KMS implemented two types of reports: registrations and complaints. Registrations and complaints were specially formatted messages that a node sent to notify the KMS of the establishment or dissolution of trust.

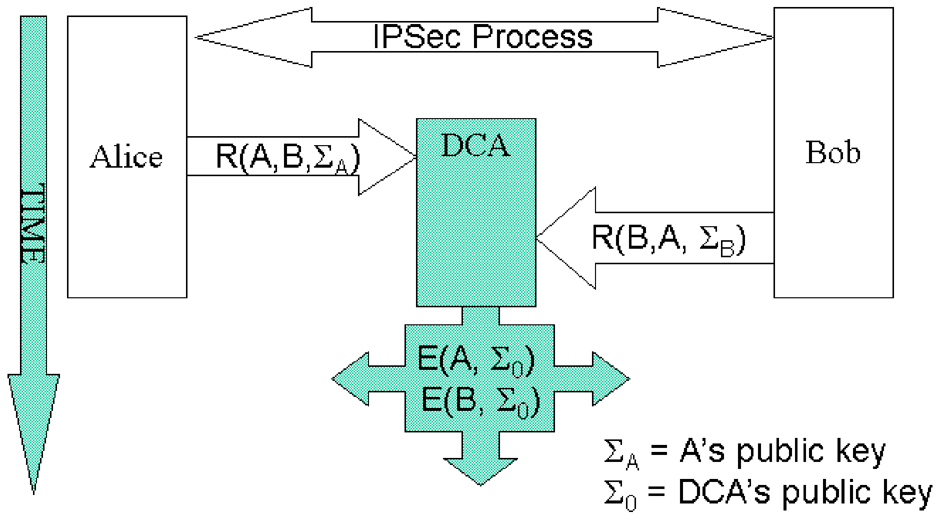


Figure 3-2 Registering a Security Association.

The registration message, illustrated in Figure 3-2, was specially formatted and signed by each node to provide non-repudiation. The message specified the identity of the nodes in the association and the signature of the node submitting the registration. Upon receipt, the DCA assigned the event message a serial number and broadcast two establishment messages to the network. The establishment message notified everyone in the network of the new association.

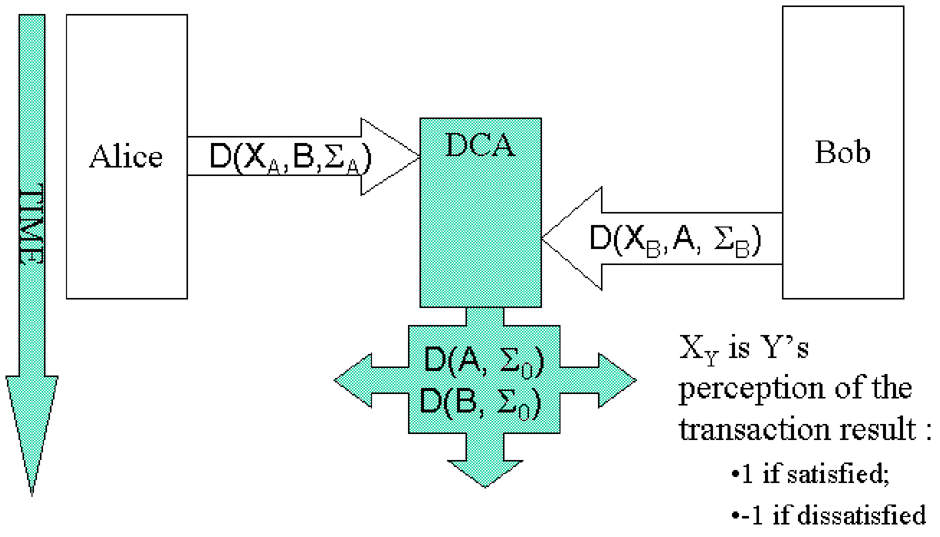


Figure 3-3 Dissolution Process

While the DCE was making its notification, both associates updated their list of TPs with each other's identification and began collecting behavior information on their new associate. Once the transaction was completed, both sides reported the dissolution of the association to the KMS. The dissolution message, illustrated in Figure 3-3, included an indication of each party's satisfaction with their partner's behavior during the transaction. This message type was treated as a signed event by the KMS for the purposes of auditing. Like the registration message, the dissolution message was broadcast to the network.

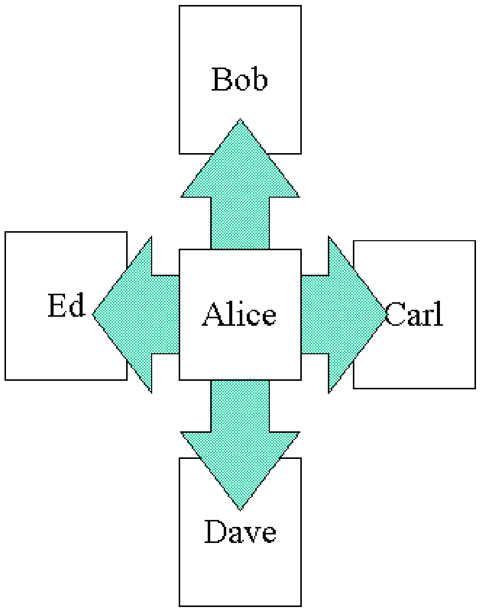
3.1.5 Elements from the Intrusion Detection System

The IDS or network monitor provided periodic performance observations on peers in the network (Buchegger and Le Boudec 2002b; Michiardi and Molva 2002a). The TMS informed the IDS of what to observe by providing lists of peer identities and contexts. Observations, it should be noted, were records of an individual node's expectations. Because observations stemmed from perceptions, they are not completely trusted but are used to confirm or augment reports received from the KMS.

The observations compared a node's expectations against the observed performance of its neighbors. Observations were made on trusted peers as well as on neighboring nodes that were within "listening range" but were not necessarily directly trusted. Nodes observed performance in areas such as resource sharing or file access and periodically generated positive or negative observations.

A node received a “good” observation by doing what was asked of it. If Alice asked Bob for a file or to print an email and Bob agreed, Alice gave him a positive behavior observation that she shared with her other trusted peers. If Bob refused, she gave him a negative observation, regardless of the reason he had for refusing. If Bob did not answer her request, Alice could assume that he had moved out of range or was asleep, withholding any observation. This was acceptable because the TMS was an autonomous but not intelligent or rationalizing decision making system.

The observations contained the identifiers of the observer and the observed, an observation, and the observer’s signature, as shown in Figure 3-4. These observations were proliferated through the network in a modified epidemic routing algorithm, similar to the selective dissemination scheme proposed by Datta (Datta, Quarteroni et al. 2004), to spread information between trusted peers rather than flooding the network with observations.



- Broadcast to all TPs and neighbors
- Contains:
  - Msg Type
  - Observer ID
  - Observed ID
  - Observation:
    - 1 if satisfied;
    - -1 if dissatisfied
  - Observer Sig

Figure 3-4 Composition of a Behavior Observation

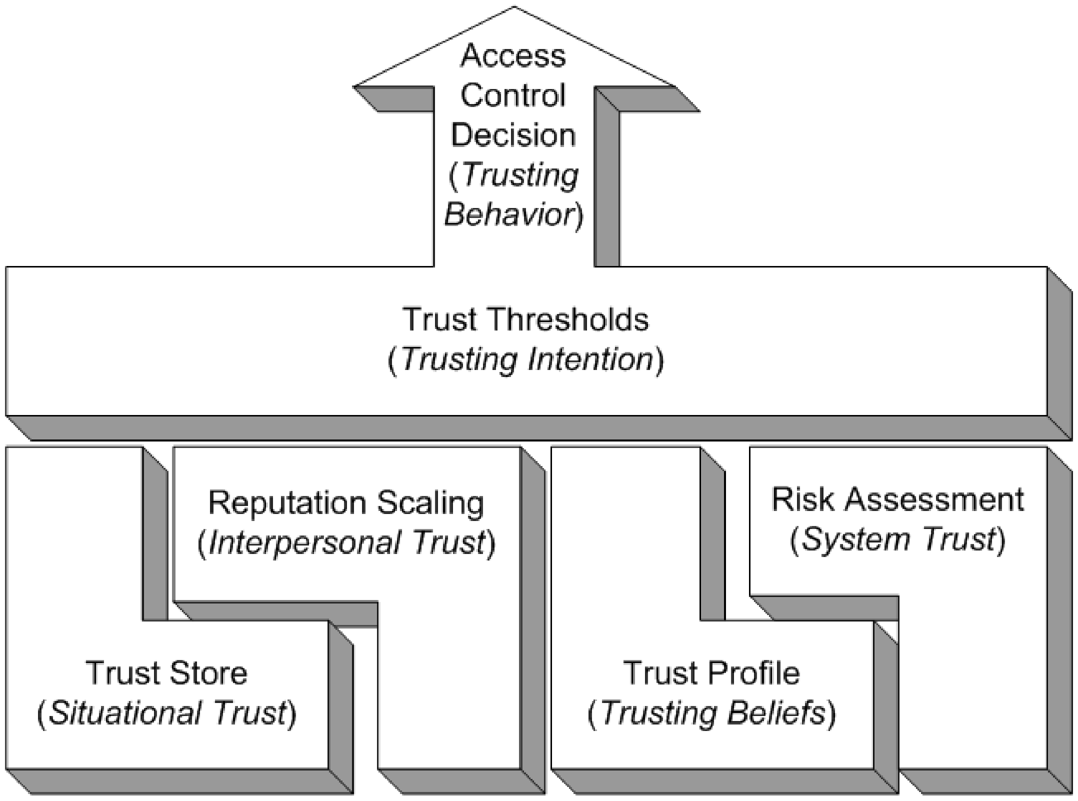

Access Control Decision (*Trusting Behavior*)
Trust Thresholds (*Trusting Intention*)
Reputation Scaling (*Interpersonal Trust*)
Risk Assessment (*System Trust*)
Trust Store (*Situational Trust*)
Trust Profile (*Trusting Beliefs*)

Figure 3-5 Implementation of Trust Types and Constructs

3.2 Implementing Trust

"Love all, trust a few. Do wrong to none."

William Shakespeare

Previous sections described how the KMS and the IDS layers of the system's security architecture provided reports and observations to the TMS in return for assessments or information. Within the TMS, the KMS-provided identity was used to anchor behavior information collected on an associate. The source of the information was included in the assessment of the associate's behavior, resulting in a reputation index that served as an expression of the associate's trustworthiness.

The TMS then applied the trustworthiness estimate to authorization decisions, whether we call these decisions access control or privilege management. Trust-based decisions were useable in distributed system security because traditional, centralized authorization mechanisms were perceived as inadequate (Blaze, Feigenbaum et al. 1999). A TMS gave the ability to identify misbehaving nodes that moved around in the network, with the intention of isolating them from the rest of the network. This isolation was achieved when the “good” nodes refused to interact with the “bad” nodes.

This research developed a system of trusted agents and user nodes that cooperated to exchange behavior reports and establish a record of each node’s behavior history. This history, based on reports and observations, was expressed as a reputation index (RI). The RI, with evidence in the form of signed FIs, provided an expectation of their partner’s behavior before entering into or dissolving an association. By providing an indication of each other’s trustworthiness, nodes avoided misbehaving nodes.

Starting with the theoretical work conducted by McKnight and Chervany (McKnight and Chervany 1996), trust types and constructs were combined into modules and procedures, shown in Figure 3-5. These components were then linked through information flows to create the trust management system. The augmented trust construct diagram is shown in Figure 3-6.

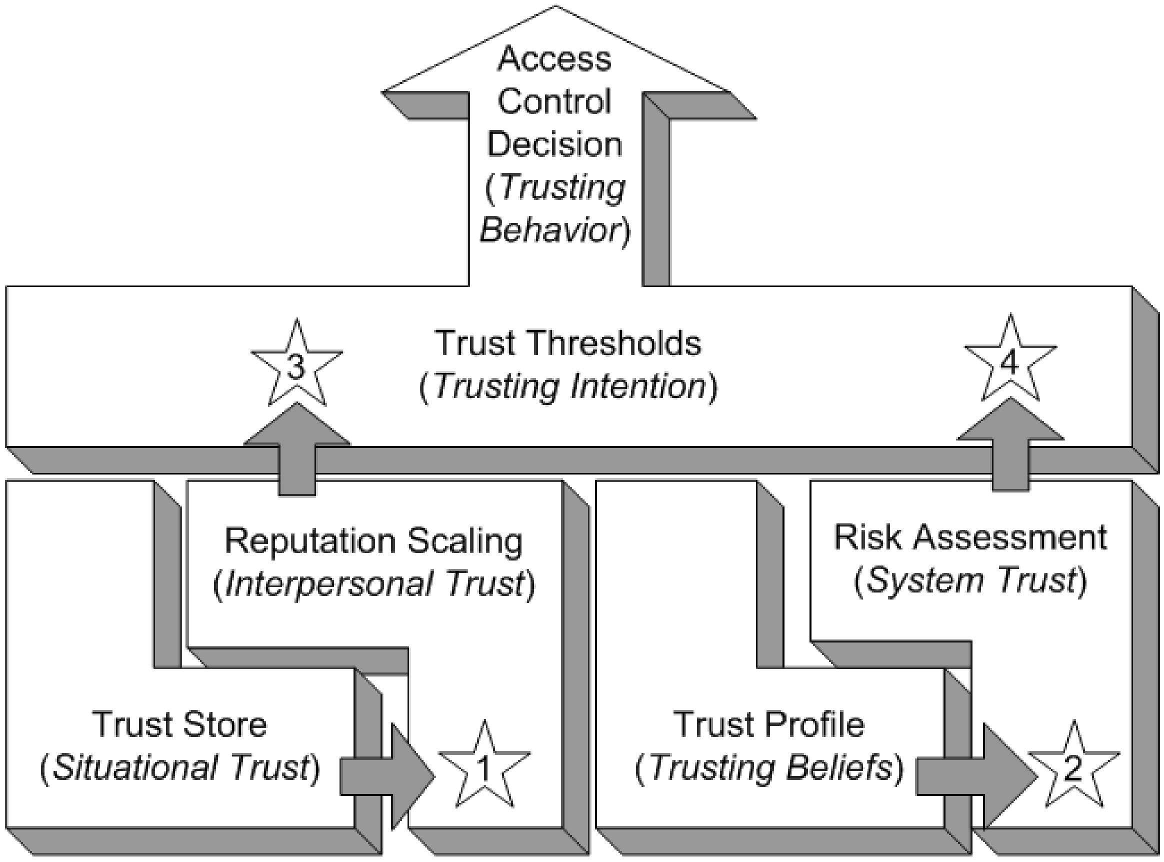

Access Control Decision (*Trusting Behavior*)
Trust Thresholds (*Trusting Intention*)
3
4
Reputation Scaling (*Interpersonal Trust*)
Risk Assessment (*System Trust*)
Trust Store (*Situational Trust*)
1
Trust Profile (*Trusting Beliefs*)
2

Figure 3-6 Trust Constructs with Information Flows

First, the trust store provided evidence to the reputation-scaling process in the same way that a situational trust construct provided the basis for the establishment of interpersonal trust. At the same time, the individual's innate tendency to extend trust guided the processing of global information to determine the current state of trust in the system or the surrounding the person found himself in. The information was processed and passed to a higher-level process to assert the trust intention.

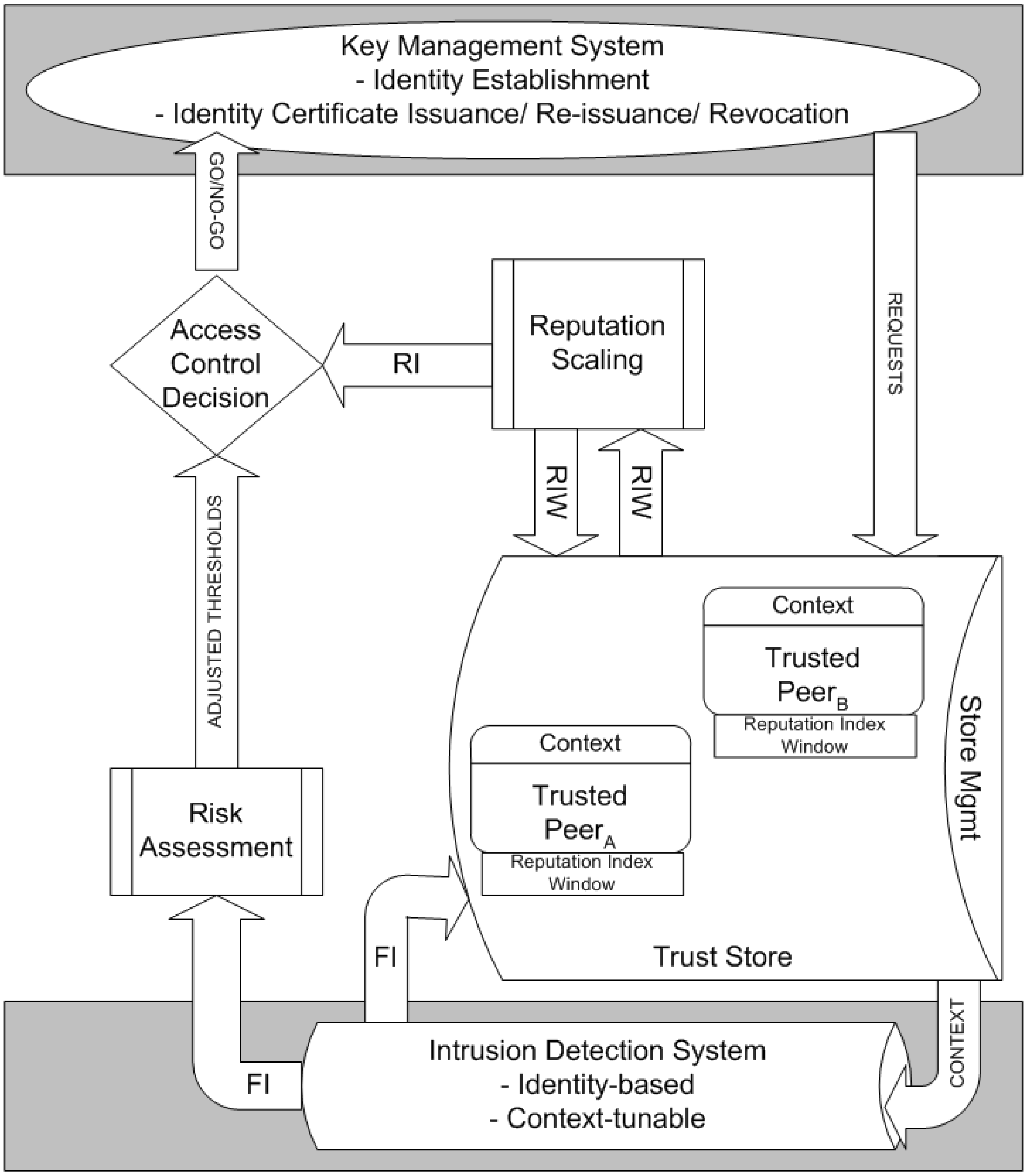


Figure 3-7 Trust Management System Architecture

The trust intention was described as the current state of trustworthiness of an associate in the prevailing circumstances. Using interpersonal and system trust, the trust intention compared the associate's actions with the general risk state in the local area to produce a trust decision.

The transformation of the trust construct relationships to an operational system suitable for a mobile dynamic collaborative environment was fairly straightforward. Previous sections of this chapter discussed the quantitative mechanisms and the input/output relationships between the modules that represented the constructs. The system design, therefore, was the blueprint for the implementation of Chevarny and McKnight's hierarchy of trust.

Illustrated in Figure 3-7, this system design was centered on the trust management components that reside on each node of the network. The KMS and IDS were depicted at the top and bottom of the diagram, respectively. Arrows show how the modules exchanged information from these external entities. These flows were the same as discussed in Figure 3-6 but were described in terms of the information component rather than the conceptual element.

The center of the diagram shows how a node processed, evaluated, and stored the behavior information using the trust store (TS) and the reputation-scaling module (RSM). Risk assessment, a background process, continually adjusted trust thresholds based on current network conditions. When the KMS requested a trust decision, the prospective associate's reputation was compared to the current trust threshold. Once the evaluation was complete, the TMS forwarded an access control decision to the KMS. The remainder of this document concerns itself with the specification and simulation of the TMS operation.

3.2.1 Trust Profiles

A trust profile provided an easy way to quantify a user's initial trust expectations. Trust thresholds, which represented the amount risk a user was prepared to accept when forming or maintaining associations, were one example of information that was contained in the trust profile (Prietula 2000). Constraints on the risk assessment and reputation-scaling algorithms were also included in the profile.

Users established how trusting they could afford to be based on their individual goals and objectives. They selected the trust profile that most closely aligned with these individual needs, establishing their initial state before entering a collaborative environment. In a general sense, users were grouped into four trust profiles after Prietula's work (Prietula and Carley 2001). The general tendencies are illustrated in Figure 3-8 and explained below.

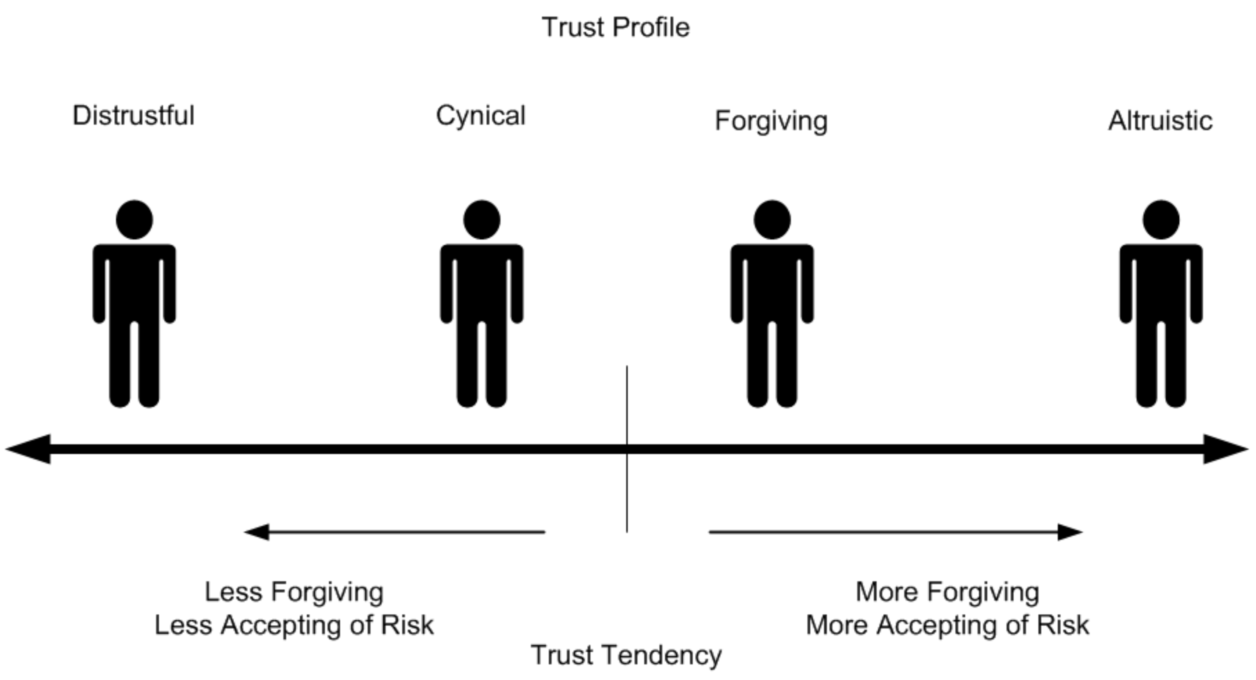


Figure 3-8 The Relationship of Trust Profiles to Trust Tendencies

Altruistic users did not interpret the behavior of their peers. Instead, they performed services for the network for their own benefit. Altruistic users trusted any peer and did not maintain reputation indices. Because of this constraint, peers could not receive referrals from Altruistic users, as they had no evidence to share.

Forgiving users exhibited responses to peer behavior by adjusting the reputation of their peers in a positive or negative manner based on direct experiences or the observations of other peers. Forgiving users aggregated peer reputations in such a way as to allow a misbehaving peer to redeem or rehabilitate its reputation given a sustained period of positive or desirable behavior. Forgiving users established trust thresholds and evaluated peer reputations against these thresholds to determine a peer's trustworthiness.

Cynical users were similar to Forgiving users in that they exhibited responses to peer behavior and maintained reputations of the peers who they interacted with. Importantly, however, Cynical users did not allow trust rehabilitation. Once a Cynical user determined that a misbehaving peer was untrustworthy, he made no attempt to collect further behavior information or readjust the peer's reputation.

Distrusting users did not trust any other user. These users typically relied on MAC or RBAC to form associations rather than trust. Because of this requirement for pre-configuration and run-time verification, a distrusting user would be unable to operate in an ad-hoc environment, except perhaps through contact with altruistic users.

An example of the initialization settings for each trust profile is given in Table 3-1. These values were determined through analysis and testing to offer gradient levels of protection when the user joined the network. These values, like those published by other researchers (Michalakopoulos and Fasli 2005), provided a starting point or basic level of assurance. The TMS, as a self-protecting system, would begin adjusting these levels to current network or environment conditions immediately. A constraint listed as “None” indicates that the risk or reputation values are allowed to rise and fall based on periodically updated information. The Cynical user’s “Rising Only” constraint reflects the profile’s inclination to raise its thresholds without ever relaxing them, regardless of the current information.

Table 3-1 Example Initialization Values Based on Trust Profiles

Trust Profile	Trust Threshold	Distrust Threshold	Risk Constraint	Reputation Constraint
Altruistic	-0.99	-1	None	None
Forgiving	0.6	0.3	None	None
Cynical	0.7	0.4	Rising Only	Rising Only
Distrusting	NA	NA	None	None

This research focused on what was considered the worst-case user – the Forgiving trust profile. In this profile, both reputation and risk were allowed to increase and decrease based on current information. This feature made the user accept risk and maintain the trustworthiness state of their peers. The mechanisms and effectiveness of the TMS in performing these tasks are described in the following sections.

3.2.2 The Trust Store Module

“Forgive your enemies, but never forget their names.”

John F. Kennedy

The TS was the TMS’s memory. It held the behavior history of associates, providing a virtual space for maintaining and storing the behavior history of nodes that the user had interacted with. The TS was also responsible for the very important recognition process, implemented as a memory management algorithm.

The remainder of this section discusses the goals and objectives of the TS. The format of the Atomic Behavior Record (ABR) is presented, followed by an explanation of the memory management algorithm, the recognition process and, finally, the introduction procedure.

Where Interpersonal trust was dependent upon peer behavior trends and System trust was determined through an evaluation of system behavior tendencies, Situational trust was independent of the behavior of other users altogether. This type of trust used the trust store, representing the user's memory of previous peers and situations, to determine what action it would take. A situational trust decision was predicated on remembering a previous decision that had yielded a positive outcome, regardless of the behavior of peers that may or may not have been involved.

The amount of memory had a direct impact on which associates could be remembered and how much information could be maintained on each associate. In theory, a system preferred to keep everything possible. In reality, however, there were two practical limitations on the amount of memory that was possible or practicable to devote to maintaining trust information.

The first limitation was that a user's node has a finite memory. Any memory consumed by the trust system was unavailable to other processes required by the node. The TMS required storage space to hold trust information. It also consumed communications capability by sending and receiving trust information. Since the trust system operated in the background, it was desirable for this system to occupy as little space as possible in the node's memory.

The second limitation was that trust information had a finite time duration during which the reports and observations were applicable. Users were expected to change their behavior, so keeping older information yielded diminishing returns. Because the system could not assume a shared time source, each user had to have a mechanism to eliminate old information. For the behavior grades, this mechanism was designed as part of the 3Win algorithm. For the identities and behavior information for past and present associates, the Trust Store managed a queue of records, called ABRs.

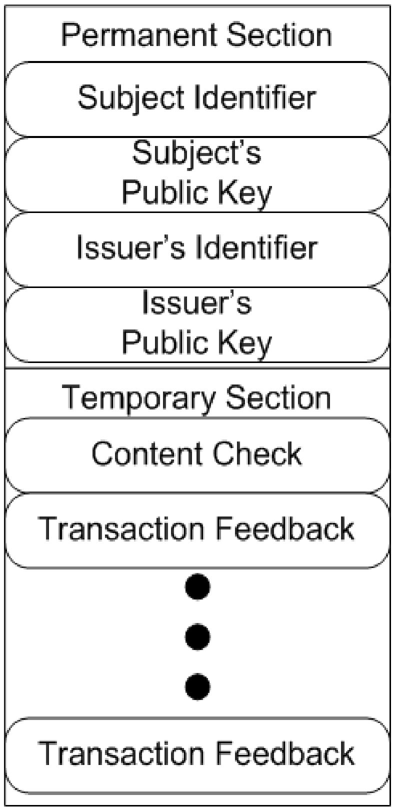


Figure 3-9 Format of an Atomic Behavior Record

The TS created ABRs for associates as a result of the introduction process. Once the ABR was established, the TS accepted reports and observations as inputs, finding the appropriate ABR and storing the information. The TS provided the ABR to the RSM when requested, applying the memory management algorithm as it restored the ABR to the store.

The ABR was used as the trust credential within the network. Users carried their own ABRs, as well as those of their associates, as a means of eliminating the problem of credential discovery in distributed networks (Freudenthal, Pesin et al. 2002; Dewan and Dasgupta 2004).

The TS contained the identities and the Reputation Indexing Windows (RIWs) for each of a node's TPs. The RIWs consist of collections of FIs that were organized by subject node and context. The TS also carried some *recognition evidence* for past TPs. The TMS needed to remember good and bad experiences from the past so that it would not waste time or place the user in danger while getting re-acquainted with former (possibly malicious) peers. Given this requirement, a node kept the identities and RIWs concerning former peers; those that have either moved out of range while on good terms or whose association was dissolved as the result of a complaint.

The permanent section of the ABR established the link to the user's identity certificate, shown in Figure 3-9. The identity section was derived from the Distinguished Name contained in X.509 v3-type certificates, issued by the KMS as described in Section 3.1.4.1. Although it was expected that a user entered and left ad-hoc networks based on their own goals and objectives, their identity would not change. Because of the permanence of the user identity, the TMS could track a user's behavior over a period of time. Recording and storing a user's behavior allowed other users to analyze past performance as a means to predict future behavior (Adams, Hadjichristofi et al. 2005).

The ABR's temporary section contained the evidence of the user's past behavior. Every time a user participated in a transaction (e.g., buying, selling, exchanging files), they received feedback on their performance. Feedback was a common feature of ecommerce sites and provided observations on a user's behavior that were used by peers to judge trustworthiness before engaging in a transaction. The number of transaction records stored within the temporary section varied with the user's activity within the network, as the network's reputation calculating mechanism read these records and determined the user's trustworthiness. Because the reputation calculating mechanism only read a certain number of transactions, there was no utility in keeping transactions above and beyond the required amount but there was no system-wide policy against doing so. Nodes discarded the unnecessary records, preferring to economize memory consumption through properly formatted ABRs.

The TS was more than a storage location. Information stored in the trust store was used in three procedures: recognition, introduction, and context tuning. Certificates in the trust store facilitated *recognition* by remembering former peers and reactivating their reputations based on verification of their identities. When a node came into contact with another node, it checked its TS to see if it had ever had an association with the new node. Because the node maintained a memory, it made a decision on whether or not to pursue an association with the new node based on information in the TS (Seigneur, Farrell et al. 2003). This action was a form of passive recognition; passive because another process triggered the recognition procedure. Nodes also had the ability to "self-trigger" the recognition process, as in the SECURE project (Seigneur, Farrell et al. 2003). In other words, a node might actively search for nodes that it *recognized* by monitoring communications and comparing the identities of communicators against its TS.

The TS was also crucial in the *introduction* procedure, a critical element in the integration of new nodes into the network and is shown in Figure 3-10. This introduction included a mechanism for the two nodes to share observed behavior history in such a way that they could derive the reputation of their prospective partner by having the proof to substantiate the given value. As a result, the TMS kept a certain number of its behavior observations so that it could provide non-reputable, references to other nodes when requested. A node confirmed the new peer's identity by verifying its certificate. It then filled its reputation windows with the FI received from its TPs and calculated the new associate's reputation. Finally, the node decided to establish an association with the new peer by comparing the new peer's RI against its trust thresholds.

A user took advantage of the contents of the ABR in the introduction process by soliciting a prospective associate for their ABR before initiating the transaction. The introduction process was iterative and halted whenever the user was satisfied of the peer's identity and had accumulated enough FI to fill their RIW. A user (e.g., Alice) began the introduction process, illustrated in Figure 3-10, by attempting to contact the peer's (e.g., Bob's) home domain to verify his identity. If Bob's home domain was not available, usually for reasons of network partitioning, Alice examined his ABR and attempted to find mutual TPs. These TPs were users that both Alice and Bob had chosen to trust in the past. If there are no mutual TPs or none could be reached, Alice had to make a decision based on the evidence within Bob's ABR on whether or not to trust him. It should be noted that relying on Bob to vouch for himself was the least desirable option and Alice would involve the risk assessment and situational trust elements of the trust management system before taking this decision.

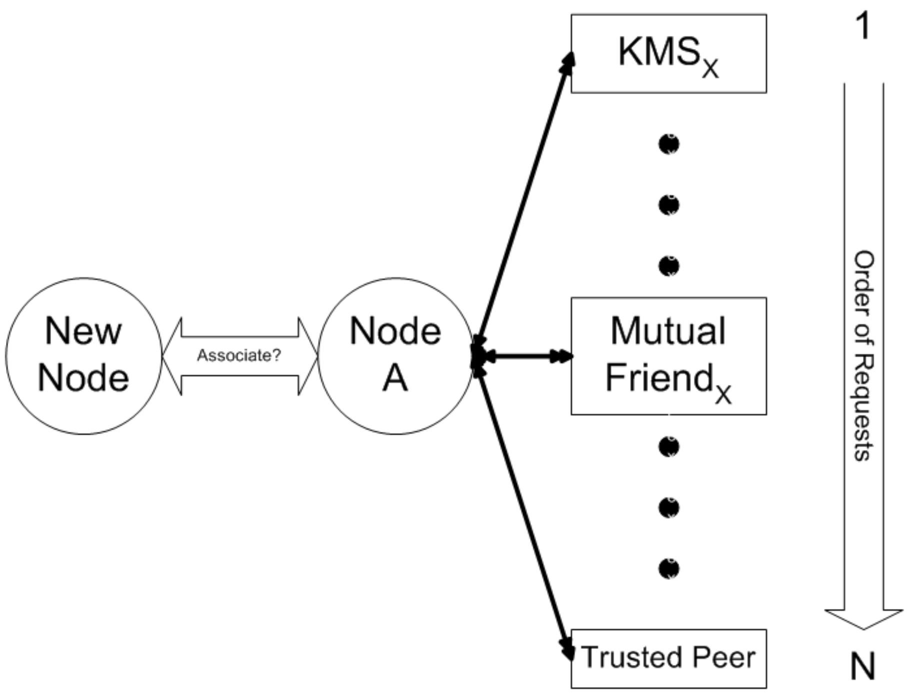


Figure 3-10 Introduction Process

The benefit of the introduction process was that Alice could be assured of Bob's identity and behavior through communication with trusted parties. In other words, she asked someone that she had already chosen to trust to recommend or refer Bob as a trustworthy user. Once someone verified that Bob was who he says he was, Alice used the transaction records from Bob's ABR to calculate his reputation. If his reputation was above her trust threshold, Alice trusted him; otherwise she recorded the incident and moved on in search of a more trustworthy partner.

The costs of the introductory process were in the time it took Alice to attempt to contact reputable users to provide references for Bob. Once Bob had been vouched for, Alice evaluated the contents of his ABR and calculated his reputation. In all, this was a necessary process but was one that should be performed as few times as possible.

Operating the TS encompassed more than just managing memory space. Identities and RIWs needed to be stored and organized for efficient searching. Most importantly, the TS needed to be protected against the possibility of compromise or unauthorized access during an operation. Protection was built into the design of the certificates and feedback items themselves, as they were digitally signed. Unauthorized insertion or deletion of certificates and associated RIW would cause a node to have to "re-introduce" itself unnecessarily. This caused undue network traffic and forced a node to wait while the re-introduction took place.

As a node collected identities, reports, and behavior feedback items, these items were stored in ABRs and placed in the TS. Based upon the assumption that the amount of memory that a node could apportion to the trust store was bounded in some way, this store had to be managed efficiently (Gray, Seigneur et al. 2003). When the bound was reached, the node selectively eliminated or "forgot" items to make space for new objects. Efficiency, in this case, was defined as limiting the number of "re-introductions" that a node required. Re-introductions were cases where a node forgot a peer that it once had security associations with and had to go through the entire introduction process, as if the two nodes had never dealt with each other.

The presence of the TS enabled a user to manage the behavior histories of its associates. The TS also provided a source for the credential exchange process called introduction. While the TS was closely linked to the formulation of reputations and associations, the results of TS's work had to be compared against the system's risk assessment before reputations could yield a trust decision.

3.2.3 The Reputation Scaling Module

"A good reputation is more valuable than money."

Publilius Syrus

The RSM was the heart of the TMS. The RSM was responsible for accepting an associate's behavior history and calculating a RI. The RI was a weighted estimation of the associate's trustworthiness and formed the basis for the TMS's trust decision. Without the RSM, the TMS would not have had the ability to process an associate's behavior history and, thus, have been unable to make any sort of trustworthiness decision.

The remainder of this section explains the basic trust model that the RSM implemented. The section then describes the method of how behavior observations were transformed into weighted FIs. The subsequent section explains the process by which FIs were "aged" through a series of weighted windows in the 3Win algorithm. Finally, the RI is defined in terms of a quantitative assessment of an associate's trustworthiness.

The RSM was the processing module that processed feedback to calculate a usable RI for its peers. The TMS implemented an Interpersonal Trust model (Abdul-Rahman and Hailes 2000) to represent the reputations that were compiled by a node on each of its peers. This trust type was node specific, so that the trust of one node to another was associative and not transitive. The following list summarizes the properties of the system's trust model:

a. Trust was independent, subjective, and unidirectional, such that different nodes calculated different reputation values for the same observed node.
b. Trust had positive and negative degrees of trustworthiness. Trust was expressed in continuous values that represented the range of reputation between untrustworthy (negative reputation) and trustworthy (positive reputation), as in Marsh (Marsh 1994).
c. Trust was based on experiences and observations between individuals.
d. Trust information was exchanged between nodes as performance observations and reports, as described below.
e. Trust was dynamic and was modified, in a positive or negative direction, based on new observations and reports.

The design of the RSM required an examination of other reputation management systems, as described in Section 3.1.3. The requirements analysis indicated that the TMS needed to exchange and process evidence of behavior to produce a usable RI. The importance of the evidence was to justify how a node arrived at a reputation value for its peer. The same evidence was also used to provide non-reputable behavior history referrals to peers as they encountered nodes they met in the network and wished to assess the trustworthiness.

The system denoted the reputation Alice maintained of Bob as $RI_A(B)$. The *RI* was represented by number values in the range [-1, 1]. This value represented the trust Alice placed in Bob; the higher the number, the more trust was imparted. This number was never stored. It was always recalculated with the latest information. Peers viewed a user with a reputation of –1 as completely untrustworthy. Peers viewed a user with a reputation of +1 as completely trustworthy. The notion of "completely" in terms of trust was a theoretical limit; in practice the only peer that was completely trusted was the KMS.

Peers viewed users that they had no information about with a reputation value of 0. Liu and Issarny (Liu and Issarny 2004) pointed out that this assignment made no attempt to differentiate between newcomers, strangers, users that had not participated (freeloaders), or users whose reputations had been calculated to be 0. Zero was considered a neutral value as it gave a new user a basic reputation to start with while limiting the impact strangers, freeloaders, and active users with low reputations had on the network. Referring back to Section 2.2.4, an *RI* of 0 implied that the user needed more information before making a decision, rather than implying trust or distrust.

Although the RSM used behavior grading from a variety of sources, the TMS required the RSM to differentiate between reports and observations. The former were provided by the KMS as described in Section 0 and indicated an action taken to establish or dissolve a trust relationship. Observations dealt with the performance of TPs and were a function of the network monitor that supports the reputation management system. A node observed the performance of its TPs and Friends. It periodically generated positive or negative FIs and passed them to its TPs. The generation of a negative observation did not imply immediate dissolution of the association.

The TMS implemented a system of dynamic weighting to apply the observer's reputation to the observation as part of the associative nature of the trust evaluation. If Yvette (user Y) observed Xavier (user X) and reported his behavior to Alice (user A), the system worked as follows. The performance observation (*$obs_y(x)$*) was weighted using the observing node's reputation (*$RI_A(Y)$*) before being integrated into the reputation calculation as feedback, shown in Equation 2. As explained above, *$RI_A(Y)$* was the recipient's (in this case, Alice's) current RI value for the observer (who was Yvette.) The reporting node's current reputation was applied to the observation each time the FI was used. This method allowed the reputation-scaling method to consider the changes in observers' reputation values during the calculation of the RI.

$$FI = RI_A(Y) * obs_y(x) \tag{2}$$

Once the reports and observations had been gathered, they were merged and processed to provide a meaningful value that a node could use in making its trustworthiness evaluation. The reputation value needed to give a lagging or conservative approximation of the feedback input; trust with a healthy dose of skepticism, as it were. The system wanted to emphasize current behavior while, as discussed in previous sections, aging older input to diminish its impact on the reputation calculation. As in CORE (Michiardi and Molva 2002a) and CONFIDANT (Buchegger and Le Boudec 2002b), a node maintained a reputation value for each peer that it associated with. Nodes entered the network with a reputation value of 0, giving new nodes a neutral level of trust. Similar to other reputation mechanisms, the expectation was that a node would naturally desire to have a positive reputation (Resnick, Kuwabara et al. 2000) and any node with a negative reputation would be isolated as nodes refused to interact with it (Michiardi and Molva 2002a).

Using the previous example, we can explain how Equation 2 worked. We begin by assuming that Alice calculated Yvette's current RI (e.g., *$RI_A(Y)$*)to be 0.65 previously. When Yvette shared a positive behavior grade on Xavier, Alice modified the behavior grade by Yvette's reputation so that the value of the FI was actually 0.65 (i.e., FI = 0.65 * 1). Because every node started with RI = 0, it was imperative that the first introduction be performed by the KMS, as discussed in Section 4.3.

This research developed the concept of a three-window weighted average (3Win). Modeled after a method of removing transients using batches or sub-samples (Jain 1991), this method divided a node's history into three weighted performance windows that revealed tendencies in a node's behavior. These windows were named Reputation Indexing Windows (RIWs) and were numbered one through three, with RIW_1 containing the newest FIs and RIW_3 holding the oldest FIs. FIs were "pushed" through the windows (i.e., from RIW_1 to RIW_2 to RIW_3) as new FIs arrived. When an FI was pushed out of RIW_3 it was discarded.

When a node began collecting FI, a simple, un-weighted average was used until RIW_1 was filled. Once RIW_2 was started, the FIs were averaged within each RIW and the weights specified in Equation 3a applied. The weights changed when RIW_3 was established so that the RSM used Equation 3b.

The reporting rate of the nodes established the window sizes. The window size was directly related to the frequency of routine performance observations. Longer intervals between reports or fewer network nodes resulted in smaller sample sizes and therefore smaller window sizes, but this was not found to result in appreciable differences in reputation values. Heuristically, it was determined that the sum of the three window sizes should be 16% of the time period's total anticipated number of observations to produce a curve that approximated a node's behavior trend. The total window size was divided into a 10:30:60 ratio among the three windows (e.g., $RIW_1 : RIW_2 : RIW_3$). This ratio highlighted the most recent input (a small sample with the heaviest weight) and enabled a memory of past behavior (Marsh 1994) with two larger but less weighted windows.

Nodes entered the network with a neutral reputation and started processing behavior feedback using Equation 3a, while RIW_1 and RIW_2 filled with FI. When the third window was established, the weights for each window shifted to those shown in Equation 3b. The second window was more heavily weighted than the third to emphasize the more recent input.

$$\text{For } 1< |FI| \leq (|RIW_1|+|RIW_2|),\ \lambda = 0.66;\ \mu = 0.33); \\ RI = (\lambda * RIW_1) + (\mu * RIW_2) \tag{3a}$$

$$\text{For } |FI| \geq (|RIW_1|+|RIW_2|),\ \lambda = 0.66;\ \mu = 0.22;\ \nu = 0.11 \\ RI = (\lambda * RIW_1) + (\mu * RIW_2) + (\nu * RIW_3) \tag{3b}$$

The evaluation of the 3Win began with a lambda value of 0.66, to stress the importance of the most current input. Experiments with the values of the weights of the three windows showed that a lambda value of 0.66 produced a curve that was responsive to the latest changes in input value trends. The RSM allowed a node to recover its reputation after having received negative input through continued positive performance. The rehabilitation process was the result of aging input items by shifting them through the windows. While the 3Win method enabled a node to overcome past mistakes, it yielded a skeptical approximation of the input. A hysteresis effect was desirable because it forced nodes to demonstrate sustained good behavior, rather than reward them too quickly and allow them to oscillate around the trust threshold. Furthermore, the widowed technique gave the advantage of being able to produce the non-reputable evidence (i.e., the FIs in the windows) upon which the reputation had been calculated. This meant that the system could implement both dynamic FI weighting and the introduction process in the trust management system.

Dynamic weighting was a powerful tool for the reputation-scaling algorithm as it had a significant impact on nodes that had been observed or associated with misbehaving nodes. Dynamic weighting relied on the existence of a system memory to store the identities and observations of its associates. This memory function and design are described in the following section.

3.2.4 The Risk Assessment Module

"Take calculated risks. That is quite different from being rash."
George S. Patton

Just as the RSM was an instantiation of Interpersonal Trust, the Risk Assessment module (RAM) implemented System Trust. The RAM monitored globally available information, in the form of reports from the KMS. These reports were aggregated to determine the general uncertainty in the network. This process gave a node the general impression of the network's "trust state." This was an expression of how risky an action was likely to be, given the current state of trust events in the network. Simplistically, this trust state can be phrased as "if other people are having success then I'm more likely to give it a try."

The original system design for the RAM had specified the use of an algorithm that constructed a decision surface (Jøsang and Presti 2004). The surface was calculated using probabilistic variables that expressed the expected gain and willingness to risk for an individual. These variables were set arbitrarily, instead of quantitatively or even heuristically. Because of this perceived flaw, Jøsang's method was not implemented in the proof of concept system.

The SECURE project (Dimmock, Bacon et al. 2005) provided the following definitions:

- *Risk* describes situations where one is unsure of the outcome but the odds of success or failure are known.
- *Uncertainty* applies to situations where one is unsure of the outcome and the odds are unknown.

Using this definition, the TMS estimated Uncertainty instead of Risk, although the remainder of this document will use the terms interchangeably. The decision to use an uncertainty estimate to adjust a node's trust thresholds was a recognition of the arbitrariness of attempting to define the world in terms of cost-benefit or payoff rather than in view of expected success or failure.

Uncertainty was calculated using global trust information from the KMS and first hand observations. In other words, each user approximated the risk of his local network by listening to the sources that he trusted implicitly. By monitoring the trend in reports and complaints, the user adjusted his trust and distrust thresholds to protect himself.

Because risk trends changed slower than reputation, the Global Risk Index (*GRI*) was calculated using a third order filter, shown in Equation 4. The third order filter was selected over other methods through testing. First order filters were too reactive to changes in environmental conditions, as discussed in our previous work (Adams, Hadjichristofi et al. 2005). Second order filters were better but still failed to provide the smooth trend line that described sociological conditioning to a risky environment. Equation 4 applied weights to the current FI and previous inputs before computing the new *GRI* (Jain 1991).

$$\begin{aligned} &\alpha = 0.35; \beta = 0.25; \gamma = 0.25; \delta = 0.15 \\ &GRI_t = (\alpha * FI_1) + (\beta * FI_2) + (\gamma * FI_3) + (\delta * GRI_{t-1}) \end{aligned} \tag{4}$$

This newly calculated *GRI* was then used to calculate the relative change in trust in the local network, shown in Equation 5. Here the default distrust threshold (T_D) was modified by the current *GRI* to arrive at an operational distrust threshold (τ_D).

$$\tau_D = \mathrm{T}_D - (GRI_t * \mathrm{T}_D) \tag{5}$$

This equation had the effect of increasing the distrust threshold (along the y-axis of the threshold graph) but not allowing the threshold to relax or decrease. The decision was taken to avoid adjusting the trust threshold, to prevent self-isolation. In essence, the risk assessment carries a grudge – even if the user moves to a safer neighborhood.

3.2.5 Trust Thresholds

Once a node selected a prospective peer, calculated that peer's reputation, and made a general assessment of risk, it needed to combine these into an evaluation to produce a trust or, in the case of a resource provider, an access control decision. This decision was accomplished by comparing the reputation to the risk-adjusted trust threshold.

Marsh (Marsh 1994) expressed the "cooperative threshold" in terms of expectations. A user calculated a threshold for each associate based on the perceived risk of the transaction and the perception of the associate's competence to complete the transaction. These context-sensitive perceptions are then tempered with the overall experience with that user to produce a threshold, which was applied for that specific transaction.

This approach displayed the common flaw of relying on expectation. Although expectations could be used in a logical or behavioral system, an individual's expectations were impossible to use in a quantitative system. Marsh's work (and those of the researchers who followed him) also failed to discuss a user's thresholds when they joined the network. Researchers implemented an initial state and analyzed how prevailing trust conditions acted to adjust these settings. Setting expectations remained a heuristic process and was unsuitable for nodes interacting with new and unfamiliar associates.

In the TMS, every user had two trust thresholds, as shown in Figure 3-11. Since reputations were expressed in values in the range of [-1,1], a user evaluated the reputations of prospective peers against these thresholds. One was a positive threshold, above which a user extended trust. The other was a negative threshold, below which a user withdrew trust. Between these two thresholds was a continuous range of trustworthiness. Reputation values varied within this "trust zone" based on interactions and environmental conditions.

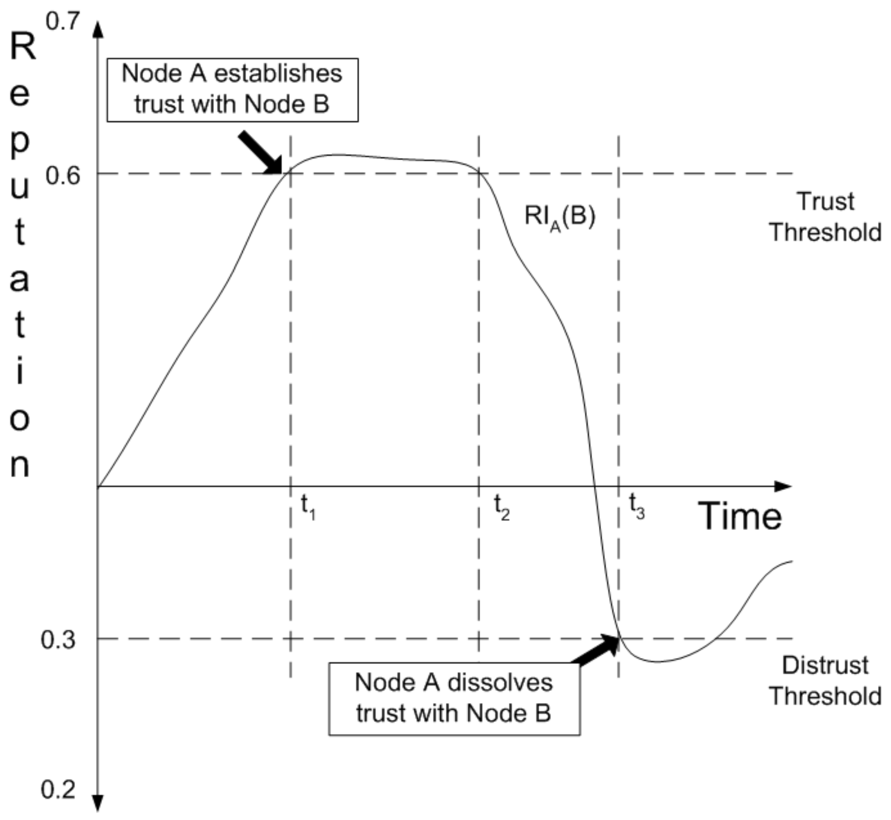


Figure 3-11 Examples of Trust Threshold Use

In this research, therefore, the TMS made initial estimates based on the trust profiles discussed in Section 0. These estimates were quantified as shown in Table 3-1. It was recognized that these estimates needed to allow a new user to meet and trust associates while, at the same time, provide a reasonable level of protection. Once a user had began to interact with other network members, the TMS adjusted the initial threshold values to protect the user. Because the TMS proved to be responsive to network conditions, the reliance on heuristically set initial values was not deemed a significant risk.

A user established a trust threshold based on the selected trust profile, a shown in Table 3-1. Given that the system explicitly revoked the identity of any user with an RI =-1 and that users joined the network with an RI = 0, the following provide a practical example to the use of trust thresholds in reputation-based systems. An altruistic user trusted a peer with an RI >-1, enabling any user that is allowed in the network the ability to establish trust with it. A forgiving user trusted any peer with an $RI \geq 0.6$, facilitating connectivity with new users and any existing user that had demonstrated desirable performance. Cynical users required a peer to have an

$RI \geq 0.7$, indicating that a user had a demonstrated positive behavior history. New users or users working to rehabilitate their reputations were not extended trust by cynical users. Note that while the definition of the Cynical trust profile stated that a Cynical user would not allow rehabilitation; in reality the system allowed a user to "start again" once their reputation crossed the X-axis in the positive direction. Distrusting users discounted trust and reputation, choosing to rely on MAC rules to allow access.

3.3 Simulation Design

Designing the TMS was only the first step on the road to establishing the TMS as a viable access control mechanism. Two additional components were required in order to test the TMS. The first component was a behavior model that would provide the simulated reports and observations on the performance of DCE members. The second component was a set of policies that the TMS used to protect itself from abuses commonly used against reputation-based systems. The combination of these two items allowed us to examine the response of the TMS to a variety of simulated network and behavior conditions.

3.3.1 Behavior Modeling

The first step to simulating the TMS was to create a model of how network members behaved. The model needed to first determine what the member's actual behavior was. Members were divided into three broad behavior types: Good, Selfish, and Bad. Because the system operated on the observation and reporting of behavior, the simulation then had to determine how that behavior would be reported based on the observer's behavior type.

Given that people's basic inclination was to exhibit stable behavioral patterns, a finite state machine was constructed to model a member's behavior. In our model we emphasized the states rather than the transitions or stimuli, as our interest lay in the behavior exhibited during the state rather than in the significance of the state transition. Users started their behavior model in an initial or "natural" state. Behavior was based on the state that a member occupied at the time of making the observation. This model, shown in Figure 3-12, emphasized the probability that a member would change their behavior. Stable behavior types, such as the Good or Bad type, would start in their natural state demonstrating their dominant behavior. Selfish behavior types would start in a randomly selected state with positive behavior. The probability that a member would change states is given in Table 3-2.

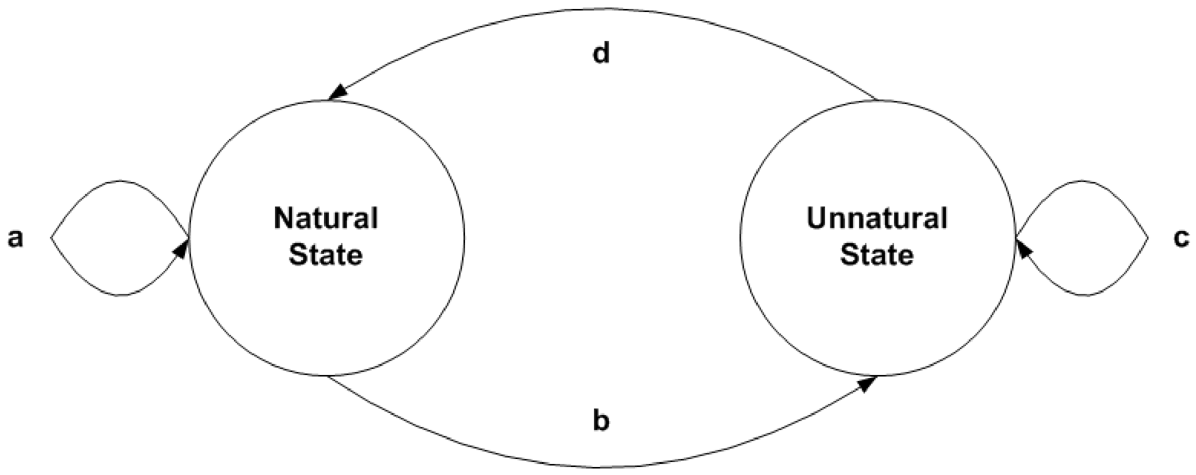


Figure 3-12 Finite State Machine Behavior Model

The state change probabilities (b and c) were selected to provide variance in individual behavior without creating random, irrational activity (Laird and Wray 2003). By moving the state change probabilities, we predicted the TSM's actions. The system approached a trivial state of complete predictability as these probabilities approached 0. Setting the probabilities closer to 50% to encourage state change reflected random activity that did not model human behavior. The settings in Table 3-2 were selected to provide the network with a stable population of members that demonstrated willingness to join a collaborative enterprise.

Table 3-2 Behavior Model State Change Probabilities

Behavior Type	Initial State	a	b	c	d
Good	Good	90	10	10	90
Selfish	Random	70	30	30	70
Bad	Bad	90	10	10	90

In this case, the term stable implied that there was a low standard deviation in reputation indices. Users whose reputation index displayed a high standard deviation were displaying unrealistic behavior. The goal was for “good” users to show positive behavior the majority of the time. Figure 3-13 shows the impact of changing the settings on the average reputation index value of good members. The left side of the graph demonstrates that a stable, good user had an RI suitable for extending trust. As the parameters were changed to introduce more erratic behavior, good users changed states too frequently to demonstrate sustained positive behavior and thus earn trust.

The next part of the behavior script used the same behavior model to predict how the observed behavior would be reported. Table 3-3 shows how the cross mapping of behavior types to current behavior states yielded observations. In most cases, the observer reported the behavior truthfully but, in the cases of a Bad user, there was the chance that the report would be misleading. During testing, an element of collusion was added by mandating that the Bad user always report positive behavior for its confederates but followed the table for all other observations.

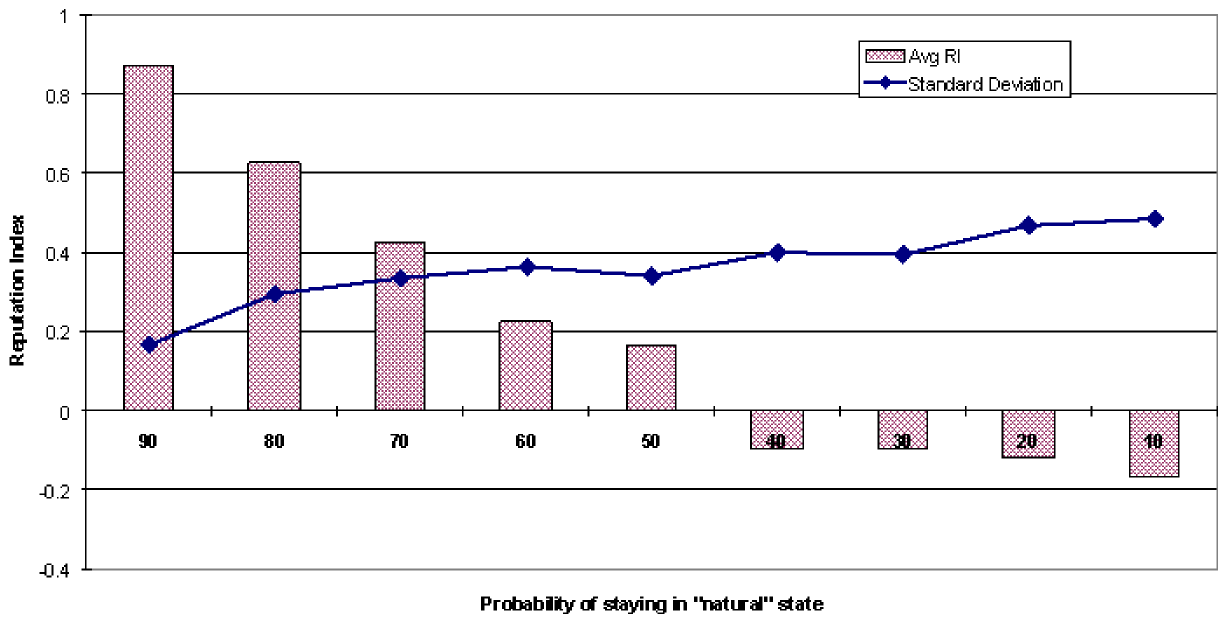


Figure 3-13 Analysis of Behavior Model Settings

Table 3-3 Behavior Reporting Based on Behavior Type and State

If a	Is in its	It will report
Good or Selfish user	Natural state	The behavior as observed
	Unnatural state	The opposite of what was observed
Bad user	Natural state	The opposite of what was observed
	Unnatural state	The behavior as observed

3.3.2 Gaming the System

The second step in simulating the TMS was to construct a set of operational procedures that would allow the system to eliminate or diminish the impact of common abuses of a trust-based system. A commonly perceived flaw in reputation systems was the ability for a user to "game" or manipulate their reputation to gain an unfair advantage. Gaming reputations undermined trust in the overall system. Users that knew their reputation often went to great lengths to maintain a high reputation, for both legal and illegal purposes. In KARMA (Krishnan, Smith et al. 2003), nodal entities successfully gamed the system to raise their overall KARMA rating to achieve greater access levels on a virtual bulleting board. EBay users inflated their client/seller ratings, which attempted to demonstrate accumulated trustworthiness based on transaction success, through a number of schemes like stacking (Keser 2003).

This research used a Soft Security concept termed *Dissuade Reputation* (Shah 2005). This reputation system sought to isolate rather than exile misbehaving community members. Dissuade Reputation was used extensively in online communities such as Slashdot (Commander_Taco 2005) and Kuro5hin (Kuro5hin 2005). As with our TMS, "it's possible to get an elevated K5 status (a term for reputation) but it takes continued effort to keep it." (Shah 2005)

In an article on online reputation, Derek Powazek (Powazek 2003) maintained that users only cared about their karma score if they knew what it was. This assertion was also stated by Ben Salem (Ben Salem, Hubaux et al. 2004), who pointed out the difficulty in manipulating an unknown quantity. With any score or scoring system, there was the danger that community members would compete to get the highest score, manipulating what should have been a control into a game. If the game (i.e., gaining karma) was too easy or predictable, the control aspect of karma lost its meaning. This phenomenon was evident on Slashdot, with its use of karma points to gauge user activity and site privileges. As Slashdot users raced to increase their karma, the administrators made the decision to cap the amount of karma points any individual could accumulate. Rather than dissuade gaming, capping karma created a new type of gamer called a "karma whore." These individuals strove to reach the karma cap as quickly as possible, then dissipating their karma through malicious or disruptive behavior before starting all over again.

Another popular online community is Kuro5hin. This site concealed the user's mojo score (its name for reputation). Users gained a perception of their standing only by observing their privilege levels. Concealing reputation eliminated the gaming aspect but users may become frustrated and disruptive if the system prevents them from accessing resources.

Because an individual's reputation varied amongst his peers, direct "one on one" reputation gaming as is exhibited on eBay or Amazon.com was impossible in our TMS. In our TMS, every user (e.g., Alice) maintained an individual reputation index for each associated peer (e.g., Bob). This reputation index ($RI_A(B)$) was never shared as a single value. Instead, when Alice was asked to recommend Bob, she provided evidence in the form of non-reputable behavior grades. The requester was then left to calculate their own reputation index for Bob. Bob never knew what Alice thought of him, only that she granted or denied him access to her resources.

3.4 System Metrics

The TMS was an access control system. We determined that we could assess its efficiency by examining how often the system correctly allowed access. The cost of making the decisions, in terms of communications and storage overhead, was also included in the calculation. While acknowledging that the success metric of an access control system was comparative (i.e., one system performs better than another given a set of circumstances), we also experimented with critical settings to determine a feasible parameter range within which the system was effective.

In the most basic sense, the system was efficient when it correctly allowed access more often than it made incorrect decisions. Incorrect decisions came in two forms. *False positive* decisions occurred when a trustworthy user was incorrectly denied access. *False negative* decisions occurred when untrustworthy users were incorrectly allowed access (Bryce, Dimmock et al. 2005).

We examined the ratio R of correct answers to false negative and false positive answers, shown in Equation 6a. D was the total number of trustworthiness decisions the TMS was asked to make. P was the number of false positive answers and N was the number of false negative answers.

$$R = (D - (P + N))/ D \tag{6a}$$

When tabulating access control decisions, we differentiated between false positives and false negatives, as Bryce did (Bryce, Dimmock et al. 2005). We did this differentiation in recognition of the fact that the cost of a false positive was much less than the cost of a false negative. The cost weight (ω) was a value selected to represent this difference in cost and added to the previous equation. The result is shown in Equation 6b.

$$R = (D - (P + \omega N))/ D \tag{6b}$$

Having examined the efficiency of the TMS, we evaluated the overhead required by the system to render its decisions. The general intent of the overhead metric (C) was to determine the cost of the level of efficiency. Two forms of overhead were included in the calculation of C.

Communications Overhead (C_C) was defined as the number of FI that needed to be sent between trusted peers to gain enough information to determine a trustworthiness decision on a specific peer. Equation 7a illustrates how the system divided the number of Introduction transactions (I) by the RIW size. This computation assumed that the user would, in the worst case, attempt to fill their RIW before calculating a new associate's RI. This assumption is not as far-fetched as it may seem, especially if the number of reports was few.

$$C_C = I * |RIW| \tag{7a}$$

Storage Overhead (C_S) was defined as the number of FI each node stored to create a decision. Equation 7b determined C_S by multiplying the TS size (expressed in the number of stored ABRs) by the RIW size.

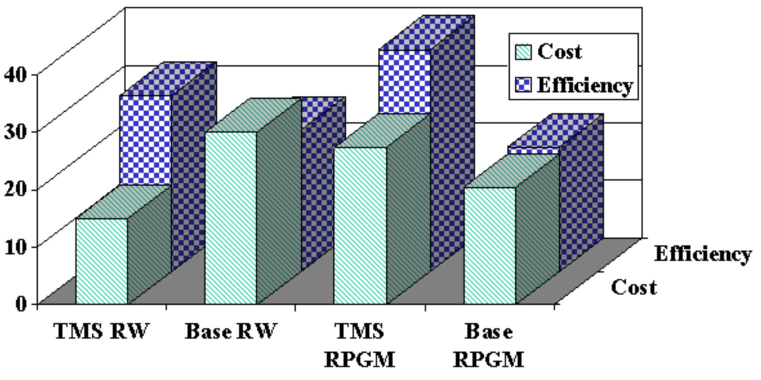


Figure 3-14 Example Comparison of Scenario Efficiencies versus Costs

$$C_S = |TS| * |RIW| \tag{7b}$$

Adding the two costs together yielded the number of FIs maintained by the TMS over a period of time. Equation 7c used this result, divided by the number of correct access control decisions (*D* - *(P+N)*), to provide the total cost for each correct decision.

$$C = (C_C + C_S)/(D - (P+N)) \tag{7c}$$

When we developed the test scenarios, each scenario had independent values for *R* and *C*, as shown in the example in Figure 3-14. We called these values R_{Sx} and C_{Sx}, where *x* was the scenario number. In analyzing R_{Sx}, we wanted a value as high as possible. The opposite was true of C_{Sx}, where we wanted the smallest number possible. Because the number of different scenarios was bounded only by the limits of imagination, we used Lo Presti's format (Lo Presti, Butler et al. 2005) for devising and organizing test scenarios (see Appendix A).

3.5 Summary

Access control in a dynamic coalition was presented as a mixture of mandatory and discretionary access controls. Systems used role-based access controls as a shortcut to assigning a consolidated set of permissions and privileges to users. While this worked in centralized environments inside hierarchically structured organizations, it was not well suited for situations with little or no structure. In an RBAC environment, users from outside the organization were given temporary identities and roles within the organization. These roles had to be scoped to allow the new user to perform their task and were then revoked when the user returned to their home organization. The administration involved in this process was unworkable in an ad-hoc collaborative environment.

At the other end of the scale, peer to peer environments had little structure and no fixed identities. All transactions were conducted based on reputations or recommendations from other users. While this solution was far more distributed than the RBAC schemes, it assumed that users performed the same roles in the network. This assumption was valid for file sharing or sensor data routing but was not acceptable in a network where there was a requirement for any sort of structure.

The proposed access control system filled the gap between these two extremes. Users declared an identity, usually from within their home organization, and were assigned a coalition role by a DCA or through a peer election. Role holders executed their tasks while being observed by the network monitor and their peers. Observation reports were provided to a node's trusted peers on a regular basis. The observations were aggregated to produce a reputation for that trusted peer. A node, whether it was a user or a resource, used the combination of role and reputation to determine granting access.

The trust-based access control system, therefore, operated in a decentralized manner. It allowed "foreign" users access to organizational resources yet maintained a degree of structure to enable considerations such as information releasibility. Because the system was based on trust, it did not require implementation of role delegation or revocation. The system prohibited trust transitivity and implemented an introduction process to facilitate the reputation evaluation between two peers.

4 Analysis

Trust-based systems relied on a quantitative representation of trust (Bryce, Dimmock et al. 2005). This research provided a mechanism for aggregating behavior grading information, assessing uncertainty, and applying the resultant reputation value to create a trust-based access control (TBAC) system. This reputation is called "evidence-based" since the value was based on evidence of the peer's behavior. In effect, the reputation we calculated for a peer was the result of reviewing that peer's behavior history.

The innovative application of behavior grading and reputation in a trust-based system provided flexibility in an ad-hoc collaborative environment. We have asserted that "evidence-based" systems were better than role-based access controls (RBAC) because they made their decisions at runtime rather than from static, pre-configured information. Module and system testing were conducted to verify and validate this assertion.

This chapter presents the results and analysis of the Trust Management System (TMS) verification and validation testing. Building upon the system design presented in the previous chapter, Section 4.1 explains the importance and contribution of verification testing to the overall development of the proof of concept. Section 4.2 provides the results of the implementation of the trust store (TS). Similar analyses are presented for the reputation scaling module (RSM) and the risk assessment module (RAM) in sections 4.3 and 4.4, respectively. The summation of these three sections is the verification of the TMS modules' performance and operational boundaries. Section 4.5 presents the goals and objectives of the trust management system validation, describing the simulation environment in terms of what the testing was to achieve. Section 4.6 builds upon these results and discusses the TMS performance in terms of the system metrics developed in Section 3.4 before summarizing the analysis portion of this research in section 4.7.

4.1 Verification: Goals and Objectives

Verification determined that each module had correctly transformed its inputs into the expected output. This testing involved isolating each function of each module by the arrangement of input and processing parameters. Specific outputs were then analyzed to check their correspondence to expected results. In general, verification testing revealed that the modules worked in accordance with the requirements.

Once basic verification was complete, performance boundary analysis was conducted to ascertain under which conditions the module operated best and under which conditions performance was impaired. Once these expectations were met, the modules were combined and the system validated. The following sub-sections provide the analysis of verification testing. These sub-sections follow a standard methodology (Bryce, Dimmock et al. 2005) of:

1. Defining the role of each component.
2. Analyzing the component to determine how module failure or impaired performance influences overall system functioning.
3. Adding the points of failure to the system test profile used in validation.

4.2 Trust Store

The implementation of a system memory represented a step forward in state of the art access control because the TS was actively managed to update trust records on a continuous basis. Current systems fell into one of two categories: those with passive storage and those with session-based storage. Those with passive storage collected credentials for associates, usually in the form of authorization certificates. When the storage space was full, none of these systems had defined management schemes to eliminate unneeded credentials and make space for new certificates. Systems with session based storage kept values, like reputation and threshold values, in program memory. The trust or access control system was vulnerable while it learned about its associates. This vulnerability presented itself each and every time the system started and was considered grave enough to accept the storage and processing costs of implementing the TS.

The TS provided a finite memory for the TMS. The TS held the behavior history of peers and provided the system the ability to remember and incorporate the previous actions of associates into its access control decisions. The role and design of the TS was described in

Section 3.2.2. The TS stored the records of associates' behavior in the form of Atomic Behavior Records (ABRs) and managed the memory to limit the number of re-introductions the TMS performs. The TS also provided the store from which a node made recommendations and referrals to its trusted peers.

Without the TS, the system would be forced to work on a by-session basis, remembering associates only for the term of the current session. If the TS was too small, a user could be overwhelmed performing introductions and be kept from performing actual information sharing. If the TS size was too large, a user spent too much time searching the store for associates that might or might not be present.

A properly sized TS management algorithm:

- Should minimize the number of "re-introductions" that a node required.
- Should optimize storage of the trust and identity information to allow a node to remember as many peers as possible.

Testing was conducted in network environments that contained 100% "good" or behaving users. By removing the possibility of encountering undesirable associates, we presented a testing scenario where a user would want to interact with every node it came into contact with. The following sections examine the effects of changing various parameters on determining the optimum amount of memory allocated to a TS.

4.2.1 General Testing

General testing was performed to establish the sizing of the TS under different network population characteristics. We called the relationship between population and transmission range network density, as it characterized the number of peers within a node's transmission range.

The TS should be sized to prevent the need for reintroductions. Reintroductions represented work that had already been accomplished but had been deleted and needed to be performed again. At the same time, the TS needed to be small enough to prevent saving lots of people that we no longer interacted with. We wanted to link the size of the TS to the network density, rather than a "one size fits all" approach, so that we were optimizing for all densities rather than picking a size that suited for the densest networks and assuming the cost of excess memory usage when operating in a sparse network. Finally, we wanted a memory management technique to allow us to forget older and presumably unused associates' information.

4.2.1.1 Memory Store Management

The first set of tests sought to determine the memory management algorithm that minimized the number of introductions. The tests compared the number of introductions in static and random waypoint mobility (Camp, Boleng et al. 2002) situations, varying the size of the TS in successive runs. Two algorithms, First In First Out (FIFO) and Least Recently Used (LRU) were compared. These specific algorithms were used because they did not require additional timing information, used by algorithms that were based on usage frequency. This timing information was not available to the TMS, as all frequency information was lost when an associate was "forgotten."

The FIFO algorithm placed an associate's ABR at the head of the TS queue. As newer associates were encountered, older ABRs were pushed toward the end of the queue. When an ABR was pushed from the queue, that associate was "forgotten."

The LRU algorithm required more queue manipulation than FIFO. While new ABRs were added at the head of the queue, as in the FIFO scenario, LRU moved ABRs to the head of the queue each time an ABR was accessed. Other ABRs were pushed down to fill the space. ABRs that were pushed out of the queue were forgotten.

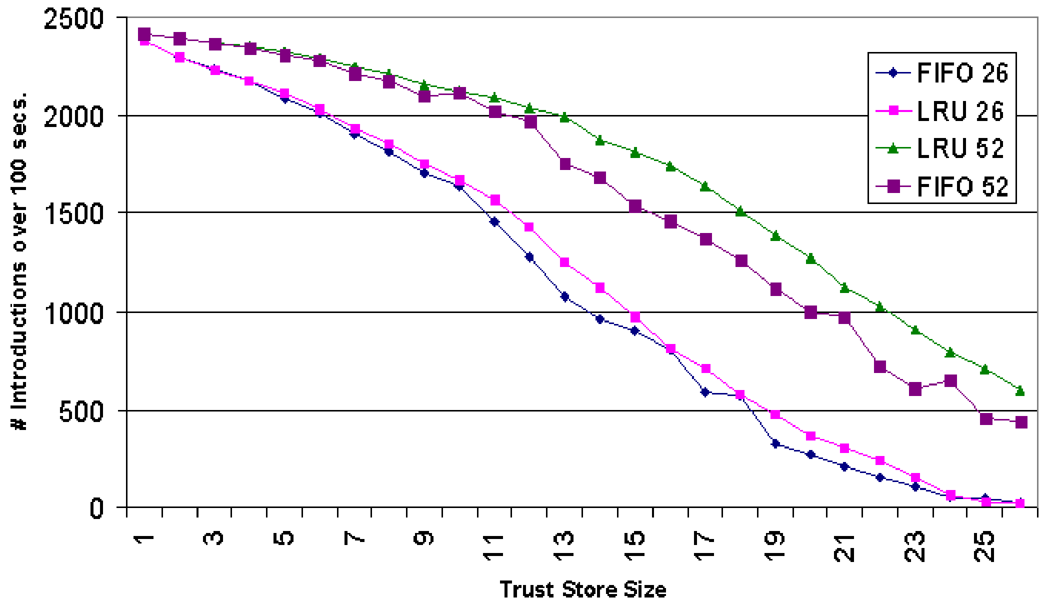


Figure 4-1 Trust Store Management Algorithm Comparison

Figure 4-1 shows the performance of the FIFO algorithm against the LRU algorithm in a mobile network of twenty six and fifty two users. Tests used different memory management algorithms on the same simulation script to determine the number of introductions a user needed to perform given a specific TS size. The goal was to determine which algorithm was most efficient at maintaining the ABRs of the associates the user needed to speak with.

In both cases, the FIFO algorithm required fewer introductions than the LRU implementation over the period of the simulation. As the TS size increased, the FIFO curve decreased faster than the LRU curve. Detailed analysis of the TS contents determined that FIFO worked better than LRU because of the node's mobility in the simulation. LRU focused on a node's current associates and forgot old associates right before the mobility model brought them back into range, thus requiring the node to reintroduce itself. Based on this testing, subsequent tests used only the FIFO memory management algorithm in the TS.

4.2.1.2 Trust Store Size Determination

Efficiency, in the case of the Trust Store management problem, was defined as limiting the number of "re-introductions" that a node required. Re-introductions were cases where a node forgot a peer that it once had security associations with and had to go through the entire introduction process, as if the two nodes had never dealt with each other before. Through testing, we wanted to find the smallest trust store that minimized the number of introductions the system had to perform.

Sizing tests performed using a random waypoint mobility simulation showed that the optimum size for a trust store was approximately 30% of the total number of users of the network (N), as shown in Figure 4-2. This optimum was determined by comparing the increasing amount of memory required for the ABRs against the number of introductions required in a network of a certain density. Sparse networks (e.g., those with less than 100 members per square kilometer) required a user to store a higher percentage of associates' credentials due to mobility. Once a certain density was reached (e.g., 100 mobile users in a 1000 x 1000 meter area), however, the tradeoff point stayed near the 30% mark. This meant that, if a user could know how many nodes were in the network, he could size their system memory to minimize reintroductions and maintain only those credentials that he needed in the near future.

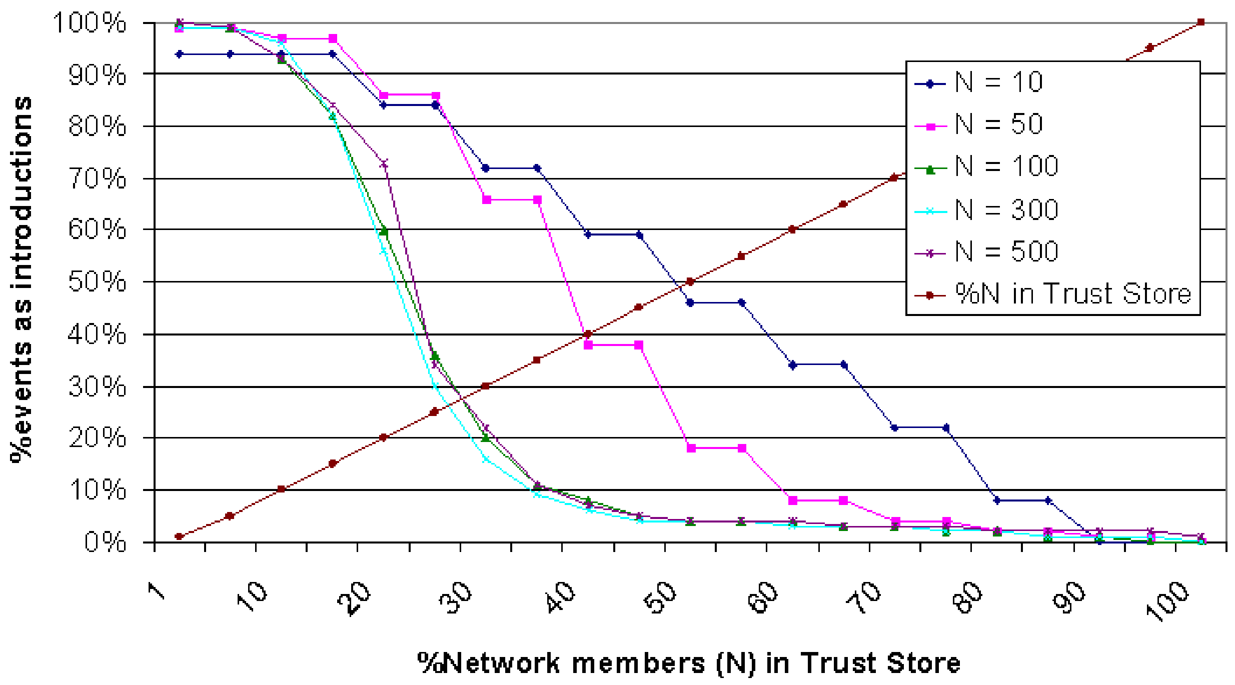


Figure 4-2 Trust Store Sizing Test

Since the ad-hoc nature of a DCE prevented anyone from knowing the number of users in the network at any time, we developed the concept of network density sampling, shown in Equation 9.

$$D = (N\pi r^2)/A \tag{9}$$

To use Equation 9, a node entered a network and listened for the beacons of its neighbors. In a hybrid environment (i.e., one having wired and wireless users), the node might sample the headers of IP packets. The user knew its *r* (the transmission radius) and assumed *A* (the area of the mobility area) to be a standard one square kilometer. As it listened, the node counted *N* (the number of nodes it could hear) and calculated *D*, the network density.

From this equation we created three representative network densities: sparse, medium and dense. In Figure 4-2, the sparse networks had ten members in a square kilometer, resulting in a *D* of 1. With this density, the node had to maintain at least 50% of the network members (i.e., five ABRs) in its memory to achieve the desired optimization of reintroductions. Similar tests showed that medium networks (50 members) had a *D* of 5 and needed to maintain 40% of the network members in the TS. Dense networks had a *D* of 10. Once Equation 9 indicated the presence of a dense network (i.e., a network with more than 100 users in a square kilometer), the user set the TS size to 30% of the number of members it sampled upon entering the network. During operation or at any point the node detected that it had to reintroduce itself more than indicated by the density factor, it re-sampled the nodes and reset the TS size.

4.2.2 Environmental Impacts

While it might seem feasible to accept the cost of a large TS, tests in a 100 user static network showed that larger stores had less efficiency than smaller, more optimally sized ones. Figure 4-3 shows how increasing the store size for a user with a 100 meter transmission range lowers the cost significantly but actually decreases the efficiency from 99% when the |TS| = 10 to 82% when the |TS| = 100. Since the actual cost of introductions in a static environment was minimal due to the increased storage and communications capacity of the nodes, there was little point in decreasing costs at the expense of losing almost 18% efficiency.

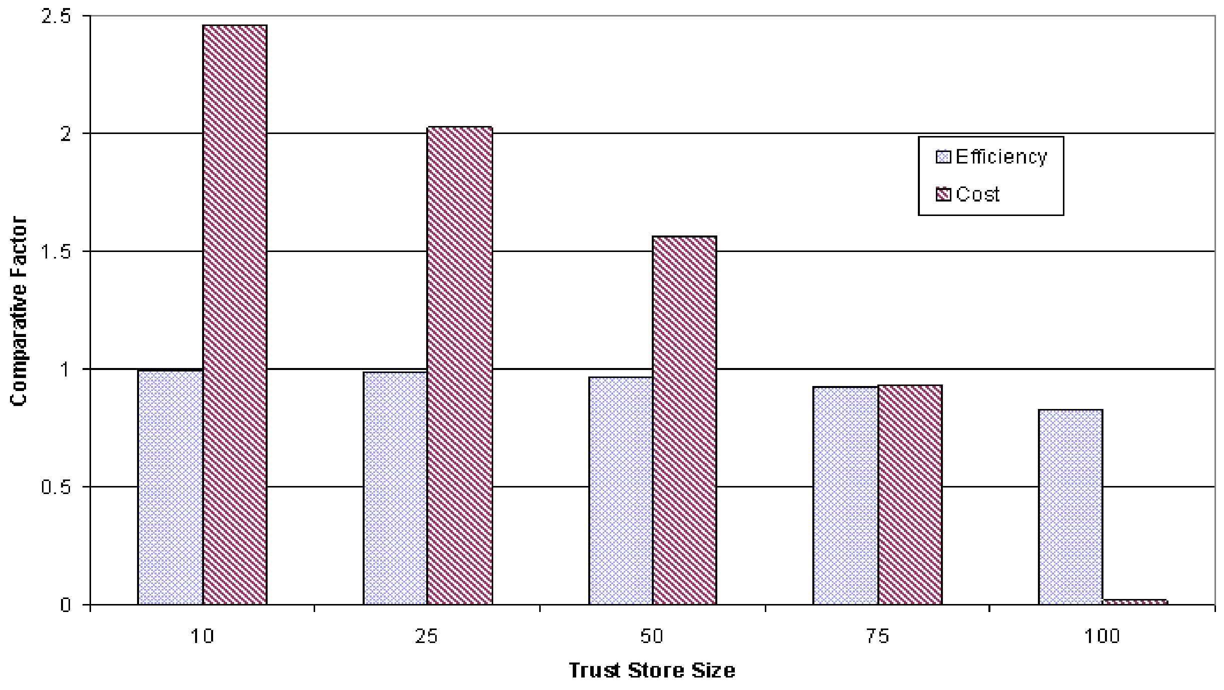


Figure 4-3 Effects of Trust Store Size on Efficiency and Cost

The decrease in efficiency seen with larger TS sizes was traced to the effects of the introduction process. While trying to minimize the number of introductions performed, testing proved that it was not beneficial to do away with them completely. The introduction process had the effect of flushing out the ABRs of old associates, in effect restoring them with more current information and improving the fidelity of the reputation and, therefore, the efficiency calculations it was required to perform.

The performance of the TS was directly proportional to the number of introductions. Network density and mobility affected the number of introductions as a result of increased interactivity. Since, as discussed above, changing the TS size had a large impact on the cost of interactivity, sizing the store to decrease costs had to be coupled with the acceptance of decrease in efficiency.

4.2.3 Contributions and Conclusion

The TS represented a number of contributions to the state of the art in trust and access control systems. The presence of the TS enabled the TMS to maintain and update the behavior history of a user's associates. Additionally, linking the concepts of network density to memory size optimized the storage cost of holding trust credentials when compared to un-managed or memory-less systems. The analysis presented in this section showed that the TMS minimized the number of credential exchanges it had to perform because the contents of the TS were actively managed. This performance increase was reflected in a lower communications cost incurred while operating the TMS.

While the TS provided a number of benefits as implemented, there remained room for improvement in certain situations. An unfortunate aspect of more mobile networks was the situation where a user forgot important associates or resource providers, since older ABRs were forced out of the TS by new acquaintances. This deficiency might be addressed through the use of a "passive" store that was user-selected and identity-based. The passive store would not be subject to the same memory management algorithm and would allow a peer to remember his supervisor and/or a set of resource providers without having to be reintroduced to them on a regular basis. The danger with the implementation of the passive store would be the dated information contained in these ABRs. The suggestion was made that ABRs stored in the passive store would contain only first hand and Key Management System (KMS) observations to minimize this deficiency.

The TS existed to provide storage for the behavior grades of network peers that a user wanted to associate with. Information, in the form of ABRs, was provided upon request to the reputation scaling process whenever the system needed to calculate an associate's trustworthiness. The process verification testing is discussed in detail in the following section.

4.3 Reputation Scaling

The Reputation Scaling module (RSM) was the heart of the TMS. The RSM's purpose was to implement a quantitative method for aggregating behavior feedback items (FIs) to generate a reputation value for each associate. A node used this value, called a Reputation Index (RI), as a measure of the trustworthiness he had of a specific network peer based on the peer's previous behavior. The RSM responded to the fluid nature of the Dynamic Collaborative Environment (DCE) by re-evaluating the source of each behavior grade before using the grade as input to the reputation scaling equation. The end result of this equation was a substantiated reputation index (RI) that was provided to the TMS. The RI was then compared against the trust thresholds to determine whether or not the system should extend trust and grant access to the requested resource.

If the RSM failed during operation, the TMS would have no way of processing behavior information or judging an associate's trustworthiness. The TMS would have to abandon a trust-based approach and resort to pre-configured access control methods, such as RBAC or identity-based mandatory access controls (MAC.) Neither method was considered acceptable in a DCE, for reasons discussed elsewhere in this research.

The RSM's approach had the following characteristics. It:

1. Was node-centric, so that each node calculated only the reputations of the peers it was concerned with;
2. Weighted FI to emphasize current behavior trends while accounting for past performance;
3. Merged behavior reports (first-hand experiences) with observations (second-hand remarks), as defined in Section 3.2.3;
4. Aged FI over time to remove outdated behavior information; and
5. Enabled nodes to recover their reputation by demonstrating desirable behavior.

Verification of the RSM required the reputation scaling equation to produce a RI that conservatively estimates the observed peer's actual behavior. Section 4.3.1 compares the RSM's performance to the actual behavior grade and commonly used estimation techniques, such as the exponential weighted moving average.

4.3.1 General Testing

The RSM was tested to verify that the RI output by the module displayed a hysterisis effect with respect to its input, as shown in Figure 4-4. As the recording node (e.g., Joe) moved through different network conditions, the RSM was expected to produce results that accurately reflected the original input as the recording node (e.g., Joe) moved through different network conditions. In testing the RSM, the 3Win method was compared against the original input and the exponential weighted moving average equation used by Buchegger (Buchegger and Le Boudec 2002b). Comparing 3Win to the actual input and an exponential weighted moving average (WMA), Figure 4-4 shows how the RSM produced an RI that lagged behind the changes in behavior, as desired.

In the subsequent tests, we wanted to investigate the RMS's response in mobile situations. Interactivity traces were constructed using MATLAB and a Random Waypoint model. A 100 node network was constructed inside a 1000 x 1000 meter area. Each simulation was run for 1000 seconds. Behavior profiles were applied to create simulation scripts, as described in Section 4.5.2.

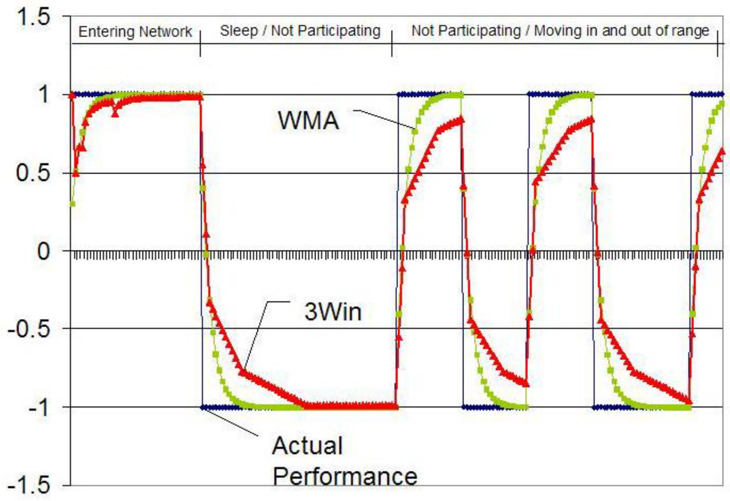


Figure 4-4 The Hysterisis Effect of the 3Win Method

Scenarios were designed to test the ability of a RSM to:

- Identify selfish behavior,
- Allow a node to rehabilitate its reputation following a period of poor connectivity, and
- Respond to nodes that try to denigrate other nodes with undeserved negative performance observations.

In addition to these three performance measures, testing also needed to gauge the impact of dynamic FI weighting. Dynamic FI weighting linked and maintained the identity of the reporting node with the observation. The reporting node's current reputation (i.e., the RI value at the time of the calculation) was applied to the observation each time the RI was calculated. This method allowed the reputation scaling method to consider the changes in observers' reputation values during the calculation of the RI.

In each test, a node (e.g., Joe) received FI generated from observations and formal reports. The data set represented a period of approximately one hour of operation in the test bed and was distributed in the interval [-1, 1] based on the previously described movement and behavior-based scenarios. Nodes provided observations on a 10 – 12 second interval. In all of the following graphs, the X-axis represents the passage of time and the Y-axis is the value of the node's RI.

The "unreliable node" scenario, shown in Figure 4-5, tested the RSM's flexibility in allowing a formerly unreliable node (e.g., Bob) to rehabilitate the reputation value that Joe maintained for him as Bob moved in and out of the Joe's transmission range. After a period of mobility, Bob relocated to a position with more stable connectivity and resumed cooperating with the network. Joe's associates observed and commented on Bob's behavior, providing the behavior grading that Joe fed to his RSM.

Because the periods of positive and negative observations were balanced, it was expected that the RSM would allow Bob to rebuild his reputation as he moved into the operating range of his peers. Figure 4-5 illustrates the performance of the two reputation scaling methods in the "unreliable node" scenario. Of note is the sharply fluctuating, optimistic curve that is produced by the WMA method. The 3Win mechanism produced an RI curve that was a smoother and more conservative approximation of the input while also allowing reputation rehabilitation.

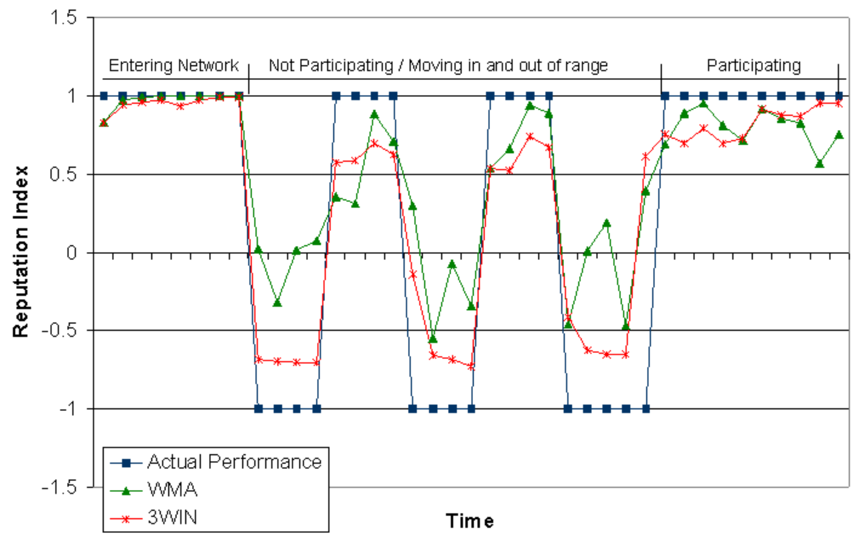


Figure 4-5 Performance of Reputation-Scaling Mechanisms in an "Unreliable Node" Scenario

Points where the 3Win curve departed drastically from the WMA showed the effects of dynamically weighting observations. Because of its lack of history, the WMA method could only weight the most current observation and then only at the time it was applied to the reputation calculation. The 3Win method reapplied the weights of the observers at each calculation. As the observers' reputations changed, the value of their recommendations (in the form of FIs), changed as well.

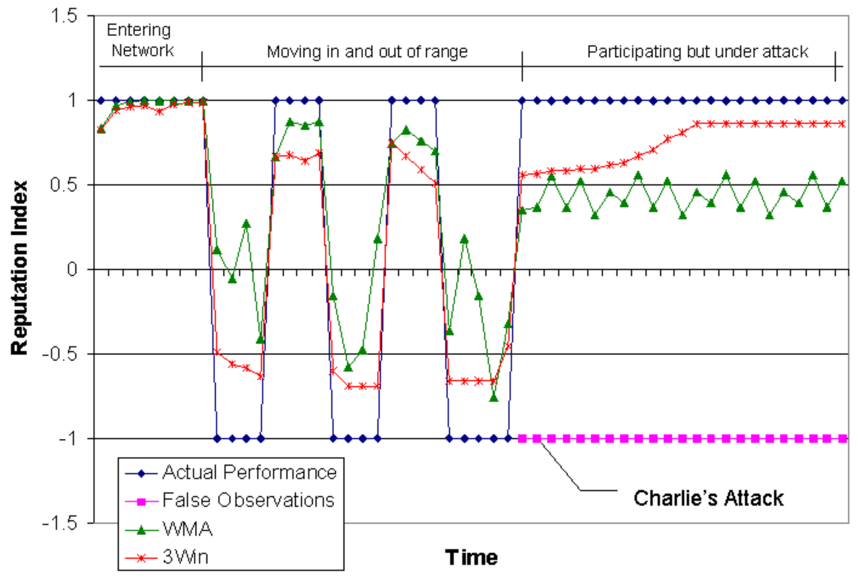


Figure 4-6 Performance of Reputation-Scaling Mechanisms in a "Smear" Scenario

The "smear" scenario tested the efficiency of the reputation management mechanisms at resisting attacks on a node's reputation (see Figure 4-6). The observed node (again we'll use Bob) moved into the observer's (e.g., Joe's) operational area and should have been receiving positive feedback but Charlie tried to smear Bob's reputation by maliciously reporting negative feedback. The results show that the reputation management methods resisted the attack by dynamically weighting the FI, effectively diminishing Charlie's ability to impact Bob's reputation. When Charlie (the smearing node) was determined to be malicious, his reports were discounted and allowed Bob's reputation to recover.

The 3Win method provided a more conservative approximation, allowing smaller fluctuations in the node's reputation than the WMA. This testing concluded that this conservative approach benefited the network because it forced nodes to sustain positive behavior for longer periods than was necessary in the WMA or other reputation management mechanisms.

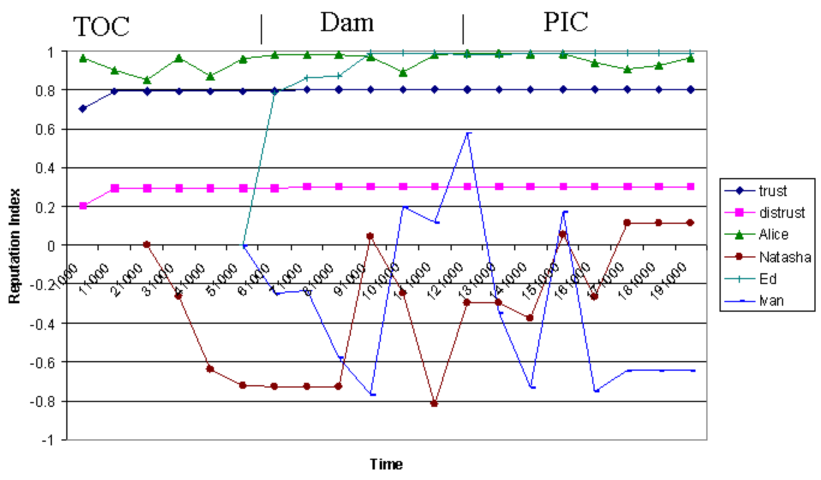


Figure 4-7 Inter-Network Mobility Effects on Reputation Scaling

4.3.2 Inter-network Mobility Testing

Following the verification of the 3Win method and the RSM, the coalition example from Section 4.5.1 was used to create test scenarios that involved moving between network environments of differing densities and peer behavior levels. These tests were intended to demonstrate the RSM's ability to adapt to network environments that displayed varied characteristics. Scripts were created using the same process of determining interactivity and mobility before applying behavior profiles.

The test, displayed in Figure 4-7, showed the system acting on the test vignettes designed in Appendix A. The scenario began in the Tactical Operations Center (TOC), a medium density network environment made up of "good" users. Joe entered the network and associated himself with peers in the area, such as Alice. The presence of unreliable user behavior increased the trust thresholds but did not require any associations to be dissolved. When a "bad" user, Natasha, joined, she was not extended trust based on the information in the referrals received from Alice and other "good" users.

At time mark 61,000, Joe left the TOC and arrived at the Balcony Falls Dam to conduct a site survey. The Dam represented a sparse density, low population network environment of unreliable or resource-constrained users. Joe encountered Ed and, again based on referrals received from other associates, extended him trust while, at the same time, denying trust to Ivan, another "bad" user.

Finally, Joe entered the Public Information Center (PIC), a large, dense network of "bad" users. Having transitioned through two previous network environments, it was important that Joe's RSM be able to continue to differentiate between desirable and undesirable associates. In other words, the RSM had to remember a sufficient amount of previous activity to re-establish or maintain association with people he met throughout the day. Although the presence of other "bad" users enabled Natasha and Ivan to improve their RIs slightly, their previous behavior still prevented them from gaining access to Joe's resources. At the same time, the "good" associates were maintained.

Figure 4-8 illustrates a situation where Joe's RSM was presented with a pair of "bad" users (Ivan and Natasha). This pair was actively colluding to subvert the network. The collusion was effected by having Ivan and Natasha only give each other positive observations while, at the same time, either not provide behavior grades or have them provide negative grades when none were warranted. In this manner, their plan was to allow Natasha to gain a foothold in the network by constructing associations with good users and then use these associations to insinuate Ivan, her confederate, into the network.

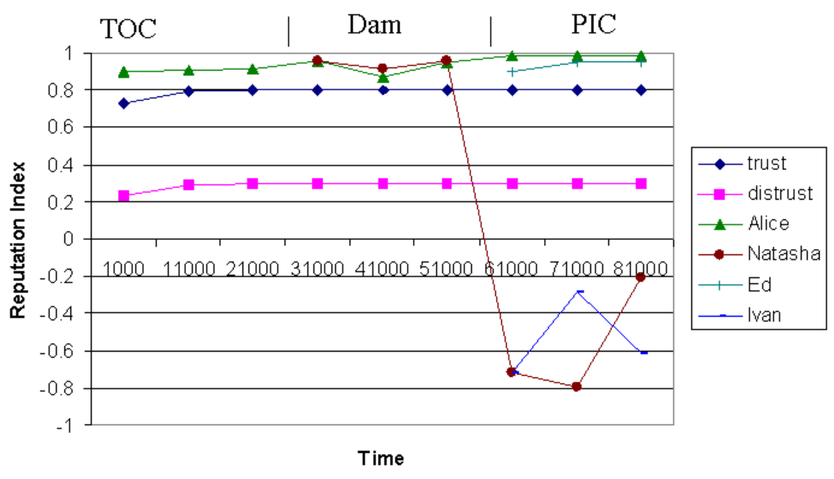


Figure 4-8 Inter-Network Mobility with Collusion

Joe entered the network at the TOC, as in the previous test. He arrived at the Dam and was introduced to Natasha, who was performing as a good user. This association continued when Joe got to the PIC but he then observed that Natasha changed her behavior after she introduced her confederate, Ivan. Joe's RSM, recognizing that Natasha's behavior had become undesirable, lowered her RI as well as the RIs of those that she had introduced or referred. While at the PIC, Joe dissolved his association with Natasha but maintained her behavior history in his TS to enable him to weight her previous observations with her now unacceptably low RI. Furthermore, despite the collusion, Joe was still able to access Ed as a trustworthy sort and allowed him access.

These tests supported the validity of the RSM's performance. The RSM demonstrated the ability to maintain RIs on individual associates in varying network environments. Additionally, the use of dynamic weighting sped the process of identifying and isolating "bad" users by applying current reputations to RI calculation rather than depending on the observer's RI from the time of the observation.

4.3.3 Environmental Impacts

Having confirmed the RSM's performance, the following sections discuss the module's sensitivity to specific network parameters. The following tests simulated a 1000 x 1000 meter area with 100 network nodes. The BonnMotion simulator was used to create mobility and interactivity scripts before a separate C++ program, called Builder, constructed the simulation script by applying the behavior profiles discussed in Section 3.3.1.

The initial perception was that the RSM would prove sensitive to environmental conditions within the DCE, such as network density, general peer behavior, and nodal mobility. During the testing, the trust store size was increased to equal the total number of nodes in the network, effectively giving the observing node an infinite memory and removing the effects of the introduction and forgetting processes. The results focused on the RI calculated for a specific user and concentrated on the accuracy of the RI in reflecting the user's behavior profile.

The following tests gauge the RSM's sensitivity to changes in network density, general peer behavior, and nodal mobility. The goal was to generate an RI that was accurate and stable, so the tests looked for an RI that matched what we expected of that behavior profile and had as little fluctuation (expressed as variance) as possible.

4.3.3.1 Network Density

Network density was tested in a static network of 100 uniformly distributed nodes to determine the effect of changing the number of associates had on the RSM's ability to calculate an accurate RI. The density was expressed as the number of nodes within the observer's communication range. These nodes formed a set of potential associates. As the range increased, the set grew larger and the network was denser from that observer's point of view.

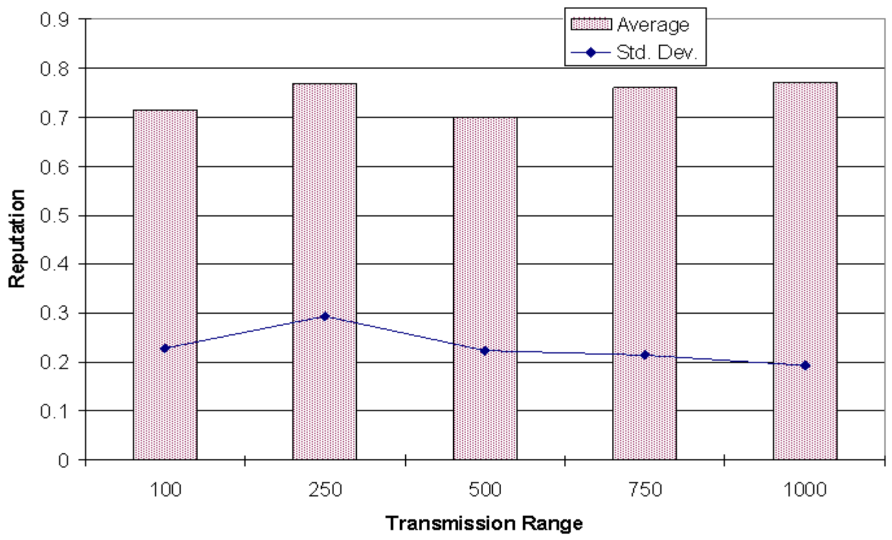


Figure 4-9 Reputation-Scaling Sensitivity to Network Density

Figure 4-9 illustrates how the RSM performs accurately in all network environments. Increasing the observing node's transmission range from left to right made the networks denser. The observing node's set of associates grew in corresponding increments. The behavior model parameters (see Figure 3-12 and Figure 3-13) were set so that a good user stayed in his natural state and performed desirable actions 80% of the time. The RSM accurately tracked this behavior and produced an RI that reflected these parameters. The low standard deviation, which matched the amount of time a user spent in his "unnatural" state, indicated that the 3Win method produced a stable value that could be used with confidence in making trustworthiness decisions.

4.3.3.2 General Peer Behavior

General peer behavior was the ratio of good users to bad users in a network. The intent was that networks with a higher percentage of bad users presented the RSM with fewer reliable behavior observations and, therefore, made the accurate reputation calculation more difficult. As demonstrated by Figure 4-10, this perceived difficulty in reputation assessment did not materialize.

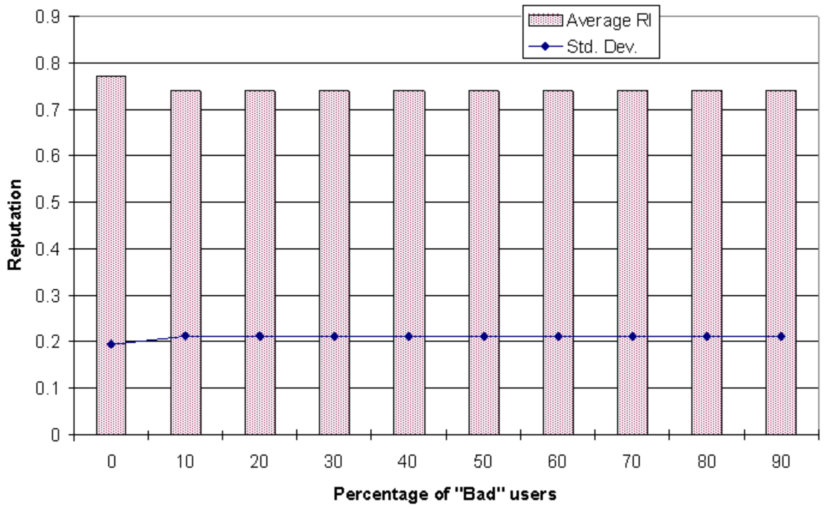


Figure 4-10 Reputation-Scaling Sensitivity to General Peer Behavior

Tests were conducted using a static network of 100 users in a 1000 x 1000 meter simulated area. As with the network density tests, the trust store size was increased to give the observing user the ability to remember all associates in the network. Transmission range was set to 1000 meters, creating a completely connected network. Tests increased the percentage of bad users in this mesh environment but the RSM performance was stable. Even in networks where the majority of users were misbehaving, the RSM accurately captured the target "good" user's performance.

4.3.3.3 Mobility

Static networks model wireless and wired networks that change through the presence or absence of users. Although the RSM had performed as expected in static networks, tests were conducted to assess the impact of mobility patterns. This testing involved users that were still part of the network but might not be in contact with all of the peer nodes at the same time.

These tests compared the RSM's performance in static environments to the Random Waypoint (RW) and Reference Point Group (RPG) models. These models were selected because they present an opportunity to examine individual as well as group mobility. Specifically, we were concerned with the mobility's impact on interaction and, subsequently, on reputation assessment.

In addition to testing with the theoretical movement, both of these models also allowed constraints, called "attraction points" to be inserted in the simulation area (Camp, Boleng et al. 2002). Nodal movement was altered to tend toward these points instead of completely random travel. Attraction points were added to the RW and RPG simulation parameters and are displayed in Figure 4-11 as RWa and RPGa. These points were located in accordance with the testing scenario described in Section 4.5.2.

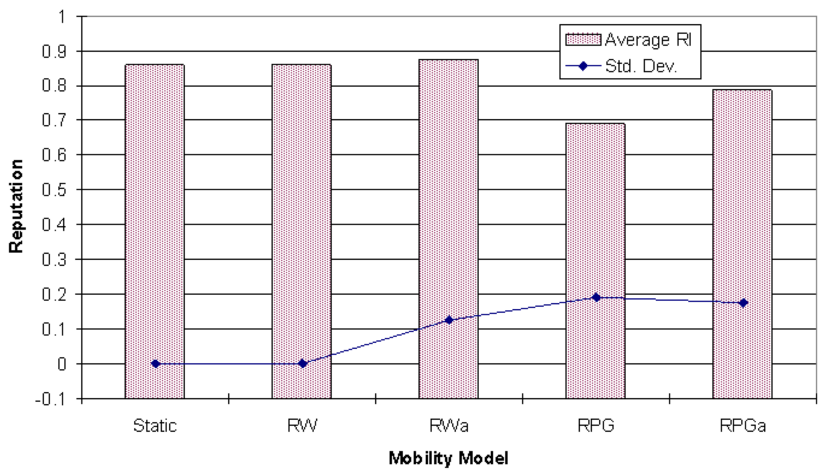


Figure 4-11 Reputation-Scaling Sensitivity to Network Mobility

The tests used a 100 node network moving through the one square kilometer area with a 100 meter transmission range. Ten percent of the network users had bad behavior profiles. The trust store size was equal to the total number of users in the network, for reasons discussed previously.

Tests showed that the RSM performance was similar for static and individual mobility models (RW and RWa). This result stemmed from the fact that interaction between nodes in these models was between two individuals. In essence, the individual-based interaction made each association bilateral and transient. Nodes met, exchanged information, and then moved on or left the network. In contrast, the group mobility models added a sort of peer pressure to establishing and dissolving trust associations because a user had more reason to trust its group members than to exchange information with new associates. The end result was that the group mobility models exhibited a lower average RI and a higher deviation range in the reputation of a selected "good" user.

Of particular interest was the effect of the attraction points on the reputation calculation. Skewing nodal movement to certain areas increased the number of interactions compared to the tests with individual mobility. As a result, users tended to join and maintain association with a flow of peers. Longer and more frequent associations lowered the average RI and increased the standard deviation because the "bad" users had the opportunity to provide more disparaging input.

4.3.4 Contributions and Conclusion

The RSM advanced the current state of the art by introducing a reputation scaling mechanism that maintained a memory of past behavior grades and observers. The RSM used this historical knowledge to apply the observer's current RI to any behavior grade he might have made. This reevaluation was called dynamic FI weighting and it proved very successful in isolating not only misbehaving nodes but also nodes that might be colluding with them.

In addition to dynamic FI weighting, the RSM's 3Win reputation scaling equation provided a more conservative approximation, allowing smaller fluctuations in the node's reputation than other equations currently in use. This testing concluded that this conservative approach benefited the network because it forced nodes to sustain positive behavior for longer periods than was necessary in the WMA or other reputation management mechanisms to achieve the same positive reputation.

The RSM performed as expected and provided the TMS with a basis for trust decisions. It maintained correct reputation assessments on associates regardless of the characteristics of the other network users. The RSM, however, was shown to be sensitive to normal network conditions like mobility and density. This sensitivity was the impetus behind the requirement for a means to adjust trust thresholds to account for uncertainty in changing network environments, which is discussed in the next section.

4.4 Risk Assessment

The RAM's uncertainty estimation was innovative in the field of access control. Previously, researchers have proposed risk analysis techniques that hypothesize cost-benefit planes or attempt to manufacture probabilistic scenarios that estimate payoff against commitment. In general, these techniques remain restricted to theoretical discussions because of the number and complexity of the variables required. In the end, these researchers set arbitrary values for variables to demonstrate their systems.

In contrast, the RAM modeled human behavior to estimate uncertainty in the surrounding area. By relying on first hand observations, the RAM gauged the trustworthiness of the local network environment much as a person estimated the safety of a neighborhood as they walked down the street. The RAM used the uncertainty estimate to adjust the TMS's trust thresholds with the intent of providing more protection to the user.

This practical implementation of uncertainty estimation was shown to improve the efficiency of access control decisions (see Section 4.4.1). Reinforcing this contribution was the fact that this improvement was implemented without increasing the cost associated with making those decisions. Finally, the RAM allowed the TMS to tune trust thresholds dynamically, resulting in a set of thresholds that was based on network conditions rather than best-guess pre-configuration.

The Risk Assessment Module (RAM), described in Section 3.2.4, produced an assessment of risk in the network. The RAM processed behavior information that was either directly observed or obtained from the KMS to estimate the uncertainty in the local network area. Using a filtering mechanism, the RAM produced a Global Risk Index (GRI). The GRI was used to adjust the user's trust thresholds in response to the local uncertainty and protect the node in changing network behavior conditions.

Without the RAM, the user ventured into a dynamic environment with fixed trust thresholds. There were two major deficiencies with using fixed trust thresholds. First, these thresholds were picked arbitrarily and had no connection with the environment within which the user was working. Second, because fixed thresholds did not take into account the uncertainty of the networking environment, this situation led to the user making incorrect trust decisions and either exposing himself to risk unnecessarily or rejecting otherwise viable associations.

Verification testing demonstrated the RAM's contribution to making accurate access control decisions. The uncertainty estimation was conducted in a mixture of network environments and mobility models, each test measuring the efficiency of adjusting the thresholds against systems with fixed settings.

The RAM was sensitive to the volume of KMS reports in the system. KMS reports indicated trust establishment and dissolution throughout the network, as replicated by the KMS, if one existed. As discussed in Section 4.4.2.1, the RAM correctly processed reports in a completely connected network. The reports concerned direct associates and were an accurate representation of the user's local network environment. In larger, less connected networks, however, concentrations of misbehaving users (e.g., "bad neighborhoods) skewed the GRI and resulted in a user in a safe network adjusting his thresholds too quickly or without reason.

4.4.1 General Testing

The RAM was tested to ascertain the impact of network behavior on trust thresholds. While it was desirable for the RAM to adjust the thresholds to protect the TMS, the RAM had to regulate itself to prevent the TMS from "self-isolating" in a malicious environment. In the following tests, RAM efficiency was measured by comparing the TMS's ability to make correct trust decisions with adjustable trust thresholds and with fixed thresholds.

A user in a completely connected or mesh network had the ability to associate with all of the nodes in the network. Figure 4-12 shows the improved protection allowed by enabling the RAM to adjust trust thresholds based on the current network population. Static mobility in a mesh network models an Internet or wired environment, where the network changed because users entered or departed. The RW mobility model presented the worst case in interactivity potential (McNett and Voelker 2004) in terms of changing the environments a user saw in a network. In both cases, testing demonstrated the efficacy of adjusting the thresholds to reflect the current uncertainty in the user's network environment.

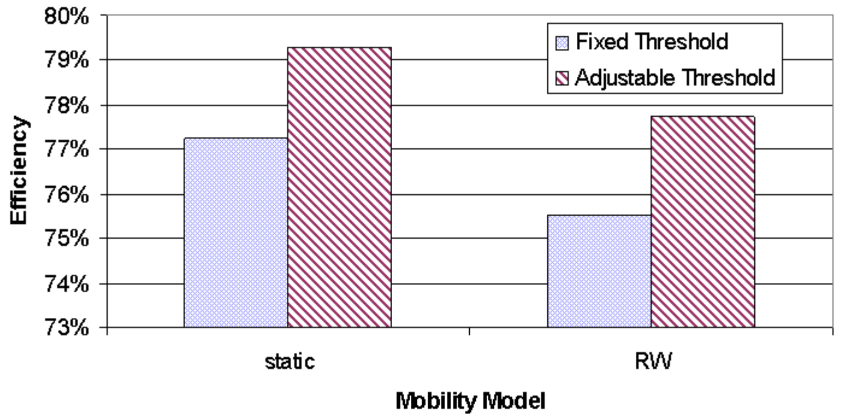


Figure 4-12 Risk Assessment Module Efficiency in Completely Connected Networks

4.4.2 Environmental Impacts

Because of its reliance on reports and observations, the RAM showed some of the same sensitivities that the RSM did. In general, KMS activity posed the greatest possibility for the RAM to over-compensate for poor behavior. Network density, defined as the number of peers within a user's communication range, and general peer behavior were investigated but did not pose significant challenges to the RAM's ability to assess the network's uncertainty level correctly. Finally, tests using different mobility models did not appear to affect the RAM and were more closely aligned with the results of the network density tests in terms of RAM performance.

4.4.2.1 KMS Activity

At first, the RAM relied on KMS activity to determine the general uncertainty in the network. During testing, however, this reliance was showed to contribute a distraction to the TMS and was removed. This distraction took the form of an increased number of false negative decisions; instances where trustworthy associates were denied access to a resource because the RAM had incorrectly adjusted the trust threshold. In place of the KMS reports, the RAM focused on first hand observations to gauge the local uncertainty level.

4.4.2.2 Network Density

While the RAM yielded positive contributions to the system's efficiency in mesh networks, as depicted in Figure 4-12, the RAM performed significantly better in less connected and mobile situations, depicted in Figure 4-13. This increased performance was the result of a user dealing with a smaller set of associates, even if the associations were for shorter periods of time.

The tests used a 100 node network moving through the one square kilometer area with a 100 meter transmission range. Thirty percent of the network users had bad behavior profiles. The trust store size was equal to the total number of users in the network, in order to isolate the performance of the RAM.

4.4.2.3 General Peer Behavior

As shown in Figure 4-13, the RAM reacted to adjust the thresholds. The riskier the network, the quicker and higher the thresholds were adjusted. The RAM was actually more efficient in environments with high percentages of misbehaving users. Testing demonstrated that the RAM never self-isolated, even when the network was populated with 100% bad users. These adjustments resulted in increased protection for the TMS and more accuracy in judging the trustworthiness of a prospective associate.

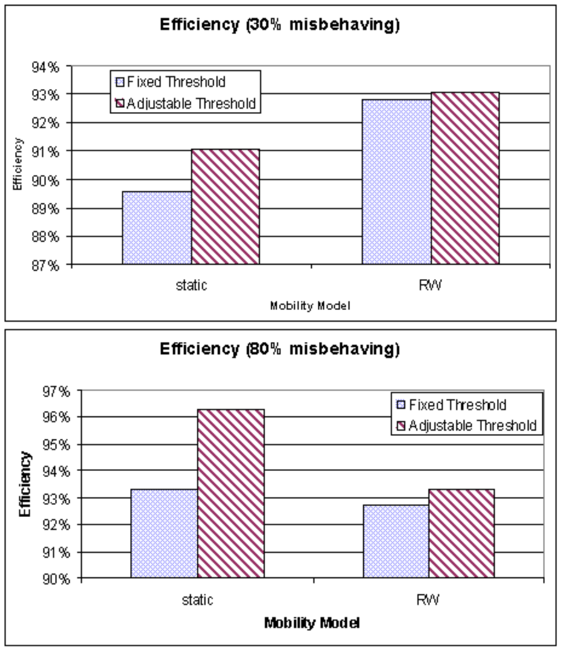


Figure 4-13 Risk Assessment Efficiency in Changing Behavior Situations

4.4.3 Conclusion

Verification testing demonstrated the RAM's contribution to making accurate access control decisions. The uncertainty estimation was conducted in a mixture of network environments and mobility models, each test measuring the efficiency of adjusting the thresholds against systems with fixed settings. In all cases, the dynamically tuned thresholds were more effective, resulting in more accurate access control decisions.

Reinforcing the increased accuracy was the fact that this improvement was implemented without increasing the cost associated with making those decisions. Risk assessment techniques that used probabilistic methods or cost –benefit models presented a higher cost in storage and processing than did the RAM's filtering equation. Although the RAM used a small amount of memory in its estimation, it did not require the near global knowledge and content specific parameter settings that other systems required.

4.5 Validation: Goals and Objectives

Validation ensured that the system met the user requirements. In our case, validation guaranteed that the modules of the TMS worked together to make access control decisions correctly under a variety of network conditions. Validation differed from verification testing in that the system was tested against operational requirements instead of purely quantitative comparisons.

The requirements used in validation testing came from two sources. The first source was verification testing. These requirements, derived in part from the analysis presented in Section 3.1.3, placed the system in a test environment that simulated the target operational conditions. As mentioned in Section 4.1, the points of failure identified in each module during verification testing were added to the validation test profile to determine the impact of a module's limitations on the system as a whole. The goal was that the system continued to operate or at least failed in a safe state when these points of failure were reached. For an access control system, such as the TMS, failing in the "closed to all" state was desirable, since it is better to deny access to everyone at the expense of false negative responses than to fail in the "open" position and suffer false positive responses, which are more costly.

The second source of validation requirements was an operational scenario, in our case the scenario needed to involve mobile, collaborating users asking each other to share resources.

Once the general conditions of the scenario were determined, we applied a narrative technique to construct the test environment for the system (Lo Presti, Butler et al. 2005).

First, the objectives that needed to be tested within the system were enumerated. These objectives addressed operational issues within the broad topic areas, such as mobility, network density, and general peer behavior. Objectives were expressed in the form of task-condition-standard in order to be evaluated.

Then, a detailed operational scenario was created. The scenario provided a framework that modeled the user requirements and gave realistic vignettes in which to test interaction. In this scenario, we detailed the composition and deployment of a notional disaster response task force, as in Section 4.5.1. Since the system is specifically concerned with the access control of resources within a collaborative environment, notional users were assigned as resource providers.

Finally, vignettes were written to frame the objectives within the scenario to test the objectives. The scenario not only provided a realistic approach to developing the vignettes but also helped order the tests if need be. Lo Presti's narrative technique included mapping objectives to vignettes and this exercise was included in Appendix A.

Once the system's functionality was established, boundary analysis provided an indication of the most and least favorable conditions for the system's use. The following sub-section begins the presentation of validation testing by discussing the operational scenario.

4.5.1 Operational Scenario

The operational scenario used as an extended example in this research was based on Training Scenario 2: Slow Building River Flood – Natural Disaster (FEMA 1994b). This scenario called for cooperation between a variety of government organizations, local volunteers, and federal advisors. County and state law enforcement agencies, a local volunteer fire department, volunteer rescue squad, and county health facility initially formed the coalition. Army National Guard (ARNG) units arrived as soon as possible to provide technical or specialist assistance. The coalition was formed using the Incident Command System (ICS) (FEMA 2004) as the basis for its organizational and procedural structure. In accordance with the ICS guidelines, a public information officer established a cell to provide media news services with information.

Figure 4-14 illustrates how the coalition might deploy in response to the notional emergency response situation. In this example, the coalition deployed to evacuate the inhabitants of a small riverside town and secure the town's infrastructure against damage. The regional Emergency Service Management Agency established a coalition HQ and information center at the incident site. County and state law enforcement, assisted by fire units and local volunteers, directed the evacuation of the town's inhabitants to the higher ground behind the town, where Red Cross and other local organizations established a shelter. Medical units treated injured people and evacuated a senior citizen home, assisted by the MEDEVAC helicopters and rescue squads. An ARNG engineer unit deployed sub-units to contain contamination from the town's two sewage disposal sites and to reinforce the Balcony Falls Dam. Although this scenario included severe inclement weather, members of the coalition and the outside populace were able to move about the scenario location. Communications, although not reliable, were present between the coalition locations and the unaffected "safe" areas.

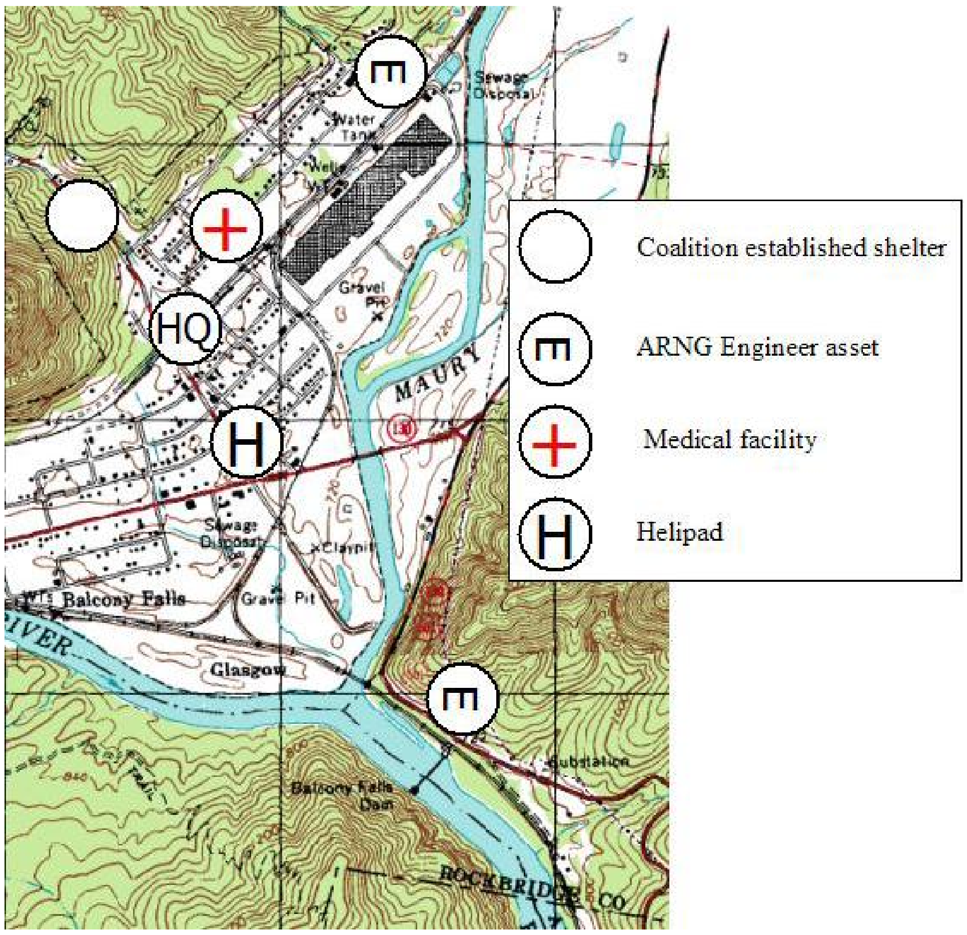


Figure 4-14 Operational Scenario Map

Table 4-1 Coalition Locations and Resources

Location	FEMA	ARNG	Fire/ Police	Red Cross
Shelter	🧍 🖧			🧍
Tactical Ops Center (TOC)	🧍 🖧 ✉ ⊠ 🖨	🧍	🧍 🌐 ✉ ⊠ 🖨	
Medical Facility			🧍 🖧 ✉ 🖨	▨
Helipad		🧍 🌐 ✉ 🖨		
Engineer 1 (Sewage Plant)	▨	🧍	▨	
Engineer 2 (Balcony Falls Dam)	▨	🧍		
Engineer 3 (Power Substation)	▨	🧍		
PIC	🧍 🌐		▨	🧍

🧍	People (▨ = transient)	🖧	Intranet (HTTP)
🖨	Printer	🌐	Internet (HTTP)
✉	Web access (HTTP/SSL)	⊠	File server (HTTP)

In terms of information resources, the coalition represented a hastily formed DCE. Users possessed a variety of computing and communications platforms that utilized both wired and wireless communications. This research focused on the ability for users to access coalition resources and, similarly, for the resource owners to maintain control and protect their resources for the use of contributing coalition members.

Given the composition and deployment of the notional coalition, we distributed resources for coalition use. For example, the coalition might leverage the connectivity present at the police building to co-locate the TOC and coalition Headquarters (HQ). The community fire department and clinic would provide a location for the medical unit.

Table 4-1 provides an example of the resources that require access control. This distribution of coalition assets was used in TS and RSM verification testing to represent the types of networks present in the operational scenario (see Sections 4.2 and 4.3). Later, validation testing placed the TMS in the role of the TOC's server to assess system performance.

4.5.2 Test Environment

Testing the TMS involved simulating user interaction in a mobile, dynamic environment. A four-step process was developed to create scripts of behavior reporting. These scripts simulated the activity of the KMS and the Intrusion Detection System (IDS) layers of the security architecture and were read by the TMS during the course of the simulation.

The first step constructed a simulation area using parameters applicable to the operational scenario. BonnMotion 1.3a (de Waal and Gerharz 2005) simulated node movement inside a 3,000 x 4,000 meter bounded area. Attraction points mimicked the effect of roads and facilities on nodal movement. Each Attraction Point was given an (*x,y*) coordinate, roughly corresponding to the map in Figure 4-14. The intensity value of the point weighted the attraction points so that a point with an *x* intensity level attracted nodes with a probability *x* times higher than an un-weighted point. Locations with higher intensity values were predicted to have heavier traffic. Nodes would approach an attraction point to a location within the point's standard deviation from a Gaussian distribution with a mean of 0 in meters. The resulting Cartesian representation of the simulation area is shown in Table 4-2.

Table 4-2 Simulation Attraction Point Parameters

Location	X	Y	Attraction	Std. Dev.
Shelter	300	3200	1.5	20
Hospital	800	2800	2.0	20
TOC	600	2200	4.0	20
Helipad	1000	1900	2.0	10
Bridge	1800	1800	5.0	10
Dam	1600	1000	2.0	10

The second step of the process involved three mobility models. The first model was static, meaning that the nodes were homogeneously distributed over the simulation area and did not move. The second mobility model was the RW model (Camp, Boleng et al. 2002). Simulations run using the RW model used the node speed and pause time parameters indicated in Table 4-3. The RPG mobility model was used to simulate group mobility. In addition to the speed and pause parameters that it shared with the RW simulations, RPG required settings to describe the group dynamics in the simulation. These settings, shown in Table 4-4, show that the simulation had groups of one to five people. These groups were stable, in that the chance of people changing groups was low. Raising this probability skews the RPG results toward those seen in individual mobility models, such as RW. The groups moved in open order, as the group members could be as far as 2.5 meters from each other. Each mobility model was executed on the same area mentioned above with and without attraction points. By executing the chosen parameters on the selected grid and mobility model, BonnMotion created a movement trace file for all the nodes in the network.

Table 4-3 Simulation Parameters

Duration	5000 secs.
Warmup	3600 secs.
Sim area	3000 x 4000 meters
Nodes	100

Speed	Min = 0.5 m/s
	Max = 10 m/s

Pause Time	60 sec. (Max.)

Table 4-4 Group Mobility Simulation Parameters

Average Nodes per Group	3 (Std. Dev. 2)
Group Change Probability	0.01
Distance to Group Center	2.5 meters

Third, the movement trace was fed into BonnMotion's companion program, LinkDump. This program read the movement trace and applied a transmission range of 100 meters (selected to simulate 802.11 traffic) to determine when pairs of nodes could interact. The interaction file that was produced listed each node and its unidirectional communications link. Having each interaction listed twice reflected the "one-way" nature of the link. For example, if Alice could interact with Bob, two links were listed: "Alice to Bob" link was listed in Alice's part of the file and the "Bob to Alice" link was listed in Bob's portion. Having the links listed in this manner facilitated the next step, which was determining who could observe whom.

The fourth and final step of the script generation process was to generate behavior and trust related network traffic. A reporting period was selected. In this case it was assumed that each node generated a behavior grade once every ten seconds. A bin in a linked list represented each reporting period. Each bin was itself a linked list of behavior grades for that time period. A C++ program called Builder was constructed to read the interaction list and populate the bins with observations and reports. The simulation began with a set of introduction requests and responses, the format of which is illustrated in Figure 4-15. These transactions placed associates in the TMS' Trust Store. Once an associate was known, the generated traffic represented the flow of behavior observations and reports.

As Builder read each link from the interactivity list, it called on the behavior model to determine a grade for the observed node for that reporting period. That grade was then adjusted based on the observer's behavior model. The final behavior grade, in a format shown in Figure 4-15, was placed at a random point in the reporting period's bin. Once Builder had read the entire interactivity list and filled all of the appropriate bins, the behavior grades were written to a script file of network traffic formatted for the TMS.

Initializing the scenario required that the user (e.g., Joe) be introduced to someone by the KMS. Once Joe had a trusted peer (TP), he could roam around and make other friends. This startup requirement was viewed as feasible; since Joe would be introduced to the people he would be working with when he arrived at the TOC, thus allowing Joe to start associating in the DCE.

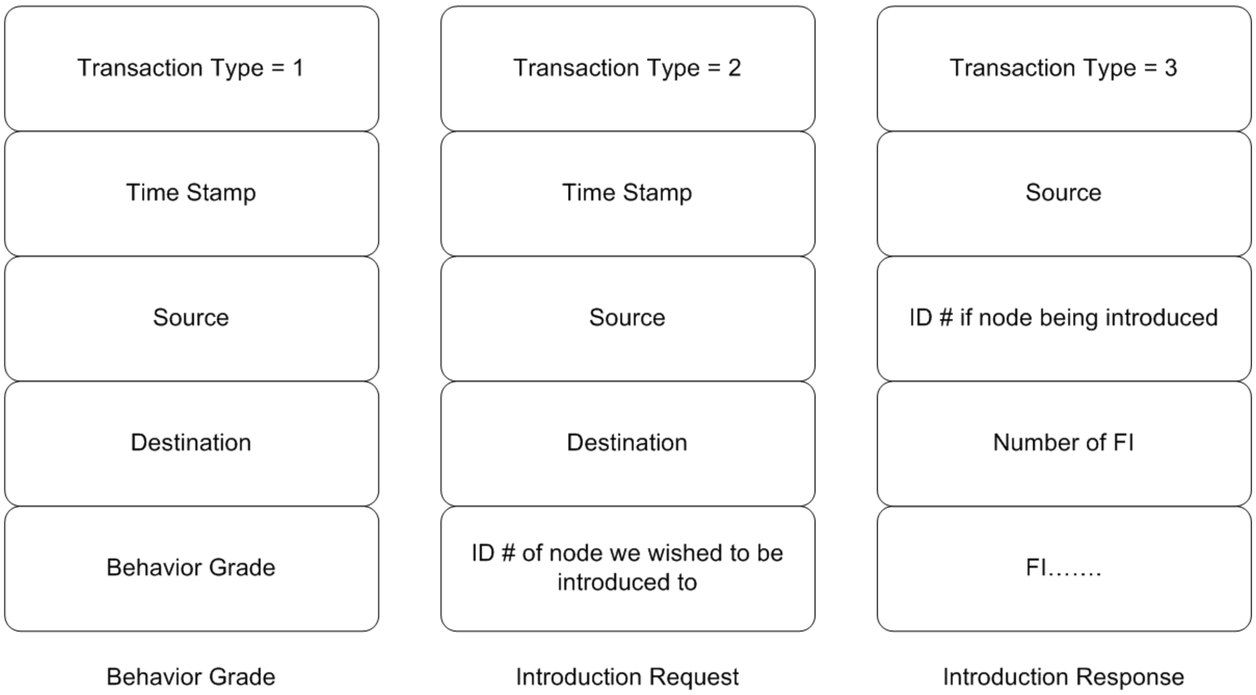


Figure 4-15 Format of Trust Information Transactions

4.6 System Testing

The previous testing created interactivity scripts using BonnMotion and the behavior model detailed in Section 3.3.1. Building from this experience, the system testing took the point of view of a single user in a DCE. Starting with the characteristics and resources of a DCE responding to a natural disaster as discussed in Section 4.5.1, the test system simulated the network environment a user would come into contact with.

Testing the TMS required a means of simulating service requests received by a resource providing DCE member from associates. Our simulation assumed the viewpoint of the server in the TOC and processed requests for files via a HyperText Transfer Protocol (HTTP) user interface. Modeling requests typically made to the resources illustrated in Table 4-1, we examined the process of modeling a typical wireless system (Ost 2001). Given a generic inter-arrival rate we can determine the number and period of resource requests in our notional scenario.

Requests were classified by the nature of information being sent or received. There were two general types of information: simple files and composite files. Simple files were single data type (text or graphics) files. Examples of these included email, web page (without graphics), or text files that were exchanged through an HTTP process. Composite files were multiple simple files linked together. Web pages with graphics were the most common examples. Each file type could come in one of three sizes. After determining the type and size of a request, the request duration was determined by approximating the times depicted in a "slow Internet" connection (Heidemann, Obraczka et al. 1997), again following Ost's example.

The test system simulated resource requests in a three step process. First, the system determined if there was a request being serviced. If the system was free, it checked to see if there was a request. Requests were serviced on a first come, first served basis, with no attempt being made to restore or save requests that might be lost if a system was busy. When there was a request, the system determined the type. The system was then flagged as busy for the duration specified for that type of request. The probability and duration for each type of request is shown in Table 4-5.

Table 4-5 Probability and Duration of Resource Requests in a Simulated Collaborative Environment

Request Type	Probability	Duration (seconds)
Small Simple File	0.6	1
Medium Simple File	0.1	2
Large Simple File	0.05	8
Small Composite file	0.15	1
Medium Composite File	0.075	6
Large Composite File	0.025	27

In order to provide a frame of reference for the results gathered during testing, a base system was constructed using the basic reputation aggregation equations and principles developed by Buchegger (Buchegger and Le Boudec 2003b) and through the SECURE project (Gray, O'Connell et al. 2002). The base system utilized an exponential weighted moving average equation for reputation scaling. It had fixed trust thresholds and exchanged reputation index values during a modified introduction process.

In addition to the work of the previously mentioned authors, the base system was equipped with a trust store-like reputation storage to enable the system to weight behavior grades upon receipt. During all tests, the same underlying interactivity traces and behavior models were applied during the creation of the test scripts. Although the simplicity of the base system appeared beneficial at first glance, testing revealed serious deficiencies in its performance. The most notable deficiencies were noted in the investigation of the success metric.

4.6.1 Success Metric

The system metric, developed in Section 3.4, was comprised of two components. The first was a success metric, explained in Equation 6 (q.v., page 69), that expressed the number of correct decisions the system made as a ratio against the number of false positive and false negative decisions. The remainder of this section will utilize Equation 6a to eliminate the arbitrary nature of the weighting factor ω that is used in Equation 6b.

Figure 4-16 shows the three components of the success metric. These tests, performed in a 100 node network with 30% bad users, illustrated how well the TMS responded to resource requests in three mobility cases. The graph shows the proportional contribution of each response category to the over success rate. Ideally, the column should be 100% correct ($D - (P + N)$) to represent that the system answered all of the requests correctly. Barring this situation, the goal was to minimize the number of false negative responses (*N*) and then to eliminate the number of false positive responses (*P*).

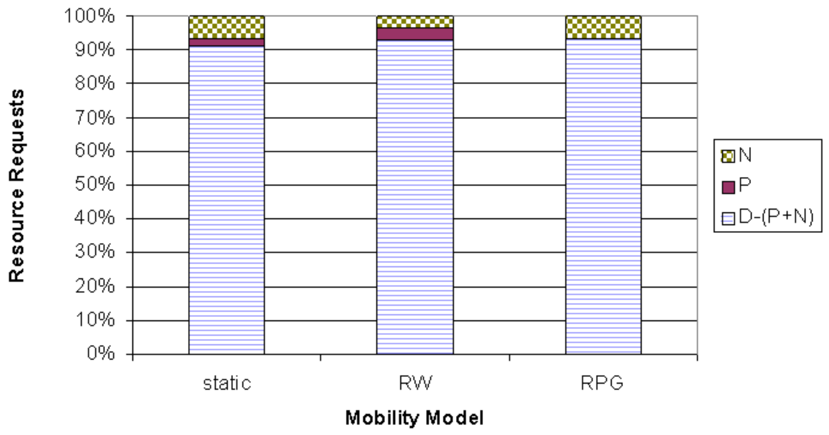


Figure 4-16 Components of the Success Metric

The TMS performed well in the static case, having 91% overall success in responses, but had moderate numbers of false positive and false negative responses. The overall success rate improved slightly in the RW case to 93% but the incidence of false positives almost doubled as a proportion of the incorrect responses. These false positive responses are of concern because they represent missed opportunities for information exchange and the possibility for a trustworthy peer to submit negative behavior reports on an otherwise "good" user. The RPG case was the most worrisome. Although the overall success rate increased to 94% and there were no false positive reports, the proportion of false negative reports doubled once again to represent 6% of the total number of requests. This testing illustrated the importance of examining the contributing components of the metric in addition to examining the overall percentage of correct responses.

The ratios presented in the previous tests are put into a better frame of reference when the TMS results are compared against those of the base trust system. Extending the discussion on the metric components, Figure 4-17 shows how the base system performed. In addition to having a lower overall success percentage, the base system exhibited an extraordinarily high percentage of false negative responses. This high proportion was due to the lack of historical knowledge maintained by the TMS for dynamic weighting of behavior grades.

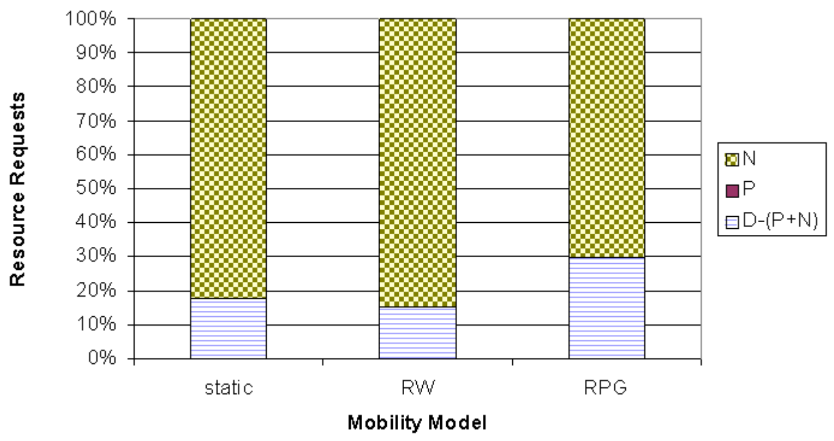


Figure 4-17 Success Metric Components of the Base System Test

The comparison between the TMS and the base system clearly showed the benefits of the 3Win method and the impact of dynamic grade weighting. Figure 4-18 shows the comparison of success of the TMS and the base system in different mobility models using a general peer behavior condition of 30% bad users as TMS30 and Base30, respectively. While it had been expected that the base model performed would show less efficiency than the TMS, the poor success percentage in the static and RW models was surprising considering the general ratio of good users to bad was rather high. While the base system efficiency increased slightly in the models with attraction points (RWa) and group mobility (RPG), it never demonstrated better than 30% efficiency.

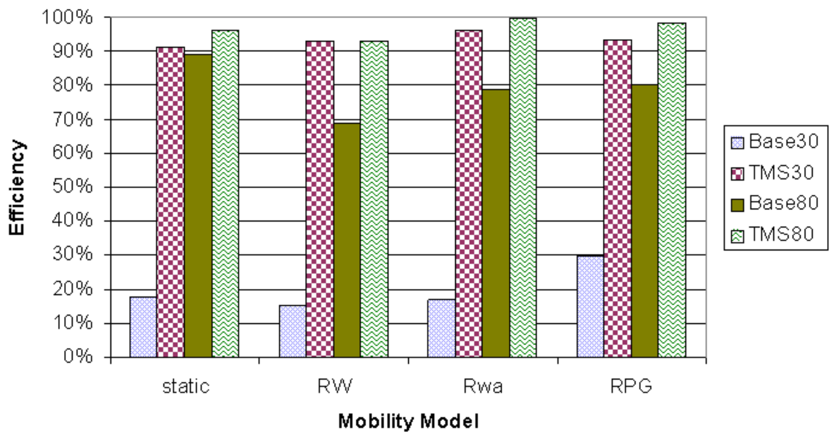


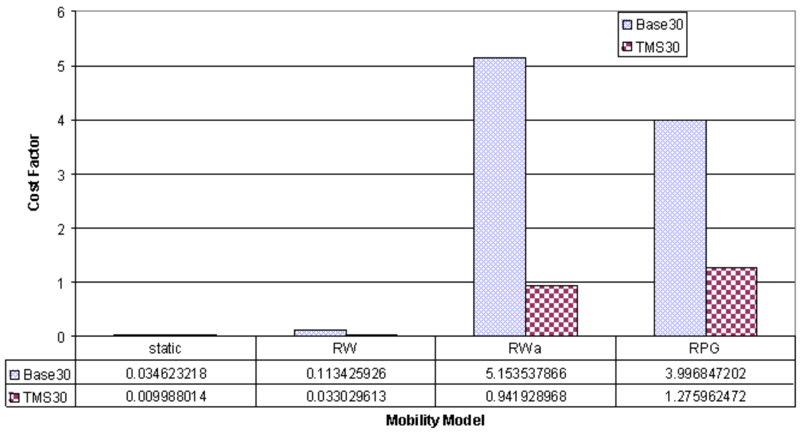
Figure 4-18 Comparison Base and Trust Management System Success Rates

As the proportion of bad users increased, the TMS efficiency remained at or over 90%. The base system reached its highest performance level when there were 80% bad users (see Figure 4-18, TMS80 and Base80, respectively). This case simulated a situation where the TBAC system was effectively presented with fewer trustworthy associates to select from.

4.6.2 Cost Metric

The second factor in the system metric captured the cost of making the decisions. Using Equation 7 (q.v., page 70) the communications and storage costs of each method were combined to convey a sense of the behind the scenes requirements for making trust-based access control decisions. The TMS incurred a fixed cost of having to store and exchange all of the FI in its RIW while the base system only maintained an RI, so the general perception was that the TMS would be at a disadvantage in this comparison.

What tests determined, however, was that the TMS cost was far lower than the cost incurred by the base system under the same test conditions, as shown in Figure 4-19. This phenomenon occurred because, while the size of the base case introductions was much smaller than those used by the TMS, the number of introductions was an order of magnitude higher. In most cases, the difference between the base system and the TMS was 3:1 but, under RW mobility tests, the difference grew to four or five to one bias against the base system.



	static	RW	RWa	RPG
▨ Base30	0.034623218	0.113425926	5.153537866	3.996847202
▨ TMS30	0.009988014	0.033029613	0.941928968	1.275962472

Figure 4-19 Comparison of Cost Factor Between TMS and Base System

When the two factors are combined and displayed, the overwhelming efficiency of the TMS is reinforced. The TMS costs are several times less than those of the base system, while providing much higher efficiency. As discussed above, the combination of maintaining historical behavior grades, dynamic weighting of feedback items at every reputation calculation, and adjusting trust thresholds based on current levels of uncertainty have resulted in a much more robust trust system.

Figure 4-20 illustrates this point using the RW model and a network of 30% bad users. The TMS displays low cost and high efficiency while the base system provided less success and more cost. Although changes to the base system might compensate for some of the deficiencies, the use of a memory-less computation method like the exponential weighted moving average puts and keeps the base system at a disadvantage. Furthermore, the implementation of adjustable thresholds and dynamic weighting in the TMS make it more flexible and able to adapt to a wider range of network conditions.

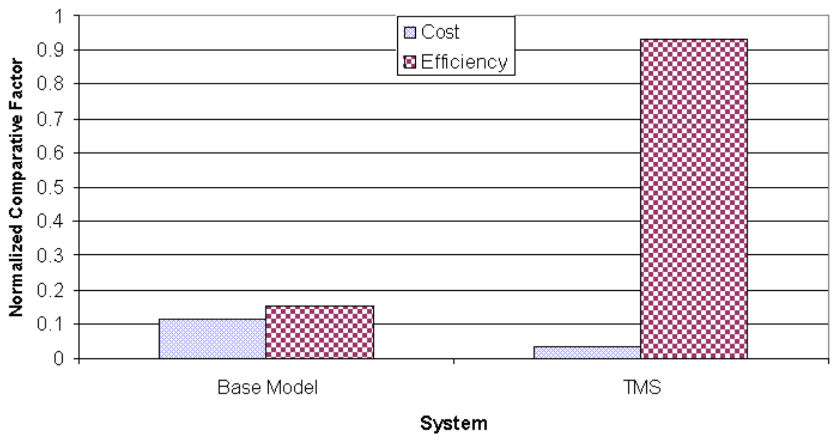


Figure 4-20 Comparison of TMS Success Rate to Base Case System

Having established the comparative performance of the TMS in a variety of network environments, the following section isolates specific DCE criteria to determine the sensitivity of the TMS to changes.

4.6.3 Environmental Impacts

The TMS was tested to determine the impact of network density and general peer behavior on the efficiency of the TMS. Previous testing had established the efficiency levels given various networking parameters but further investigation was needed to quantify the most favorable operating conditions. These tests all used a network of 100 users in a one square kilometer area. The TS size was set to |TS| = 50, as determined by the verification testing to be at the optimal tradeoff point between cost and efficiency.

4.6.3.1 Network Density

Sparse networks were defined as having few users within each other's communications range. Dense networks were the opposite, having lots of people for a user to chose to interact with. The TMS worked best in sparse networks because the network population rarely outran the size of the TS. In effect, the TMS could afford to remember everyone. The downside was that the number of observations was lower, leading to a lack of fidelity in the reputation index. This deficiency was not remedied by requiring the nodes in a sparse network to increase their reporting rate. Rather than improve the fidelity, in fact this increased the "padding" in the RIW and decreased the effectiveness of the RSM. It was, therefore, not implemented.

The impact of network density on the system metric was tested in a static environment, with the same parameters as used during the verification testing of each of the modules. As the observing node increased its transmission range, more and more peers were interacted with and the number of introductions increased. This fact is observed in Figure 4-21 by noting the increase in cost as the range increased. At the lowest range, the efficiency was highest (96%) but so was the cost.

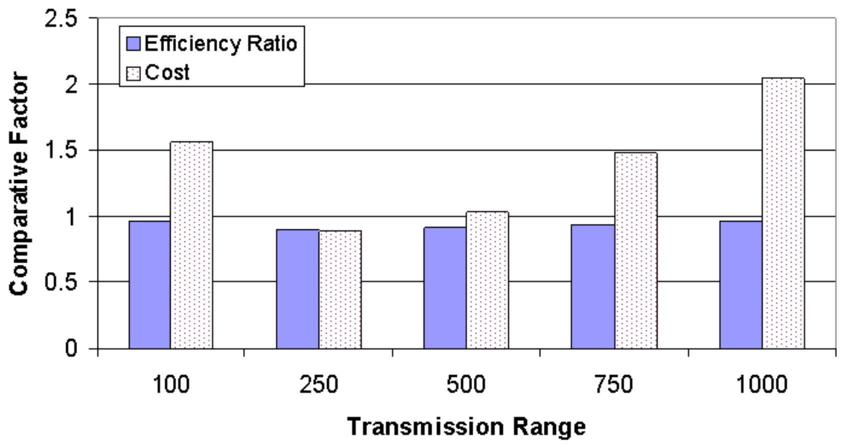


Figure 4-21 Sensitivity of System Efficiency to Network Density

4.6.3.2 General Peer Behavior

The ratio of good versus bad users had an impact in both the reputation scaling and risk assessment modules. This test was conducted in a completely connected network with a TS size equal to the total number of users in the network. Although it was already been shown that this combination of system parameters was not optimal, they were selected to demonstrate the effect of the general uncertainty in the network on efficiency. Figure 4-22 illustrates the steady rise in efficiency as the percentage of bad users in the network increased. This rise equated to an approximately 1.5% improvement in efficiency for each 10% increase in the number of bad users. In the RSM, having a lot (>50%) of bad users led to increased efficiency as the pool of prospective (i.e., trustworthy) associates decreased.

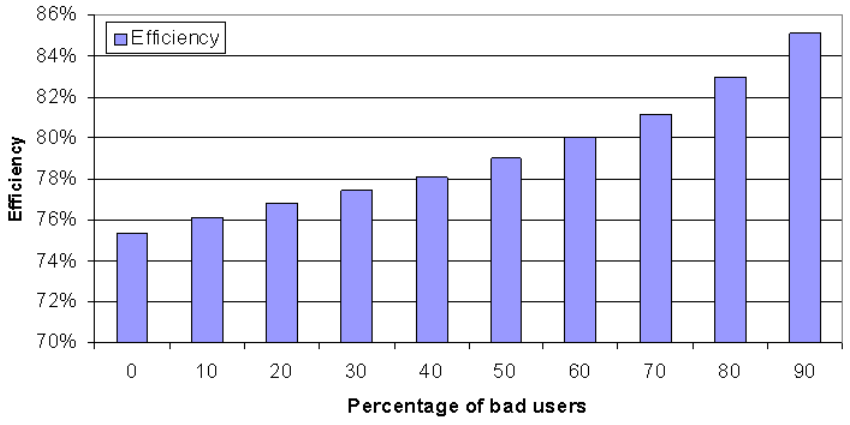


Figure 4-22 Effects of Peer Behavior on Efficiency and Cost

In all cases, the cost of the TMS operation was near zero because of the selection of TS size. The cost was independent of the bad user percentage because the TMS had to perform an introduction before it could ascertain that the prospective associate was trustworthy or not. This requirement reinforced the fact that the cost was linked to the interactivity rates in the network rather than the profiles or behavior characteristics of the users.

4.7 Conclusions

This chapter presented the results and analysis of the verification and validation testing performed on the TMS. The goals and objectives of verification testing were explained in Section 4.1. This testing used simulation scripts to test the performance of individual module functions. Starting with mobility and interactivity traces, these scripts applied the behavior models, profiles, and procedures discussed in Section 3.3.

Section 4.2 analyzed the work accomplished on the TS. First, a comparison of memory management algorithms showed that the TS operated more efficiently using the FIFO method than the LRU. After implementing the FIFO, tests were conducted to determine the optimal size of the TS. Optimizing the TS to minimize introductions was intended to reduce the operating costs of the TMS. It was determined that the number of introductions was directly related to the interactivity rates of the test scenario. Attempting to reduce the number of introductions by increasing the size of the TS met with mixed results, as the decrease in cost was matched by a decrease in system efficiency. What was found, however, was that an optimal point could be determined by finding the tradeoff value between efficiency and cost. In future, this analysis also showed the benefit of using smaller store sizes rather than larger storage allocation, should cost be proven to be less of a concern because of network capacity or nodal capability.

The trust store also introduced two novel tools that were used to optimize the process of gathering and maintaining trust records. First, a network density estimate was created. This estimate allowed a TS to be sized according to the network it was operating in rather than a "bigger is better" approach to consuming memory blocks. Once the trust store was established, the credential exchange process was implemented to "introduce" users to each other. The introduction process provided each user with evidence of the potential associate's behavior and allowed a user to calculate reputation values without having to rely on unsubstantiated referrals, as is common in most other systems.

Section 4.3 used the TS sizing exercise as the starting point for the evaluation of the 3Win reputation scaling method. The RSM was designed to calculate a conservative reputation for peer nodes, based on the observations and reports received by the calculating node. The unique 3Win method of assessing reputation that allowed behavior grades to be dynamically evaluated to reflect changes in reporting nodes' trust levels before calculating the potential associate's reputation. The 3Win reputation scaling method provided the desired performance and showed itself able to detect and react to different, typical, network situations. These results were critical, as the RSM represented a substantial piece of the TMS implementation.

While the RSM determined the trustworthiness of a specific associate, the RAM gauged the amount of uncertainty in the network and adjusted the user's trust profiles for increased protection. The dynamically tuned thresholds were more effective than systems constrained to fixed thresholds, which was the norm. The tuning process resulting in more accurate access control decisions. Reinforcing the increased accuracy was the fact that this improvement was implemented without increasing the cost associated with making those decisions. Section 4.4 showed that the RAM never allowed the user to become self-isolated, even when the network was populated with 100% bad users. These adjustments resulted in increased protection for the TMS and more accuracy in judging the trustworthiness of a prospective associate.

The chapter began by enumerating the goals and objectives for this phase of the analysis of the TMS. Section 4.5 described the operational scenario that was used to frame the simulations. Creation of this scenario was invaluable in eliminating a number of assumptions concerning user interaction that would have otherwise left the results open to charges of being arbitrarily configured. The scenario was translated into a simulation environment and used as the basis for the creation of the previously mentioned scripts.

Section 4.6 discussed the implementation of the system metric. Consisting of a success or efficiency measure and a cost factor, this metric determined how accurate and how expensive the TMS was. As expected, many of the results found during the verification of the individual modules appeared during system validation. The TMS was compared to a base system, which utilized the techniques found in current trust-based access control systems, and performed overwhelmingly better. The efficiency of the TMS was almost five times greater than that of the base system. In addition, the cost of this efficiency was low as the presence of a memory-enabled reputation scaling mechanism and adjustable trust thresholds reduced the number of information exchanges required by the system.

This chapter provided ample evidence of the efficient and robust performance of the TMS. The assertions made at the outset of this research were shown to have direct and positive impacts on the ability of a system to use behavior grades as a means of enabling access control. The next chapter discusses the progress toward the implementation of the other system layers and the possible avenues of future investigation.

5 Conclusions

Global events continue to reinforce the need for rapid, trusted information exchange in dynamic collaborative environments (DCEs). Homeland Security Director Michael Chertoff gave testimony about Hurricane Katrina before Congress in February 2006 that exposed the many failures in alerting, updating, and directing concerned agencies to collaborate in the relief effort. What this disaster demonstrated was that the lack of information exchange can actually compound a disaster's effect by slowing appropriate response and confusing the allocation of needed resources. In particular, first-responder organizations need a means to control access to their scarce resources and critical information as they work to provide assistance and inform higher levels of the relief effort.

This research discussed the theoretical basis and practical development of trust-based system for access control in DCEs. Using behavior grading and risk assessment, this work focused on an access control system suitable for deployment in a DCE, such as a disaster relief coalition responding to a civil emergency. The system's performance was evaluated by its ability to accurately grant or deny access to users in a coalition. Component modules of the trust management system were expected to identify and react to network conditions. Part of the reaction was to adjust security parameters and protect the server from unauthorized access while continuing to allow access from authorized users. Simulation scenarios were constructed to model a mobile dynamic network environment, including selfish and malicious node behavior.

This research's primary design goal was to allow users to enter and participate in a DCE while retaining control over the assets they brought to the coalition. Because of the nature of a DCE, pre-configuration of member identities or organizational roles was neither probable nor effective, ruling out the use of currently implemented, centrally managed access control systems. A system that used trust and behavior grading as a basis for access control decisions was flexible enough to allow DCE members the freedom to move and interact while still providing security. When integrated into a system's security architecture, trust-based access control (TBAC) formed a responsive and adaptable defense for the network's resources.

5.1 Significant Contributions

The over-riding contribution of this research was to design a trust-based access control system that combined reputation and risk (representing two distinct types of trust) and a memory to sustain them both to implement a dynamic access control mechanism. Creating a trust-based access control mechanism enabled coalitions to form quickly so that the group could cooperate to work toward shared goals through the fusion of credential-based identification and reputation-based access control system research.

The investigations of reputation aggregation and calculation showed that trust-based systems could track the behavior of users, a criterion that was then applied to access control decisions. The unique 3Win method of assessing reputation that allowed behavior grades to be dynamically evaluated to reflect changes in reporting nodes' trust levels before calculating the potential associate's reputation.

The Trust Management System (TMS) used a risk assessment process, basically assessing the trust of the network as a whole, to adjust the trust thresholds that granted or removed trust. The innovative mechanism of gauging uncertainty and adjusting trust thresholds to protect against the presence of risk. The dynamically tuned thresholds were more effective than systems constrained to fixed thresholds, which was the norm. The tuning process resulting in more accurate access control decisions. Reinforcing the increased accuracy was the fact that this improvement was implemented without increasing the cost associated with making those decisions.

The TMS implemented an innovative combination of risk and reputation to create an access control decision based on past performance and current network conditions. The results of this research demonstrated that trust management provided accurate and effective access control. This combination of a quantitative reputation index and dynamically adjusted trust thresholds provided better security to a DCE member node than other trust-based systems. The TMS was also able to enforce multiple access levels without the administrative and operational burden of implementing and maintaining multiple cryptographic keys, Certificate Revocation Lists, or Access Control Lists.

Through the implementation of the Trust Store, the TMS functionality and accuracy were greatly enhanced by the implementation of a memory capability. This capability allowed the TMS to weight behavior feedback dynamically and continually, resulting in better assessments of the observed peer's behavior. The TS was also the basis for the evidence-based approach that set this research apart from the current state of the art trust systems.

Once the contribution of the individual modules had been established, the implementation of the system metric allowed us to illustrate the effectiveness of the TMS as a whole. Consisting of a success or efficiency measure and a cost factor, this metric determined how accurate and how expensive the TMS was. As expected, many of the results found during the verification of the individual modules appeared during system validation. The TMS was compared to a base system, which utilized the techniques found in current trust-based access control systems, and performed overwhelmingly better. The efficiency of the TMS was almost five times greater than that of the base system. In addition, the cost of this efficiency was low as the presence of a memory-enabled reputation scaling mechanism and adjustable trust thresholds reduced the number of credential exchanges required by the system.

The next section addresses the specific research goals set forth in Chapter 1.

5.2 Research Accomplishments

Trust-based access control was best used in social networking situations. The focus of this research was to estimate the trustworthiness of an associate among a group of peers. These situations typically offered little time for pre-configuration and, since all of the participants were considered equals, had little hierarchy upon which to implement role-based access control (RBAC). These collaborative environments recently began to implement soft security measures, but these measures have been either based on individual sessions or were centrally managed. Soft security users still required protection mechanisms but these mechanisms needed to be based on some characteristic other than identity or role. Without TBAC, systems had to either assign privileges using broad generalizations, such as giving all users the same access, or were forced to contend with the administrative overhead of assigning privileges to each individual.

The benefits presented by the TMS were that TBAC required no pre-configuration, making it adaptable to a wide range of social network and collaborative situations. It was node-centric, which meant that each node calculated reputation and risk information, rather than observing and accepting work done on a central server. It was evidence-based because it relied on behavior grades rather than a reputation value computed by someone else.

Throughout the investigation, the TMS produced correct answers in a wide range of networking and behavior conditions. The end result was an adaptive access control mechanism for DCEs. Section 4.5 showed the accuracy of TBAC by comparing the number of correct answers to the number of incorrect (i.e., false positive and false negative.) In terms of efficiency and cost, we showed how TBAC relied on behavior information shared between nodes. This information exchange was most visible in the introduction process, as well as the routine behavior observations. This cooperative monitoring enhanced the fidelity of the system but the system could live without it. To prove this, we relied on first hand observations only, disregarding any shared information.

The real cost benefit was found in the absence of administration cost. TBAC was a pure form of Discretionary Access Control (DAC), since every node administered its own permissions, evaluating access requests without having to verify externally generated credentials. Because of the node-centric approach, each TMS operated without mandates or requirements from any outside body. This not only empowered the individual users with control over the assets they contribute to the DCE but it also removed the reliance of the DCE on any single user. In place of any subset of super-users, the TMS relied on a cooperative exchange of behavior grades and credential exchanges to grade and refer peers to each other.

The information exchange that made TBAC powerful was also its weakest point. There was a lot of communications overhead in the system, some of which could be minimized by a communications protocol that allowed behavior information to be carried in the headers of other messages. Decreasing the grade sharing also decreased the accuracy but did not completely invalidate the system, as Reputation Scaling Module (RSM) testing showed in Section 4.3.

The workload required by the TMS was well within the storage and computation capabilities of the personal digital assistants (PDAs) currently available in the market today. As stated earlier, communications consumed most of the time and energy of the node. If the nodes were as powerful as a generic laptop computer, the strain of communications was decreased but, if communicating over wireless networks, might still pose a bottleneck for the user. Nodes of any type connected to a wired network enjoyed the TMS as a transparent service.

The TMS demonstrated that it was effective in reacting to different network risk conditions through the combined application of risk estimation and reputation assessment. The Risk Assessment module (RAM) estimated the uncertainty in a user's network environment in Section 4.4. This estimate was then used to adapt access requirements to protect the user in varying network conditions. We showed how adjusting trust thresholds dynamically was more effective than fixed thresholds. The 3Win method required behavior grades to be reassessed with the observer's current reputation before each reputation calculation. This reassessment enabled the impact of misbehaving users to be minimized quickly. It also helped discourage collusion between misbehaving users, since the decrease in one reputation had a ripple effect on other conspirators.

Despite the apparent focus on misbehavior, the TMS never allowed a user to self-isolation, as this was viewed as a way misbehaving peers could trick a user into denying himself access to resources. The RAM prevented self-isolation through the implementation of a threshold ceiling. In future refinements, it might be expected that the RAM would alert the user to threshold changes to enable him to either physically relocate, change communications channels, or find another forum to collaborate in.

While it might appear that the TMS represented the culmination of work in TBAC, we identified several supporting areas that could be improved or implemented to refine the fidelity of the information the TMS worked upon. Some of these ideas are presented in the following section.

5.3 Future Work

Implemented as a middle layer of a node's system security architecture, the TMS demonstrated that it could provide a node's Key Management System (KMS) with accurate access decisions based on trust. The TMS based these decisions on an aggregation of behavior reports that came from a network monitor or Intrusion Detection System (IDS).

TBAC's success ultimately relies on the ability to identify patterns of behavior and react by adjusting trust. Because the main variable is behavior grading, the need to develop means of recognizing behavior is essential. Future work on the TMS should include constructing a node-centric IDS capable of providing behavior assessments. This function could employ rule-based artificial intelligence as well as user input to grade the behavior of peers.

Once behavior recognition can be implemented, a means of extending the concept of context-sensitive trust should be explored. While almost every researcher acknowledges the importance of context in trust assessments, none of the current systems implement a means of initially differentiating and eventually aggregating peer behavior from different contexts. An agent-based approach that uses context-specific reputation scaling modules while sharing the same trust store and risk assessment modules might provide a lean profile on each node.

5.4 Concluding Thoughts

TBAC is growing in popularity, largely in step with the rise in implementation of social network-focused applications. Whether fielded in a DCE that is responding to a natural disaster, a peer-to-peer network file sharing network, a web portal for school project collaboration, a matchmaking website, TBAC allows users to judge the trustworthiness of potential associates in a virtual society. During the course of this research, interest in TBAC has spawned an international conference specifically for Trust, has seen the initiation of trust management tracks in major ACM and IEEE access control conferences, and the unveiling of a number of social network websites like Outfoxed and Facebook.

The TMS developed in this research is applicable to any of these social network settings. Because of the qualities described above, the system quickly and accurately allows a user to associate with "good" people. The TMS is applicable to any user platform. It is transparent, requiring no user configuration beyond the initial selection of a trust profile. While this research dwelled on only one threshold to provide basic access, there is nothing preventing a user from segmenting their resources and protecting them with any number of different TMS operating under different profiles.

The intent of this research was not to replace systems like RBAC or Bell-LaPadula. They will always have a place in environments like military operations, where hierarchies, administrators, and roles are available to be leveraged. TBAC will, however, continue to proliferate in Internet applications as it fills the gap between pre-configured access control and going without access control. TBAC is adaptable and efficient, allowing a user to participate in a DCE as and where they wish.

Appendix A: Simulation Environment

The following subsections utilized the scenario developed in Section 4.5.1

Test Objectives

Task 1: A user should be able to enter the community.
Condition: A user enters a location with an established identity.
Standard: The user joins the community and can interact with altruistic users or the control plane until he or she establishes a reputation with other users.

Task 2: A user should be able to meet another user through the introduction process.
Condition: A community member meets another community member and wants to establish an association. Other members, known to one or both of the prospective associates as trusted peers, are available to provide references.
Standard: The prospective associates request and receive information on each other from their trusted peers. This information is processed to determine the reputation index of each other.

Task 3: A user should be able to recognize another user from his organization without need of introduction.
Condition: A community member encounters another community member and determines (through examination of their name) that the encountered user is from the same organization.
Standard: These community members can become trusted peers without requiring an introduction process.

Task 4: A user should be able to recognize another user who was previously trusted.
Condition: A community member encounters a former associate. The former associate's credentials are still in the member's trust store.
Standard: The community member should recognize the former associate and reestablish the association (if the associate's RI is acceptable) without requiring an introduction.

Task 5: A user should be able to recognize another user who was previously distrusted.
Condition: A community member encounters a former associate. The former associate's credentials are still in the member's trust store.
Standard: The community member should recognize the former associate and avoid establishing the association without requiring an introduction.

Task 6: A user should be able to move between sites (i.e., geographically separate sub-networks) and continue to operate.
Condition: A user enters a location with an established identity.
Standard: The user joins the community and can interact with established trusted peers, members of their own organization, altruistic users, or the control plane until he or she establishes a reputation with other users.

Task 7: A user's trust thresholds should respond to less risky network communities by lowering the distrust threshold for the period of time the user is in that neighborhood.
Condition: A user is operating in a network location and is listening to control plane messages.
Standard: The user can estimate the amount of risk at the location and adjust the distrust threshold for greater accessibility.

Task 8: A user's trust thresholds should respond to risky network communities by raising the distrust threshold for the period of time that the user is in the neighborhood.
Condition: A user is operating in a network location and is listening to control plane messages.
Standard: The user can estimate the amount of risk at the location and adjust the distrust threshold for greater protection.

Test Vignettes

Dave enters the Tactical Operations Center (TOC).

Dave meets Alice, the task force operations officer.

Dave meets Alex, the task force engineer.

Dave moves to the Balcony Dam site (good location) to perform an assessment.

Dave meets Alex's colleague Bob at the Dam. Bob shares the most recent assessment. Dave verifies this assessment report, adding annotations, and shares this information with Bob and Alex.

Dave meets Boris, a contractor who was supposed to have completed work on the Dam. Dave receives recommendations from Bob and Alex and decides not to extend trust to Boris, citing confidentiality of the information.

Dave moves to the Public Information Center (PIC) (risky location) to deliver a synopsis of the report to Don, the Public Affairs Officer (PAO), to use in that day's briefing. Another user, Natasha, asks for the synopsis. She is unknown to Dave and he declines.

Dave returns to the TOC and delivers the complete report to Alice.

Table A-1 Mapping of Test Vignettes to Objectives

\ Objective Vignette \	1	2	3	4	5	6	7	8
Dave enters the TOC	X							
Dave meets Alice		X						
Dave meets Alex		X						
Dave moves to the Balcony Dam site (good location)						X	X	
Dave meets Bob								
Dave meets Boris								
Dave moves to the Public Information Center (PIC) (risky location)						X		X
Dave meets Don			X					
Dave returns to the TOC						X		
Dave meets Alice				X				
Dave meets Boris					X			

References

Abdul-Rahman, A. and S. Hailes (2000). Supporting trust in virtual communities. Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, 4 - 7 Jan 2000, Pp. 1 - 9.

Aberer, K. and Z. Despotovic (2001). Managing trust in a peer-2-peer information system. Proceedings of the 10th International Conference on Information and Knowledge Management, Atlanta, GA, Pp. 310 - 317.

Adams, W. J. and N. J. Davis (2005). Toward a Decentralized Trust-based Access Control System for Dynamic Collaboration. Proceedings of the 6th Annual IEEE Workshop on Information Assurance, West Point, NY, 15 - 17 June 2005, Pp. 317 - 324.

Adams, W. J., G. C. Hadjichristofi, et al. (2005). Calculating a Node's Reputation in a Mobile Ad-Hoc Network. Proceedings of the 24th IEEE International Performance Computing and Communications Conference (IPCCC 2005), Phoenix, AZ, 6 - 9 April 2005, Pp. 303 - 307.

AFSC (1997). AFSC Pub 1: Joint Staff Officer's Guide. Norfolk, VA, Armed Forces Staff College.

Bartram, L. and M. Blackstock (2003). "Designing Portable Collaborative Networks." Queue **1**(3): 40 - 49.

Ben Salem, N., J.-P. Hubaux, et al. (2004). Reputation-based Wi-Fi deployment protocols and security analysis. Proceedings of the 2nd ACM International Workshop on Wireless Mobile Applications, Philadelphia, PA, Pp. 29-40.

Binnendijk, H. and S. Johnson (2004). Transforming for Stabilization and Reconstruction Operations. National Defense University, 2004.

Biskup, J. and S. Wortmann (2004). Towards a credential-based implementation of compound access control policies. Proceedings of the 9th ACM Symposium on Access Control Models and Technologies, Yorktown Heights, NY, Pp. 31 - 40.

Blaze, M., J. Feigenbaum, et al. (1999). The Role of Trust Management in Distributed Systems Security. Secure Internet Programming: Security Issues for Mobile and Distributed Objects. V. a. Jensen, Springer-Verlag.

Blaze, M., J. Feigenbaum, et al. (1996). Decentralized trust management. Proceedings of the 1996 IEEE Symposium on Security and Privacy, Oakland, CA, 6-8 May 1996, Pp. 164 - 173.

Blaze, M., J. Ioannidis, et al. (2002). "Trust Management for IPSEC." ACM Transactions on Information and System Security **5**(2): 95-118.

Bryce, C., N. Dimmock, et al. (2005). Towards an Evaluation Methodology for Computational Trust Systems. Proceedings of the Third International Conference in Trust Management (iTrust 2005), Paris, FR, Pp. 289-304.

Buchegger, S. and J.-Y. Le Boudec (2002a). Nodes bearing grudges: towards routing security, fairness, and robustness in mobile ad hoc networks. Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, 2002., Canary Islands, 9-11 Jan 2002, Pp. 403-410.

Buchegger, S. and J.-Y. Le Boudec (2002b). Performance analysis of the CONFIDANT protocol. Proceedings of the 3rd ACM International Symposium on Mobile Ad-Hoc Networking and Computing, Lausanne, Switzerland, Pp. 226 - 236.

Buchegger, S. and J.-Y. Le Boudec (2003a). The effect of rumor spreading in reputation systems for mobile ad-hoc networks. Proceedings of the Workshop for Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt '03), Sophia-Antipolis, FR, 3-5 Mar 03, Pp. 1-10.

Buchegger, S. and J.-Y. Le Boudec (2003b). A Robust Reputation System for Mobile Ad-Hoc Networks. Technical Report, Ecole Polytechnic Federal de Lausanne, July 2003.

Buttyan, L. (2000). "Removing the Financial Incentive to Cheat in Micropayment Schemes." IEEE Letters **36**(2).

Buttyan, L. and J.-P. Hubaux (2000). Enforcing service availability in mobile ad-hoc WANs. Proceedings of the 1st ACM International Symposium on Mobile Ad-Hoc Networking and Computing, Boston, MA, USA, Pp. 87 - 96.

Buyya, R., H. Stockinger, et al. (2001). Economic Models for Management of Resources in Peer-to-Peer and Grid Computing. Proceedings of the International Symposium on the Convergence of Information Technologies and Communications (ITCom 2001), Denver, CO, 20 - 24 August 2001, Pp. 1-13.

Byrd, G., F. Gong, et al. (2001). Yalta: A secure collaborative space for dynamic coalitions. Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, West Point, NY, 5 - 6 June 2001, Pp. 30 - 37.

Camp, T., J. Boleng, et al. (2002). "A Survey of Mobility Models for Ad Hoc Network Research." Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications **2**(5): 483 - 502.

Capkun, S., L. Buttyan, et al. (2002). Self-Organized Public-Key Management for Mobile Ad Hoc Networks. Proceedings of the ACM International Workshop on Wireless Security, WiSe 2002, Pp.

Capkun, S., J.-P. Hubaux, et al. (2003). Mobility helps security in ad hoc networks. Proceedings of the 4th ACM International Symposium on Mobile Ad-hoc Networking & Computing, Annapolis, Maryland, USA, Pp. 46 - 56.

Chadwick, D. and A. Otenko (2002). PERMIS X.509 role based privilege management infrastructure. Proceedings of the 4th ACM International Symposium on Mobile Ad-Hoc Networking and Computing, Monterey, CA, Pp. 135 - 140.

Chadwick, D. and O. Otenko (2003). A Comparison of the Akenti and PERMIS Authorization Infrastructures. Proceedings of the ITI First International Conference on Information and Communications Technology (ICICT 2003), Cairo, EG, Pp. 5 - 26.

Commander_Taco. (2005). "Slashdot." Retrieved 15 December, 2005, from http://slashdot.org.

DaSilva, L. A., S. F. Midkiff, et al. (2004). "Network mobility and protocol interoperability in ad hoc networks." Communications Magazine, IEEE **42**(11): 88-96.

Datta, A., M. Hauswirth, et al. (2003). Beyond "web of trust": Enabling P2P E-commerce. 2003 IEEE International Conference on Electronic Commerce (CEC 2003) Newport Beach, CA, 24-27 June 2003, Pp. 303-312.

Datta, A., S. Quarteroni, et al. (2004). Autonomous Gossiping: A self-organizing epidemic algorithm for selective information dissemination in mobile ad-hoc networks. Technical Report, Ecole Polytechnique Federal de Lausanne, June 2004.

de Waal, C. and M. Gerharz. (2005). "BonnMotion." from http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/.

Dewan, P. and P. Dasgupta (2004). Securing Reputation Data in Peer-to-Peer Networks. Proceedings of the International Conference on Parallel and Distributed Computing and Systems (PDCS 2004), Cambridge, MA, Pp. 1-10.

Dimmock, N., J. Bacon, et al. (2005). Risk Models for Trust-Based Access Control. Proceedings of the Third International Conference in Trust Management (iTrust 2005), Paris, FR, 23 - 26 May 2005, Pp. 364-371.

Douceur, J. (2002). The sybil attack. Proceedings for the 1st International Workshop on Peer-to-Peer Systems (IPTPS 02), Cambridge, MA, 7 - 8 March 2002, Pp. 251-260.

Ellison, C. (1996). Establishing Identity Without Certification Authorities. Sixth Annual USENIX Security Symposium, San Diego, CA, USA, July 1996, Pp. 67 - 76.

Ellison, C. (1999a). RFC 2692: SPKI Requirements. RFC, Internet Engineering Task Force, Ellison, C. (1999b). RFC 2693: SPKI Certificate Theory. RFC, Internet Engineering Task Force, September 1999.

Estrin, D. (1986). Inter-organization networks: implications of access control: requirements for interconnection protocol. ACM SIGCOMM Conference on Communications Architectures & Protocols, Stowe, VT, USA, Pp. 254 - 264.

Farrell, S. (2002). RFC 3281: An Internet Attribute Certificate Profile for Authorization. RFC, Internet Engineering Task Force (IETF), April 2002.

Feiertag, R., K. Levitt, et al. (1977). Proving Multilevel Security of a System Design. Proceedings of the 6th ACM Symposium on Operating Systems Principles, Pp. 57 - 65.

FEMA (1994a). ICS Position Descriptions and Responsibilities.

FEMA (1994b). Scenario and Incident Action Plan Catalog.

FEMA. (2004). "Incident Command System."   Retrieved 9 November 2004, 2004, from http://training.fema.gov/EMIWeb/IS/is195.asp.

Fenkam, P., S. Dustdar, et al. (2002). Towards an Access Control System for Mobile Peer-to-Peer Collaborative Environments. Proceedings of the 11th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02), Pittsburgh, PA, 10 - 12 June, 2002, Pp. 95-100.

Ferraiolo, D., D. R. Kuhn, et al. (2003). Role-Based Access Control, Artech House, Inc.

Freudenthal, E., T. Pesin, et al. (2002). dRBAC: distributed role-based access control for dynamic coalition environments. Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS 2002). Vienna, AU, 2 - 5 July 2002, Pp. 411-420.

Gambetta, D. (1988). Can we trust trust? Trust, Making and breaking cooperative relations, electronic edition. D. Gambetta, Univ. of Oxford**:** Ch 13, pp 213 - 237.

Gibson, T. (2001). An Architecture for Flexible, High Assurance, Multi-Security Domain Networks. Proceedings of the Network and Distributed Systems Security Symposium, San Diego, CA, February 2001, Pp. 1-10.

Gray, E., P. O'Connell, et al. (2002). Towards a Framework for Assessing Trust-Based Admission Control in Collaborative Ad Hoc Applications. Technical Report, Department of Computer Science, Trinity College Dublin, 2002.

Gray, E., J.-M. Seigneur, et al. (2003). Trust propagation in small worlds. Proceedings of the First International Conference on Trust Management (iTrust2003), Heraklion, Crete, Pp. 239-254.

Hadjichristofi, G. C., W. J. Adams, et al. (2005a). "A Framework for Key Management in a Mobile Ad-Hoc Network." International Journal of Information Technology **11**(2): 31-61.

Hadjichristofi, G. C., W. J. Adams, et al. (2005b). A Framework for Key Management in a Mobile Ad-Hoc Network. Proceedings of the International Conference on Information Technology Coding and Computing (ITCC 05), Las Vegas, NV, 4 - 6 April 2005, Pp. 568-573.

Heidemann, J., K. Obraczka, et al. (1997). "Modeling the performance of HTTP over several transport protocols." Networking, IEEE/ACM Transactions on **5**(5): 616-630.

Hertzberg, L. (1988). "On the Attitude of Trust." Inquiry **31**(3): 319.

Housley, R., W. Ford, et al. (1999). RFC 2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. Internet Engineering Task Force (IETF), January 1999.

Hubaux, J.-P., L. Buttyan, et al. (2001). The Quest for Security in Mobile Ad Hoc Networks. Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001), San Diego, CA, Pp. 146 - 155.

Jain, R. (1991). The Art of Computer Systems Performance Analysis. New York, NY, John Wiley & Sons.

Jøsang, A. and S. L. Presti (2004). Analysing the Relationship between Risk and Trust. Proceedings of the 2nd International Conference in Trust Management (iTrust 2004), Oxford, UK, 29 March - 1 April 2004, Pp. 135 - 145.

Kagal, L., T. Finin, et al. (2001). A framework for distributed trust management. Proceedings of International Joint Conference on Artificial Intelligence (IJCAI-01), Workshop on Autonomy, Delegation and Control, 2001, Seattle, WA, Pp. 73-80.

Kagal, L., J. Undercroffer, et al. (2002). Vigil: Providing Trust for Enhanced Security in Pervasive Systems. Technical Report, University of Maryland, Baltimore County, August 2002.

Keser, C. (2003). "Experimental games for the design of reputation management systems." IBM Systems Journal **42**(3): 498 - 506.

Khurana, H. and V. Gligor (2000). Review and Revocation of Access Privileges Distributed with PKI Certificates. Proceedings of the Security Protocols Workshop, Cambridge, UK, April 2000, Pp. 100-112.

Khurana, H., V. Gligor, et al. (2002). Reasoning about joint administration of access policies for coalition resources. Proceedings of the 22nd International Conference on Distributed Computing Systems, 2002, Vienna, Austria, July 2002, Pp. 429-438.

Krishnan, R., M. Smith, et al. (2003). "Economics of Peer-to-Peer Networks." Journal of Information Technology Theory and Application **5**(3): 31-44.

Kuro5hin. (2005). "Kuro5hin." Retrieved 15 December, 2005, from http://www.kuro5hin.org.

Laird, J. and R. Wray (2003). Variability in Human Behavior Modeling for Military Simulations. Proceedings of the 2003 Conference on Behavior Representation in Modeling and Simulation (BRIMS), Scottsdale, AZ, Pp. 1-10.

Levien, R. (2004). Attack Resistant Trust Metrics. University of California, Berkley, Li, N. and J. C. Mitchell (2003). RT: a Role-based Trust-management framework. Proceedings of the DARPA Information Survivability Conference and Exposition, 2003., Pp. 201-212 (vol.201).

Li, N., J. C. Mitchell, et al. (2002). Design of a role-based trust-management framework. Proceedings of the 2002 IEEE Symposium on Security and Privacy, 2002., Pp. 104-120.

Liu, J. and V. Issarny (2004). Enhanced Reputation Mechanism for Mobile Ad Hoc Networks. Proceedings of 2d International Conference of Trust Management (iTrust 2004), Oxford, UK, 29 March - 1 April 2004, Pp. 48 - 62.

Lo Presti, S., M. Butler, et al. (2005). A Trust Analysis Methodology for Pervasive Computing Systems. Trusting Agents for trusting Electronic Societies. R. Falcone, S. Barber, J. Sabater and M. Singh, Springer.

Luhmann, N. (1979). Trust and Power, Wiley.

Marsh, S. (1994) Formalising Trust as a Computational Concept, Ph.D. Dissertation, Department of Mathematics and Computer Science, University of Stirling,

Marti, S., T. Giuli, et al. (2000). Mitigating routing misbehavior in mobile ad hoc networks. Proceedings of the 6th ACM International Symposium on Mobile Ad-Hoc Networking and Computing, Boston MA, Pp. 255 - 265.

McKnight, D. and N. Chervany (1996). The Meanings of Trust. Technical Report, Carlson School of Management, University of Minnesota, 1996.

McNett, M. and G. Voelker (2004). Access and mobility of wireless PDA users. Technical Report, University of California, San Diego, February 2004.

Michalakopoulos, M. and M. Fasli (2005). On Deciding to Trust. Proceedings of the Third International Conference on Trust Management (iTrust2005), Paris, FR, 23-26 May 2005, Pp. 61 - 76.

Michiardi, P. and R. Molva (2002a). CORE: A Collaborative Reputation Mechanism to enforce node cooperation in Mobile Ad-hoc Networks. Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security, Pp. 107 - 121.

Michiardi, P. and R. Molva (2002b). Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks. Proceedings of the European Wireless Conference (EW2002), Florence, IT, Pp. 1-6.

Molva, R. and P. Michiardi (2003). Security in Ad hoc Networks. Proceedings of the Personal Wireless Communication, PWC 03, Venice, IT, 23-25 Sep 03, Pp. 756-775.

Montenegro, G. and C. Castelluccia (2004). "Crypto-based identifiers (CBIDs): Concepts and applications." ACM Transactions on Information and System Security **7**(1): 97 - 127.

Moreton, T. and A. Twigg (2003a). Enforcing Collaboration in Peer-to-Peer Routing Services. Proceedings of the First International Conference on Trust Management, Heraklion, Crete, May 2003, Pp. 255-270.

Moreton, T. and A. Twigg (2003b). Trading in Trust, Tokens, and Stamps. Proceedings of the Workshop on Economics of Peer to Peer Systems, Berkeley, CA, Pp. 1-6.

Mui, L. (2003) Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks, Doctoral Dissertation, Electrical Engineering and Computer Science, Massachusetts Institute of Technology,

Nykänen, T. (2000). Attribute Certificates in X.509. Helsinki University of Technology,

Odlyzko, A. (1999). Paris Metro Pricing: The Minimalist Differentiated Services Solution. Proceedings of the 7th IEEE/IFIP International Workshop on Quality of Service, London, UK, 31 May - 4 Jun 1999, Pp. 159-161.

Osborn, S., R. Sandhu, et al. (2000). "Configuring role-based access control to enforce mandatory and discretionary access control policies." ACM Transactions on Information and System Security **3**(2): 85 - 106.

Ost, A. (2001). Performance of communication systems: a model based evaluation with matric geometric methods. New York, Springer.

Perich, F., J. Undercroffer, et al. (2004). In Reputation We Believe: Query Processing in Mobile Ad-Hoc Networks. Proceedings of the International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2004), Boston, MA, Pp. 326-334.

Perry, M., K. O'Hara, et al. (2001). "Dealing with mobility: understanding access anytime, anywhere." ACM Transactions on Computer Human Interaction **8**(4): 323-347.

Phillips, C., T. Ting, et al. (2002). Information sharing and security in dynamic coalitions. Proceedings of the 7th ACM Symposium on Access Control Models and Technologies, Monterey, CA, Pp. 87 - 96.

Powazek, D. (2003). "Gaming the system: How moderation tools can backfire." Retrieved 15 December, 2005, from http://designforcommunity.com/essay8.html.

Prietula, M. (2000). Advice, Trust, and Gossip Among Artificial Agents. Simulating Organizational Societies: Theories, Models, and Ideas. A. Lomi and E. Larsen. Cambridge, MA, MIT Press.

Prietula, M. and K. Carley (2001). Boundedly rational and emotional agents - cooperation trust and rumor. Trust and Deception in Virtual Societies. C. Castelfranchi and Y.-H. Tan. Norwood, MA, Kluwer Academic Publisher**:** 169 - 193.

Rasmussen, L. and S. Jansson (1996). Simulated Social Control for Secure Internet Commerce. Proceedings of the ACM New Security Paradigm Workshop, Lake Arrowhead, CA, Pp. 18 - 25.

Resnick, P., K. Kuwabara, et al. (2000). "Reputation Systems." Communications of the ACM **43**(12): 45 - 48.

Rivest, R. and A. Shamir (1996). PayWord and MicroMint: Two simple micro-payment schemes. Technical Report, MIT Laboratory for Computer Science, 7 May 1996.

Sandhu, R. S., E. J. Coyne, et al. (1996). "Role-based access control models." Computer **29**(2): 38-47.

Seigneur, J.-M., S. Farrell, et al. (2003). End-to-end Trust Starts with Recognition. Proceedings of the First International Conference on Security in Pervasive Computing, 2003, Dallas, TX, 23 - 26 March 2003, Pp. 130-142.

Shah, S. (2005). "Meatball Wiki: Soft Security." Retrieved 15 December, 2005, from http://www.usemod.com/cgi-bin/mb.pl?SoftSecurity.

Shin, D., G.-J. Ahn, et al. (2002). Role-based EAM Using X.509 Attribute Certificate. Proceedings of Sixteenth Annual IFIP WG 11.3 Working Conference on Data and Application Security, King's College, Cambridge University, UK, 29 - 31 July 2002, Pp. 1 - 12.

Shmatikov, V. and C. Talcott (2003). Reputation-based trust management. Proceedings of the Workshop on Issues in the Theory of Security (WITS), 2003, Warsaw, PL, Pp.

Solomon, R. and F. Flores (2001). Building Trust. New York, NY, Oxford University Press.

Spring, S., D. Gormley, et al. (2000). Information Sharing for Dynamic Coalitions. VPSR Report, Verdian Pacific-Sierra Research, December 2000.

Stajano, F. and R. Anderson (1999). "The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks." Computer **35**(4): 22 - 26.

Sturgeon, J. (2004). A question of readiness. Roanoke Times. Roanoke, VA**:** 1.

Suryanarayana, G. and R. Taylor (2003). A Survey of Trust Management and Resource Discovery Technologies in Peer-to-Peer Applications. Technical Report, University of California, Irvine, July 2004.

Thompson, M., A. Essiari, et al. (2003). "Certificate-based authorization policy in a PKI environment." ACM Transactions on Information and System Security **6**(4): 566 - 588.

USCG. (1999, January 1999). "On-Scene Command and Control." Retrieved 9 November 2004, 2004, from http://www.uscg.mil/hq/gm/mor/articles/osc2.htm.

Young, S. and D. Aitel (2004). The Hacker's Handbook, Auerbach Publications (CRC Press).

Yu, B. and M. Singh (2003). Incentive Mechanisms for Peer-to-Peer Systems. Proceedings of Second International Workshop on Agents and Peer-to-Peer Computing, Melbourne, AU, Pp. 77 - 88.

Yu, T., M. Winslett, et al. (2003). "Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation." ACM Transactions on Information Systems Security **6**(1): 1 - 42.

Zacharia, G. and P. Maes (2000). "Trust Management through Reputation Mechanisms." Applied Artificial Intelligence **14**: 881 - 907.

Vita

Lieutenant Colonel William J. Adams is an Information Systems Engineer in the United States Army Signal Corps. After receiving his Ph.D. in Computer Engineering from Virginia Polytechnic Institute and State University, he will become a research scientist in the U.S. Army's Information Technology Center for Excellence at West Point, NY. A native of Staunton, VA, Lieutenant Colonel Adams has a Bachelor of Science degree in Computer Engineering from Syracuse University and a Master of Science degree in Computer Systems Engineering from the University of Arkansas. Following his graduate studies, he was assigned to the United States Military Academy as an Assistant Professor in Computer Science. In addition to his academic achievements, he has commanded a communications-electronics installation company and served as a regional network operations and plans manager for the Defense Department's networks in the Benelux. In his most recent assignment, Lieutenant Colonel Adams was a project leader and chief engineer for the Supreme Headquarters, Allied Powers Europe. His academic interests include network and mobile security and digital libraries. Some of his publications include:

W. Adams and N. Davis (2006), "TMS: A Trust Management System for Access Control in Dynamic Collaborative Environments," Proceedings of the 25th IEEE International Performance Computing and Communications Conference (IPCCC 2006), 10 – 12 April 2006.

W. J. Adams and N. J. Davis (2005). "Toward a Decentralized Trust-based Access Control System for Dynamic Collaboration," Proceedings of the 6th Annual IEEE Workshop on Information Assurance, West Point, NY. 13 – 15 June 2005, Pp 317 - 324.

W. Adams, G. Hadjichristofi, N. Davis. (2005) "Calculating a Node's Reputation in a Mobile Ad-Hoc Network," Proceedings of the 24th IEEE International Performance Computing and Communications Conference (IPCCC 2005), 6 – 9 April 2005, Pp. 303 - 307.

G. Hadjichristofi, W. Adams, and N. Davis. (2005). "A Framework for Key Management in a Mobile Ad-Hoc Network." International Journal of Information Technology 11(2): 31 - 61.

G. Hadjichristofi, W. Adams, N. Davis. (2005) “A Framework for Key Management in a Mobile Ad-Hoc Network,” Proceedings of the International Conference on Information Technology Coding and Computing (ITCC 05), 4 – 6 April 2005, Pp. 568 – 573.